

Michael Anders

Second Edition

# Surveillance and the Art of Digital Camouflage

Use free, open source tools:

TOR, TOR-Browser, I2P, TAILS, VPN-Gate, SoftEther,  
OpenVPN, Ricochet, Susimail, TorBirdy, Enigmail, GnuPG,  
Academic-Signature, OTR, OnionShare,...

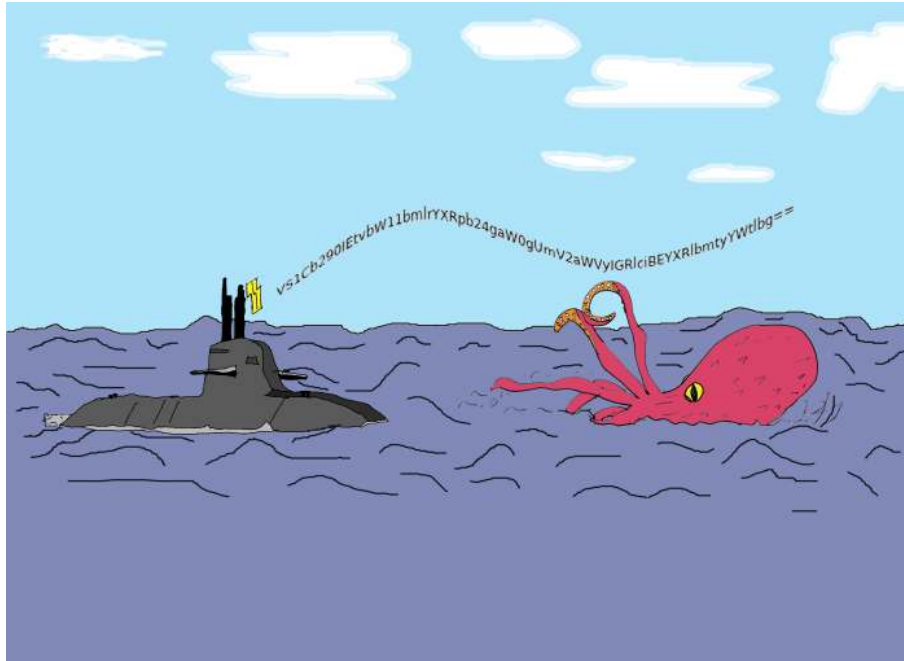
**Surveillance**  
and the Art of Digital Camouflage  
Second Edition

Michael Anders

January 2021

Für meine Kinder.

I wrote this book with great admiration for Edward Snowden, Hans and Sophie Scholl, and all the other people who can tell right from wrong and who are brave enough to speak out within striking distance of the monster.



You see the cover picture of the first German edition of this book titled: "Submarine Communication in the Estuary of the Data Kraken" (in German: "U-Boot Kommunikation im Revier des Datenkraken"). Here the Data Kraken stands for the big "be all ears" corporations like Alphabet (Google), Apple, Alibaba, Microsoft, Facebook, and government snooping agencies like the NSA, GCHQ, or the German BfV.

I wrote this book for laypeople using computers for communicating with friends or for securely browsing the Internet. Compared to a system used at the workplace and administered by the employer's IT-staff, using the private computer has higher security risks but allows for substantially more options and chances regarding techniques to protect your privacy. The book will teach you how to use this freedom and to seize these chances. You will be able to safeguard your private communication much better against eavesdropping and message manipulation by criminal organizations or even by nation-state agencies than workplace communication using your employer's systems.

### **The author:**

Prof.Dr. Michael Anders is a physicist and teaches at the University of Applied Sciences Wedel. For many years, the author served as dean of the Department of Industrial Engineering. He taught seminars on online anonymity, Darknet, and encryption. In the year 2011, Michael Anders developed and published the encryption tool Academic Signature. Using Academic Signature, users can create digital signatures, create timestamps, and encrypt files using the elliptic curve cryptosystem (ECC). The author published Academic Signature and its source code under the GNU General Public License (<https://www.academic-signature.org>).

## **Preface to the first English edition of this book:**

I initially published this book in the German language. The reader will quickly note that the English language is not my mother tongue. (Let me try to stick to American English - kudos to my long-lost fiancée from Seattle.) I apologize for presenting you with a text written in my clumsy pidgin English. I am a scientist, not a professional writer. Please forgive me.

Yet I dared to publish this first English edition of this book because its content should, in my opinion, be accessible to educated people all over the world. Writing this book has not been motivated commercially. I wish to empower, educate, and train the readers. Empowerment, education, and training require effort. However, most people will shy away from putting substantial work into learning to evade surveillance, despite being interested in avoiding digital intrusion in principle. Those people will be happier reading a book expressing some outrage and lamentation regarding ubiquitous surveillance, offering some simple advice that is easy to read, never needs to be rehearsed, and is not very impactful when deployed in practice.

There are plenty of such books, and they sell in large numbers. I wanted this book to be different. Although you only need to read and understand some theoretical sections, working with most other parts will require a running computer at your side, your gaze switching between the computer screen and book pages, and your fingers on the keyboard. I will encourage you to double-check assertions from this book against the results of your Internet inquests, to reproduce findings with your system, to install and use free, open-source software for your digital defense, and to practice challenging modes of communication. Do not just believe statements from this guidebook. Always check against your state of knowledge. Experiment with your system to verify or falsify claims. That will exhaust your ability to stay focused. Occasionally you may need a break to overcome mental inanition. Take those breaks!

The methods presented in this book are inherently defensive. Yet mastering the techniques for stealth Internet communication can give you some power. Never use this power to harm anyone by anonymous threats, personal exposures, insults, theft, Internet fraud, blackmail, stalking, or by covering conspiracy for violent action. Always respect digital privacy and preserve the digital inviolacy of all other people. Give support to people in need of protection in the digital world. Everyone may strive for learning digital camouflage. Of course, this book is open to anyone. Yet, I wish religious and political extremists, especially from the xenophobic right-wing sector, would stay away from this book. All in all, computers and the Internet are great inventions with great potential to serve humankind. They let us communicate across all borders, break free of oppression, and make the world a better place.

Michael Anders,  
Klein Nordende, Germany, in August 2020.

**Legal matters:**

I published this book under copyright protection. In writing, however, I built on great work by many volunteers who have devoted their time to the development, maintenance, and documentation of free software accessible under a public license. Thus I have mixed feelings about the copyright protection of this book and would like to apologize for publishing the book under this legal status.

I try to convince the readers of the potency of security techniques put forward in this document. My goal is to empower them. They shall come to informed decisions about what measures to apply to secure their digital communication. Readers use any of the techniques described in this book on their assessment, their decision, and their own risk. To my best knowledge, the use of the described methods is legal in Germany and the USA. Using them may not be legal in your home country or may not be legal anymore in the USA or Germany tomorrow. Please find out the present legal situation in your home country regarding private citizens using anonymization techniques and potent encryption. Don't use these techniques if your government banned them for you. With a saddened heart, I recommend subordinating to the antidemocratic, inhuman, and abasing rule of your government in this case. In some democratic European nations, the use and distribution of tools for potent encryption is only legal for entities owning a respective government license and thus banned for regular citizens. However, I am not aware of any attempt to enforce this ban in practice in any European nation.

I have compiled all the information contained in this book with due care. Nevertheless, I cannot exclude errors or outdated information. Let me clarify that I will not take any responsibility and shall under no circumstances be held liable for damage resulting from inaccurate information or mistakable wording. Additionally, I will not take responsibility for reader harassment by the local authorities in case they catch him employing techniques, which are considered unlawful or unwanted in the reader's home country.

Similarly, I strongly advise consuming freshly brewed coffee while reading this book. You might scald yourself. As a matter of course, I will be thankful for my reader's hints regarding inaccurate information in this book, any need for updating the contents, or mention of language errors. Software suggested for downloading in this book shall only be privately used. Commercial use will most likely require the consent of the owner of the corresponding license. The names of specific pieces of software or brand names mentioned in the book may be subject to trademark or patent protection without explicit mentioning.

**© 2021 Michael Anders, [michael.anders@academic-signature.org](mailto:michael.anders@academic-signature.org)  
Publishing and distribution: epubli GmbH, Berlin, [www.epubli.de](http://www.epubli.de)**



# Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Using a Self Administered Computer</b>                 | <b>9</b>  |
| 1.1      | Occupational and private use of the computer              | 9         |
| 1.2      | IT vendors' offerings                                     | 10        |
| 1.3      | Grabbing your data from your system                       | 10        |
| 1.4      | Digital privacy in a democracy                            | 12        |
| 1.5      | The situation we are in today                             | 13        |
| 1.6      | The price tag on staying in control                       | 14        |
| 1.7      | Submarine communication                                   | 15        |
| 1.8      | Training for Digital Camouflage                           | 15        |
| <b>2</b> | <b>Reign Supreme</b>                                      | <b>17</b> |
| 2.1      | Be in command   | 17        |
| 2.2      | Safe installation of software                             | 19        |
| 2.3      | Use of the digital signature                              | 20        |
| 2.4      | Tools for reigning supreme                                | 21        |
| 2.4.1    | Open-source operating systems                             | 21        |
| 2.4.2    | Digital signatures: GnuPG                                 | 22        |
| 2.4.3    | Digital signatures: Academic Signature                    | 28        |
| 2.5      | Exercises for reigning supreme                            | 29        |
| 2.5.1    | RSx1, introductory exercises                              | 30        |
| 2.5.2    | RSx2, using Academic Signature's digital signatures       | 31        |
| 2.5.3    | RSx3, using GnuPG's digital signatures                    | 31        |
| 2.5.4    | RSx4, domain certificates                                 | 32        |
| 2.5.5    | RSx5, authenticate via the operating systems' automatisms | 32        |
| 2.5.6    | RSx6, authenticate using GnuPG                            | 33        |



|            |  |           |
|------------|--|-----------|
| 2.5.7      | RSx7, authenticate using Academic Signature . . . . .                          | 33        |
| 2.5.8      | RSx8, install a free hex-editor . . . . .                                      | 34        |
| 2.5.9      | RSx9, encryption using Academic Signature and using GnuPG . . . . .            | 34        |
| <b>2.6</b> | <b>Attacks on your reigning supreme</b>  | <b>35</b> |
| 2.6.1      | Systemd (on Linux) . . . . .   | 35        |
| 2.6.2      | Trends among vendors and retailers . . . . .                                   | 36        |
| 2.6.3      | Microsoft reigns supreme on your Windows10 . . . . .                           | 37        |
| <b>3</b>   | <b>Receptive Anonymity . . . . .</b>   | <b>39</b> |
| <b>3.1</b> | <b>What is receptive anonymity?</b>  | <b>39</b> |
| <b>3.2</b> | <b>Geopolitical aspects of anonymous digital communication</b>                 | <b>41</b> |
| <b>3.3</b> | <b>Achieving anonymity by mixing data flows inextricably</b>                   | <b>42</b> |
| 3.3.1      | Simple Internet access . . . . .   | 42        |
| 3.3.2      | The VPN-Server as discreet intermediary . . . . .                              | 44        |
| 3.3.3      | Onion routing, a network of reticent intermediaries . . . . .                  | 49        |
| 3.3.4      | Anonymous access to the web . . . . .  | 51        |
| <b>3.4</b> | <b>Tools for receptive anonymity</b>   | <b>52</b> |
| 3.4.1      | VPN Gate . . . . .   | 52        |
| 3.4.2      | TOR and the TOR-Browser . . . . .  | 53        |
| 3.4.3      | Email access and TorBirdy . . . . .  | 54        |
| 3.4.4      | TAILS . . . . .  | 54        |
| 3.4.5      | I2P "The Invisible Internet Project" . . . . .                                 | 56        |
| <b>3.5</b> | <b>Exercises for receptive anonymity</b>                                       | <b>57</b> |
| 3.5.1      | REcx1, trace your data packets on their way to the destination . . . . .       | 57        |
| 3.5.2      | REcx2, access the Internet via a free VPN . . . . .                            | 58        |
| 3.5.3      | REcx3, safely install the TOR-Browser . . . . .                                | 63        |
| 3.5.4      | REcx4, monitor TOR-Browser's operation . . . . .                               | 64        |
| 3.5.5      | REcx5, combine VPN and TOR-Browser and block script execution . . . . .        | 64        |
| 3.5.6      | REcx6, check your mails without revealing your position . . . . .              | 67        |
| 3.5.7      | REcx7, install TorBirdy in Thunderbird . . . . .                               | 67        |
| 3.5.8      | REcx8, force TOR-Browser to use an exit node in a country of your choice . . . | 68        |
| 3.5.9      | REcx9, find a TOR Darknet search engine for onion sites . . . . .              | 68        |
| 3.5.10     | REcx10, practice browsing the TOR Darknet . . . . .                            | 68        |
| 3.5.11     | REcx11, install and use the anonymizing tool I2P . . . . .                     | 69        |
| 3.5.12     | REcx12, set up two TAILS USB sticks . . . . .                                  | 70        |
| 3.5.13     | REcx13, browse the Darknet on a TAILS . . . . .                                | 71        |
| 3.5.14     | REcx14, setup access to your email account in TAILS . . . . .                  | 72        |
| 3.5.15     | REcx15, read your favorite online news outlet using TAILS and TOR-Browser . .  | 72        |
| 3.5.16     | REcx16, roundup evasion maneuver . . . . .                                     | 72        |
| 3.5.17     | REcx17, transfer your GnuPG key to your TAILS . . . . .                        | 73        |
| 3.5.18     | REcx18, install and use Academic Signature on your TAILS . . . . .             | 73        |
| 3.5.19     | REcx19, spot the MAC address of your device's WLAN interface . . . . .         | 74        |
| 3.5.20     | REcx20, find free software to spoof your device's MAC address . . . . .        | 74        |
| 3.5.21     | REcx21, spoof your device's MAC address using TAILS . . . . .                  | 74        |
| <b>3.6</b> | <b>Attacks on receptive anonymity</b>  | <b>75</b> |
| 3.6.1      | Attack type 1, website fingerprinting . . . . .                                | 76        |
| 3.6.2      | Attack type 1, modulating the data rate with nearby end point . . . . .        | 76        |
| 3.6.3      | Attack type 1, physical attack on known TOR users . . . . .                    | 77        |
| 3.6.4      | Attack type 2, NSA attack, hacking the TOR-Browser . . . . .                   | 78        |

|            |   |            |
|------------|---|------------|
| 3.6.5      | Attack type 2, MITM attack at an exit node . . . . .                          | 78         |
| 3.6.6      | Attack type 3, big data attack on TOR users by statistical analysis . . . . . | 79         |
| 3.6.7      | Attack type 3, big data attack by modulating data rate . . . . .              | 79         |
| 3.6.8      | Attack type 3, trojan nodes . . . . .   | 79         |
| <b>4</b>   | <b>Expressive Anonymity . . . . .</b>   | <b>81</b>  |
| <b>4.1</b> | <b>Fictive identities . . . . .</b>   | <b>81</b>  |
| <b>4.2</b> | <b>Hazards of payments . . . . .</b>  | <b>82</b>  |
| <b>4.3</b> | <b>Tools for expressive anonymity . . . . .</b>                               | <b>83</b>  |
| 4.3.1      | Chat accounts . . . . .   | 83         |
| 4.3.2      | Jabber chat accounts . . . . .  | 84         |
| 4.3.3      | Jabber client Pidgin and TOR . . . . .  | 84         |
| 4.3.4      | Rendezvous type TOR routing . . . . .   | 85         |
| 4.3.5      | Anonymous email account, using a web interface and the TOR-Browser . . . .    | 86         |
| 4.3.6      | Anonymous email account, using shared access to minimize the attack surface   | 88         |
| 4.3.7      | Serverless message exchange using OnionShare . . . . .                        | 89         |
| <b>4.4</b> | <b>Exercises for expressive anonymity . . . . .</b>                           | <b>90</b>  |
| 4.4.1      | EXPx1, Register anonymous chat accounts, use Pidgin to chat anonymously .     | 90         |
| 4.4.2      | EXPx2, Transmit a file anonymously using chat accounts . . . . .              | 93         |
| 4.4.3      | EXPx3, Register email accounts anonymously . . . . .                          | 94         |
| 4.4.4      | EXPx4, Chat using Ricochet via TOR in rendezvous mode (primary anonymity)     | 95         |
| 4.4.5      | EXPx5, Exchange files with your partner via OnionShare (primary anonymity) .  | 95         |
| 4.4.6      | EXPx6, Publish a Darknet website using OnionShare . . . . .                   | 96         |
| 4.4.7      | EXPx7, Search the web for information on TOR rendezvous paths . . . . .       | 97         |
| 4.4.8      | EXPx8, Register an I2P email account using Susimail and exchange emails . .   | 98         |
| 4.4.9      | EXPx9, Practice buying an electronic device anonymously . . . . .             | 98         |
| <b>4.5</b> | <b>Attacks on expressive anonymity . . . . .</b>                              | <b>99</b>  |
| 4.5.1      | Nonymizing information in transmitted data . . . . .                          | 100        |
| 4.5.2      | Big data attacks . . . . .  | 100        |
| <b>5</b>   | <b>Privacy . . . . .</b>  | <b>105</b> |
| <b>5.1</b> | <b>Symmetric encryption . . . . .</b>   | <b>105</b> |
| 5.1.1      | Stretching . . . . .  | 107        |
| 5.1.2      | Salting . . . . .   | 107        |
| 5.1.3      | Estimating cracking cost and cracking times . . . . .                         | 108        |
| 5.1.4      | Privacy aid from government's notorious cost drivers . . . . .                | 111        |
| 5.1.5      | The last iron in the fire . . . . .   | 111        |
| <b>5.2</b> | <b>Asymmetric encryption . . . . .</b>  | <b>112</b> |
| <b>5.3</b> | <b>Hybrid encryption . . . . .</b>  | <b>113</b> |
| <b>5.4</b> | <b>Combining anonymity and confidentiality . . . . .</b>                      | <b>114</b> |
| 5.4.1      | Primary and secondary anonymity in confidential communication . . . . .       | 114        |
| 5.4.2      | Starter communication out of band . . . . .                                   | 115        |
| 5.4.3      | Anonymous encrypted communication with an unknown peer . . . . .              | 116        |
| <b>5.5</b> | <b>Tools for ensuring privacy . . . . .</b>                                   | <b>116</b> |
| 5.5.1      | Supplementing the encryption plugin OTR in Pidgin . . . . .                   | 116        |
| 5.5.2      | Encryption using GnuPG . . . . .  | 119        |
| 5.5.3      | Supplementing the Thunderbird mailer with Enigmail . . . . .                  | 122        |
| 5.5.4      | Encryption using Academic Signature . . . . .                                 | 123        |

|            |  |            |
|------------|--|------------|
| <b>5.6</b> | <b>Exercises for combining anonymity and confidentiality</b>                     | <b>126</b> |
| 5.6.1      | A&Cx1, Symmetric encryption using office software . . . . .                      | 127        |
| 5.6.2      | A&Cx2, Matryoshka doll encryption using (p)7zip . . . . .                        | 127        |
| 5.6.3      | A&Cx3, Symmetric encryption using GnuPG . . . . .                                | 128        |
| 5.6.4      | A&Cx4, Symmetric encryption using Academic Signature . . . . .                   | 129        |
| 5.6.5      | A&Cx5, Using the OTR plugin for encrypting anonymized chats . . . . .            | 130        |
| 5.6.6      | A&Cx6, Clone anonymized access to an OTR protected XMPP chat account             | 131        |
| 5.6.7      | A&Cx7, Supplement fictive email identities with public/private key pairs . . . . | 133        |
| 5.6.8      | A&Cx8, Minimize exploitable information in GnuPG ciphertexts . . . . .           | 134        |
| 5.6.9      | A&Cx9, Minimize exploitable information in Academic Signature ciphertexts        | 135        |
| 5.6.10     | A&Cx10, Anonymous message exchange utilizing shared cloud storage . . .          | 136        |
| 5.6.11     | A&Cx11, Nonymous encrypted email exchange using GnuPG . . . . .                  | 137        |
| 5.6.12     | A&Cx12, Use GnuPG automatisms to send email attachments . . . . .                | 139        |
| 5.6.13     | A&Cx13, Transfer a file between email accounts using manual encryption .         | 141        |
| 5.6.14     | A&Cx14, Transfer a large file anonymously and confidentially via XMPP . . . .    | 142        |
| 5.6.15     | A&Cx15, Flying substitution of chat accounts . . . . .                           | 144        |
| 5.6.16     | A&Cx16, Send a file from an I2P SusiMail account using manual encryption .       | 146        |
| 5.6.17     | A&Cx17, Transfer a file using OnionShare, starter communication out of band      | 148        |
| 5.6.18     | A&Cx18, Transfer a file using OnionShare, appointment driven . . . . .           | 150        |
| <b>5.7</b> | <b>Attacks on the combination of anonymity and confidentiality</b>               | <b>153</b> |
| 5.7.1      | Attack on anonymity and confidentiality: the NOBUS principle . . . . .           | 153        |
| 5.7.2      | Big Data complementary attack, anticorrelation with nonymous activity . . .      | 155        |
| 5.7.3      | Anecdotal evidence for the vulnerability of TOR anonymization . . . . .          | 156        |
| <b>6</b>   | <b>Concluding Remarks</b> . . . . .  | <b>159</b> |
| <b>6.1</b> | <b>Technology, snooping governments, and civilian resistance</b>                 | <b>159</b> |
| <b>6.2</b> | <b>Politics and surveillance</b>   | <b>161</b> |
| <b>7</b>   | <b>Glossary</b> . . . . .  | <b>163</b> |



## 1. Using a Self Administered Computer

### 1.1 Occupational and private use of the computer

At the workplace, our employers' IT-people will ideally care well for the safety of our computers against attacks by criminals but will care lousily for the privacy of our electronic communication. The IT-people will intrusively limit the employees' rights on their workplace computers to improve security against cybercrime. They will also request adherence to some rules. Some of these rules may be reasonable, such as deleting office documents containing macros that come as email attachments. Others may be absurd, such as the commonly enforced password regulations. You undoubtedly know such requirements as to use three unique passwords for three unique computer environments at the workplace. Each of those is supposed to contain a minimum of x capital letters, y numbers, and z special characters where all passwords are to be changed quarterly, at least, but with the strict interdiction to jot them down anywhere.

In return for being bossed around by enterprise IT, however, we can enjoy the comforting lack of responsibility for IT-security or IT-performance. Enterprise IT-security necessarily focuses on different aspects than the safety of private communication. We will mainly treat the latter in the later sections of this book. Let us now elaborate on the differences between workplace and private digital communication.

The primary task in enterprise IT-security is keeping the least motivated and least insightful colleagues from doing something stupid with their systems. IT-security must restrict their options and must narrowly guide their actions to prevent massive damage. If they still cause damage, close surveillance of their activity can at least help to detect and limit this impairment timely. The least knowledgeable and, as collateral damage, also all other employees are deprived of choices and channeled in their workflow. Security measures prevent them, for example, to click in Outlook on email attachments sent by the infamous Nigerian prince. Regular system updates and strict blocking of unneeded communication modes (blockade of ports in the firewall) put a limit on the number of software vulnerabilities exploitable by outside attackers.

Against better judgment, enterprise IT-security ignores nation-state spying agencies as a threat to confidential message exchange and as initiators of damaging data leakage. Most IT-security

people consider data leakage inflicted by the actions of naive or disloyal insiders more threatening. They perceive communicating employees' access to effective End-to-End encryption as a hazard for security in this context. Enterprise IT-security fails to recognize the staff's access to potent encryption as an effective protective measure against industrial espionage. These are very unfavorable conditions for the adequate protection of digital correspondence.

This book, however, has its focus on protecting digital message exchange using privately owned computers. Readers of this book are highly motivated and interested in improving the security of their private communication. Being the author of this guidebook, I may be unable initially to rely on the readers' profound IT know-how or even on expertise in encryption. But I can reasonably expect a vested interest, an alert mind, and an experiment-happy mood. That is a particularly promising starting situation.

## 1.2 The mixed blessing of IT vendors' offerings for computer users

Measures to ensure general IT security demand an effort from the user, and they demand a bit of knowledge to secure and administer their computers. Operating system vendors want to ease the burden for users and lessen their learning effort. They tend to lock their systems down to choke user influence in system maintenance and better prepare their products for vendor-driven remote servicing. Such remote servicing typically occurs in the background without user consent or appropriate user involvement. But such remote servicing will never result in the security and flexibility attainable through servicing and system administration by a versed and knowledgeable human.

Unfortunately, such remote servicing avoiding user involvement also comes with a strong temptation for software vendors. This unchecked access to private computers entices them to all sorts of annoying marketing activities and trading with nicked user data. The disagreeable marketing activities comprise the openly visible part of this looting.

## 1.3 The struggle to grab your data from your system

Some computer users nowadays wish to keep full control of their systems and sovereignty regarding their data. The misuse of vendor power, the diminishing capacity to fully reign private computers, and the all too eager cooperation of software vendors with snooping government agencies of their home countries are vexing those users.

**An example:** In October 2017, security researchers exposed that the library for generating RSA-keys used in many so-called TPMs (Trusted Platform Module) contained a fault<sup>1</sup>.

The lapse caused insecure RSA public-private key pairs. It did not involve the pseudo-random number generator (PRNG), though.

The TPM is a chip built into desktop computers and notebooks. In addition to many other uses, it serves to check digital signatures. The computer's operating system uses it to verify such digital signatures before executing files downloaded for software update or installation. The flawed library is residing in the TPM's firmware. Firmware is software residing on the chip. Special protective measures preclude reading the code from outside the chip and ensure its integrity.

The identifier CVE-2017-15361 and the name ROCA, Return of Coppersmith's Attack, were officially assigned to the newly found vulnerability. Among other suppliers, also the German chip manufacturer Infineon delivered TPMs containing this fault. It did so for five years.

Please note that the TPM must find two long primes to generate a public/private RSA key pair. The search for a prime number proceeds as follows. At first, we select an odd random number of the appropriate bit length. Using a probabilistic test, commonly the Rabin-Miller test, we check for

---

<sup>1</sup>see: [https://crocs.fi.muni.cz/public/papers/rsa\\_ccs17](https://crocs.fi.muni.cz/public/papers/rsa_ccs17)

primality. If the test indicates that the number is composite (not a prime), we increment the number twice and repeat the test. However, if the test reports primality, we found the first prime. Then we start the procedure again with a new odd random number and search for the second prime.

The product of both primes complemented with a smaller number called the public exponent makes up the public RSA key. Knowledge of the two prime factors is the private RSA key.

Now, if the pseudo-random number generator was not affected, as has been proclaimed, the two following questions are imminent. Which unusual criteria does the faulty software on the chip have, not to select the first prime following a chosen random number? Why does it come up with some cherry-picked primes, leading to insecure RSA key pairs? Infineon released the statement this would ensure a faster generation of prime numbers. This statement appears noncredible to me. The TPM generates an RSA key pair once, at most a few times in the chip's lifetime. The proper procedure would last about a minute. I suspect some snooping agency of an influential nation-state coaxed Infineon into introducing a backdoor.

When security researchers at the Masaryk University in Brno, at the British security enterprise Enigma Bridge and the Ca' Foscari University in Venice found the flaw, the suspected nation-state cheaters probably had to act quickly. They had to mitigate the damage, find a replacement for the backdoor, install measures to prevent criminals from exploiting the flaw, and distract journalists from asking critical questions. As a matter of course, the U.S. standards body NIST and the German civil service BSI had certified the affected TPMs as secure.

Knowing the restriction to a small subset of prime numbers, the attacker requires less effort to attack the RSA private key. Even crypto-laypeople will readily understand that. Remember, the assault consists of seeking the prime factors of the public RSA key.

In a sincere audit regarding an RSA implementation, a primary issue would be the technique to generate the used primes and whether they are chosen randomly from the set of all prime numbers. Any other answer than a clear yes to the latter question would make the members of a respectable auditing committee raise their eyebrows close to the hairline. The very fact that this did not happen in the certification procedure of a chip of such significant security relevance is disturbing. Regarding the U.S. standards body NIST and the German civil service BSI, we have to decide whether we consider the participating staff to be saboteurs or incompetent. An article in the German online newspaper "Zeit" cautiously insinuated the second alternative. It described the incident as embarrassing for the German BSI<sup>2</sup>. Unfortunately, investigative journalists did not find themselves in the position to ask further questions.

You need not share my assessment and my critical view of the actions of NIST and BSI. I welcome all readers of this guide, regardless of how they feel about such security incidents. Let me apologize here already to readers considering official bodies and large corporations to be trustworthy and to readers thinking too many people, including me, hang on to conspiracy theories.

Nevertheless, in the following parts of the book, a few valuations may inadvertently slip into the text, which may discomfort you. In the finalizing chapter 6, however, beyond the politically predominantly neutral, technically oriented sections, I will again voice my personal opinions clearly and unmistakably. In the end, my dismay looking at how our political leadership handled the NSA eavesdropping affair and at the politically motivated prosecution of the courageous whistleblower Edward Snowden was my incentive to write this book. Primarily, however, this is a technical book.

The perception of the activities of our western intelligence services and the monopolistic Internet corporations is quite polarized. Many people view us surveillance opponents as a bunch of confused conspiracy theorists. But we are in danger of seeing each critic of open-source software and anyone arguing agency and corporate-friendly as a fascistic proponent of pervasive government

---

<sup>2</sup>See the article in the German language. You may want to run the excellent free DeepL Translator on it to convert to the English language: <https://www.zeit.de/digital/datenschutz/2017-10/infineon-verschlueselung-personalausweis-tpm-bsi-zertifiziert>

surveillance. Digitally crossing swords with our respective adversaries gives us the chance to move closer politically via technically oriented discussions. Opponents of eavesdropping, ripoff, and data nicking by powerful agencies and criminals must understand the attackers' mindset. Only then can we be good at defending freedom and privacy. Additionally, we have to put our familiar concepts of the enemy to the test. Inadequately exaggerated distrust does not improve our defense.

If you belong to the corporate and authorities side and regard us surveillance opponents as paranoid anarchists, you should still try for a moment to see the state of affairs with our eyes. You should put your familiar concepts of us as enemies to the test as well. How would you like to be surveilled by government agencies trying to keep the extent of the surveillance secret? Please scrutinize for a moment if one or the other surveillance measure you welcomed initially may not indeed pave the way into our and your subjugation and a dictatorship. Anyways, I express a friendly welcome to all readers of this book, may they share my political views or not. At any rate, even my political opponents should profit from reading the technical parts of this book.

To conclude, I want to express a warm welcome to those readers I initially envisioned when I began writing. You are intelligent IT laypeople, and you are interested in computer and Internet matters. You have no ideological predetermination, and your assessment of the flaw in your computers' TPM, previously described, is still unsettled. You are aware of some facts, though: We had the Snowden revelations in 2013. Western governments' intelligence agencies had been out of control. We don't know if they are respecting democratic principles today or act worse than ever. Prudence demands to prepare for the worst. Added to this are the ongoing hostile activities of foreign intelligence agencies and the ever-growing threat of cybercrime. Thus, you want to develop your skills to communicate securely, covertly, and possibly will pass on the skills to other people in the future. Be welcome. I wrote this book for you.

Regardless of why this book has sparked your interest, with a certain amount of care, you can achieve a security level in electronic communication unattainable on a computer managed by others and even less so on a system remotely administered by a software vendor. You can engage in secured electronic communication with enigmatic partners or obscure websites far off-limits at the workplace for good reasons. Furthermore, you can reliably protect the privacy of your electronic communication against criminals or even nation-state actors. You cannot achieve that at the workplace. It is commonly undesired there. Often enough, the employer is even actively preventing it.

## **1.4 The simplicity of achieving digital privacy in a true democracy**

For citizens living in a liberal democracy respecting their privacy, or for the reigning group in a dictatorship, life is simple. They do not need anonymity. They can use a potent End-to-End encryption tool openly and assertively.

All they need under this prerequisite is a fully controlled system, be it a computer or a mobile device, a software tool for potent encryption, and the ability to install and use this tool skillfully. Programmers developed the encryption tools prevalent today for this friendly web environment. Maintaining anonymity, given it was there in the first place, was off the radar when developing those tools. When used in a hostile environment, however, these tools break or endanger anonymity.

In our western democracies, for a long time, we believed it would be sufficient to strive for completion of the pyramid of free citizens shown in figure 1.1, i.e., work with Linux, encrypt emails using GnuPG, and chat via Jabber servers.

Regrettably, numerous voices portray this simple approach as complicated. Many citizens fall for this narrative, reject the use of encryption, and continue to use vendor controlled operating systems that may violate privacy. Even in such a simplified setting, most businesses and private computer users have neglected the pyramid's base with levity. Although free by their reckoning, they have succumbed to snooping by outside IT companies, by government agencies in the operating

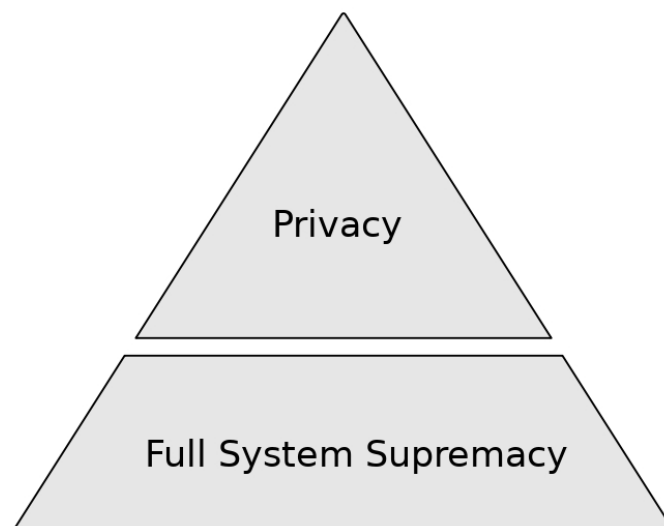


Figure 1.1: Pyramid of free citizens for secure electronic communication also called the pyramid of privacy, in short.

system manufacturer's home country, and by a significant number of unknown data traders. They use proprietary operating systems with undisclosed source code and disregard the base condition for secure electronic communication.

## 1.5 The situation we are in today

We know at least since the time of the Snowden revelations in June 2013 that, in the digital realm, we are in a transition from being free citizens to becoming tributary people. This transition is quite advanced already. Unfortunately, there is no tendency to decelerate or even roll back this decay of our digital civil rights. The process calls for countermeasures of us private citizens. We have to protect our privacy and the confidentiality of our messages against the snooping economy and the surveillance state with a more sophisticated pattern than that, shown in the free citizens' pyramid of electronic communication (figure 1.1). Only then will we, the ordinary citizens, experience more than just the illusion of communicating in privacy.

An incomplete strategy may predominantly rely on encrypting and later possibly add some elements of anonymity. But spokespeople of three-letter state snooping agencies cheekily tried to intimidate people who are using encryption. They publicly proclaimed they would put citizens' digital communication under extreme scrutiny should they dare to encrypt. Therefore we shouldn't do that according to the soup letter agencies.

It doesn't seem far fetched to assume, the agencies would satisfy their nosiness regarding the content of citizens' encrypted messages by pwning (hijacking by a hack) users' computers. Nation-state intelligence services can easily do that if the targeted individual or the used IP-address is known to the service. Nation-state services maintain collections of exploitable vulnerabilities for all widely used operating systems. Those of which yet unknown to the public are called Zero-Day Exploits. They can use these exploits to insert malware into their victim's computer and use such malware to upload data to or exfiltrate data from the victim's system. German government newly founded the service ZITiS to ensure it can pwn the computers of German citizens. A disliked citizen known to the authorities has little chance of protecting himself from such government attacks. Citizens' reliable anonymity, when surfing the Web, before being earmarked as dissident or as obstructionist, is the only viable protection.

Shortly after the Snowden revelations in 2013, many popular messaging services introduced



message encryption to reassure customers (but evaded anonymization). Some of those services were criticized in public by authorities because of using encryption and allegedly endangering the snooping services to go dark. But looking closely at the technical details of prevalent commercial chatting services disappointed me. One chat service (Signal) is almost sound. Most experts recommend using it. It has an open-source code, and it is built reproducibly, in part. Signal uses only standardized encryption algorithms and is subject to routine audits. U.S. government grants and donations fund it, and Signal is renowned for its uncomplicated user interaction. Yet this messaging service imperatively uses central servers, is incompatible with anonymous use, and may disclose metadata to third parties. The service pledges to keep user data confidential but is WhatsApp's partner and is subject to U.S. jurisdiction.

Nonetheless, those who don't feel threatened by U.S. agencies or by their governments and will never need anonymity or consider dealing with security issues regarding their messaging an annoyance may use such services and feel safe. At least no one will be harassed by immigration officers for using Signal when traveling to the U.S. But even these people should make an honest effort to master the techniques of the base layer of the pyramid of free citizens (see figure 1.1).

I wrote this guidebook for all the other people who want to stay in control of their data. I penned it for people who have a keen interest in the principles of anonymization and encryption. This book shall convey the comprehension of encryption, anonymization, and the eminently important combination of both. It should do so in a rather product independent way. Reading the book shall empower the reader to come to informed decisions on how to organize the security of his private electronic communication.

## 1.6 The price tag on digital independence and staying in control

At this point, let me stress that the pursuit of digital autonomy and informational self-determination comes at a price. For more autarky, we have to do without some of the conveniences of the usual division of labor and will have to engage in unbeloved activities. In the analog world, this is a truism already. Those who wish to live the autonomous life of a Robin Hood in Sherwood Forest may practice archery but must also stitch buttons on shirts, wash dirty underwear, dig latrines, and seal dripping roofs. Similarly, those who want to see themselves as digital rebels will need to find workarounds for irks and quirks in their system on their own. If need be, they must administer their computer via the console (see the corresponding glossary entry Console/Terminal on page 163) and must, to some extent, be able to fix causes for system errors. Those who permanently delegate such chores to other people of unknown loyalty lose control of their system and lose control of their data.

For my valuation, administering via the console is the digital equivalent to washing dirty underwear by hand in cold water in the Sherwood Forest. Be insured, I have spared no effort to minimize the share of such digital fiddling in this guidebook's exercise sections. A trustworthy, IT proficient friend who doesn't just take the work out of your hand but trains you as well, for example, for using the console can be quite helpful. But never overestimate your wizard friend's proficiency in cryptography. Solid knowledge in this field is thin on the ground. You will need to independently research the Internet, assess the quality of the sources you find, and implement the solutions you discovered for your system.

I will slowly back down from giving step by step instructions in the course of the exercises and will encourage the readers more and more to research on their own and to implement according to their findings. My goal is to empower readers. It is not drilling them. There might be readers who would prefer a list of recommended tools and apps, readers who want a shortcut to seal many data leaks with manageable effort. They might not be happy with my guidebook. I recommend to them satisfying their needs with the EFF's excellent guide for surveillance self-defense<sup>3</sup>. The

---

<sup>3</sup>see: <https://ssd.eff.org/>

guide is excellent, yet for my European taste, it relies a little too much on U.S. corporations, even when alternatives from the informal sector may be present. After all, the open-source property of a non-commercial encryption tool, which the user can easily compile and install from the downloaded source, is more reassuring than the open-source tag on the Android Signal client, installable from the binary image via the Google Playstore. With continental European eyes, we see U.S. entities with more suspicion than U.S. citizens do.

## 1.7 Communication between digitally submerged entities, in short: “Submarine communication”

In this guidebook, we want to achieve nothing less than ideally protected communication. We can visualize this kind of message exchange with the help of a maritime allegory. A powerful adversary (a nation-state secret service, for example, in short, the Data Kraken) surveils the surface of the ocean (the Internet). An unlabeled, dark submarine surfaces. (Your computer appears online anonymously in the Darknet.) The submarine releases a buoy with a message that has perfect encryption and submerges again into the abyss. Later another unlabeled dark submarine surfaces. (Your partner’s computer appears anonymously in the Darknet.) This submarine picks up the buoy carrying the message and submerges again into the dark abyss.

Given the powerful adversary finds the buoy in time, it can examine it and inspect the ciphertext (the encrypted form of a message). The adversary could then damage buoy and ciphertext or bluntly remove them. Removing them would establish evidence of the adversary’s surveillance. But the adversary cannot find out what kind of cipher the sender used to encrypt the intercepted data with or if the intercepted data are a ciphertext at all. The adversary cannot find out who sent the presumed ciphertext. It cannot get any information about the plaintext. The adversary has no information whatsoever about who has received or was supposed to receive the message. It only knows that possibly someone communicated or intended to communicate using its infrastructure.

I pursue the goal of empowering readers to conduct such perfectly protected message exchange with writing this guidebook. Going through the exercise sections, you will climb to the peak of secure communication attainable for intelligent laypeople. After that climb, you will probably wish to secure your day to day communication and will have to find your compromise between convenience and security, depending on your threat level. If need be under exceptional circumstances - like, for example, you are writing a privacy guidebook and might find evidence for interception of some of your emails - you know the way to the peak. You have been there already and know your way around.

I see four layers of proficiency on the way to ultimately protected digital message exchange, analogous to the maritime allegory given above. Each layer or level builds on all lower levels. Exercises on one level require the mastery of all lower levels.

Figure 1.2 depicts the levels in the shape of a pyramid. I often refer to this pyramid of secure electronic communication by using the term pyramid of dissident protection.

From now on, this guidebook will be all about practically completing the pyramid of dissident protection against the ubiquitous surveillance infrastructure.

## 1.8 Step by step training program for Digital Camouflage

We will systematically build up the pyramid of dissident protection shown in figure 1.2, level by level, from full system supremacy at the base via two flavors of anonymity up to privacy at the top. In contrast to conventional wisdom, I claim that dependable anonymity is a necessary prerequisite for robust privacy. Means for enforcing privacy, such as encryption tools, must not endanger or break underlying anonymity.

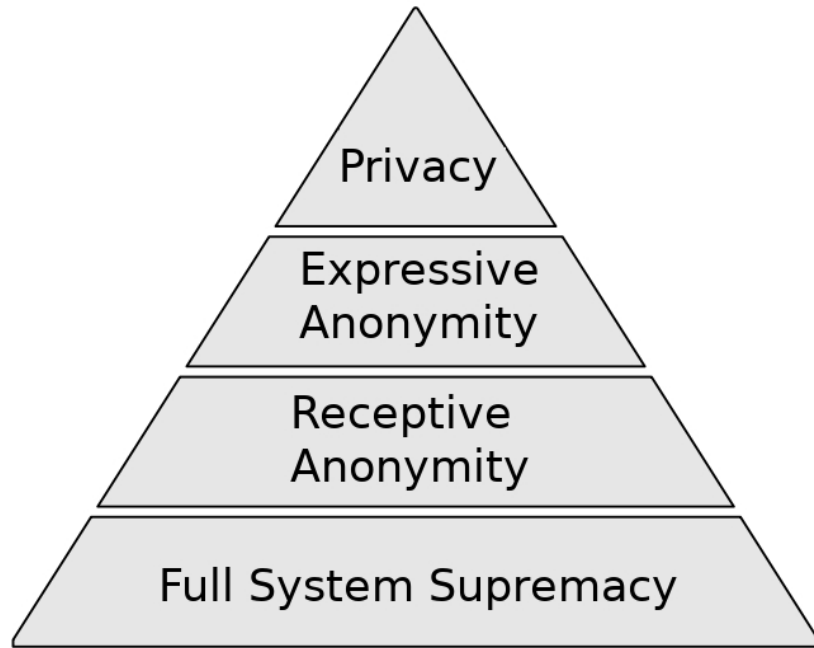


Figure 1.2: Pyramid of secure digital communication or pyramid of digital camouflage. In short, hereafter: pyramid of dissident protection. We break the path to perfectly protected digital communication down into the four milestones visualized as layers of the pyramid.

Unfortunately, most of the encryption tools available today and all of the tools, where encryption comes conveniently bundled with "easy" data transfer via pre-configured server connections, are incompatible with anonymity (messaging tools like Signal or WhatsApp, for example). Deplorably, leading security experts in the USA present such tools as modern and superior to combining flexible encryption tools, such as GnuPG, with freely selected means of transport. To some extent, even the organization EFF<sup>4</sup>, which is very helpful in general, voices this opinion. However, the "superior bundles" do not protect against your government, at least not against the U.S. government.


The ability to communicate securely can only be reached by a systematic learning process, which can't be completed simply by reading a guidebook. It would be as naive to expect this as to hope to learn to play the piano by reading a book. Extensive exercise work is essential.

In this guidebook, I treat each of the four levels of the pyramid of dissident protection by first giving the theoretical background. Then we look at some free open-source tools that allow us to achieve the goals set at that particular level. After that, I will present a selection of exercises that will give you the practical experience of doing things in the real world. The noble project of really getting things done instead of just babbling may appear cumbersome and strenuous, though. But anyone telling you, you could achieve secure communication effortlessly is lying. Anyhow, every single one of the four climbed levels of proficiency opens new opportunities for you to evade surveillance and resist digital oppression. Incidentally, reaching each of the levels also enhances your power of resistance against common Internet fraud. In the concluding section on each of the four levels of performance, I will give hints on the gravest attacks and the most painful vulnerabilities.

I will only describe measures for securing communication on the desktop systems Linux and Windows in this book. Many of the tools described here, however, can also be used on a macOS. I will not treat the mobile systems here, which we use on smartphones.

---

<sup>4</sup>see: <https://www.eff.org/>



## 2. Reign Supreme

### 2.1 Be in command of your operating system

A system is exclusively under user control if the following three criteria apply:

1. The user can check the source code or can have a freely chosen, trusted person do it for him.
2. Changing the system state needs explicit user consent.
3. The user can modify the system in any way desired.

#### Comments regarding condition 1

Proprietary closed-source operating systems such as Windows or macOS do not fulfill this condition. Because of its large private user base, I include descriptions for procedures and tools for Windows anyhow in this book. A user of the operating system Windows can practice all the techniques for what I call submarine communication presented in this guidebook. He or she<sup>1</sup> can explore the top three levels on the dissidents' pyramid. Windows users should be aware, though, that they can only reach all layers of the pyramid if they later return to the base of the pyramid and migrate to a Linux or another open-source operating system.

As long as the user cannot check the source code or have it looked at by a trusted expert of his choice, he depends on the benevolence and competence of the operating system's vendor. The vendor can openly or covertly do to the user's system whatever it wants to do. It has the power to do whatever authorities ask the vendor to do. The vendor could, for example, upload compromising data or unwanted surveillance malware to the system. It could leak sensitive data out of the system as well. It would be naive to assume that such access paths do not exist.

Be careful, though, even with open-source operating systems. Many chips built into our computers contain so-called firmware. Firmware is proprietary and usually closed-source software inaccessible to the computer user. The grave security error ROCA, which we discussed in detail in

---

<sup>1</sup>Please excuse me if I may occasionally use only the male form of address later in the book. I am well aware that many knowledgeable readers and many excellent hackers and developers are not male.

1.3 on page 10, is located in such a firmware of the chip TPM. The chip manufacturer, government agencies, or criminal hackers can maliciously manipulate the chip's firmware, even when the operating system is clean. Attacking the firmware, however, is very difficult without the consent and assistance of the manufacturer. But malware successfully planted into the firmware is very hard to detect for the defender. Attacker access through compromised firmware persists even if the operating system is nuked and installed anew. Thus, engaging in such attacks is highly enticing for powerful nation-state assailants.

### **Comments regarding condition 2**

Regular system updates are vital. They remove newly detected vulnerabilities and fix other software faults. For added convenience, many private computer users wish not to be troubled with these chores. (We have touched this point before in section 1.2.) These users want automatic updating in the background without their intervention. The providers of proprietary operating systems assume that all private users have this preference. They implement it this way and initiate such activities without asking for user consent, sometimes even without informing the computer user before such action. Critical users can only disapprove of such action, as it leads to their loss of control. Only when the user can trust the operating system provider's competence and loyalty for now and forever, this loss of control is more or less justified. On the other hand, in a customer/vendor relationship, the customer can never trust the vendor unconditionally for good and all. Customer lock-in is the least a customer of such naivety could expect.

Furthermore, Edward Snowden disclosed that all software vendors cooperated with the U.S. government's intelligence agencies. Thus, users of operating systems that allow unchecked remote servicing by U.S.-based operating system vendors suffer the loss of control and surrender to direct access by U.S. government agencies.

Each downloaded update packet must be explicitly cleared for execution and installation by the user in reasonably configured operating systems. At user request, they inform the user about the software to be updated, the reason for the update, and the changes introduced. Linux desktop systems in the standard configuration generally do that in an exemplary manner.

### **Comments regarding condition 3**

For any piece of software that would be executable on the system in principle, the user must be allowed to install and run it. It is not up to the operating system vendor to command what software may run on the system and what software may not. A popular proprietary operating system carries such unacceptable restrictions to the extreme and forces owners to use all sorts of dirty tricks to escape this so-called Walled Garden. Wittily these dirty tricks are called Jailbreak.

Similarly, the user must be allowed to delete unneeded or unwanted software with ease and complete. Software and hardware vendors tend to equip the systems they sell with all kinds of so-called crapware. They do this for marketing objectives or plainly against a fee. From the user perspective, crapware is an unwanted software, whose complete deletion is actively more or less aggravated by the vendor. A restricted version of Microsoft Office on freshly bought Windows computers, which becomes inoperable after some weeks of use, is one of the milder variants of such nuisances. In particular mobile systems tend to be delivered with an excessive load of crapware, which can only be removed after a Jailbreak or after rooting<sup>2</sup> the device. Even an issuer of a Linux distribution seems to have discovered such unacceptable means to generate revenue. In an Ubuntu (a widespread Linux distribution based on Debian), I had to repeatedly purge a so-called Music Store from my system that obtrusively showed up again and again.

Such provisions show the disrespect of software vendors towards their private customers, which also allows us to draw some conclusions regarding the intensity of the vendors' efforts to protect

---

<sup>2</sup>Rooting a mobile device means the owner hacks it to gain full control over it.

our data against government access or access by data mongers. Even if you may want to feel like a customer of the software vendor, be on the watch!

Users must be able to completely disable unwanted data exchange with the operating system's vendor. Unfortunately, since the introduction of Windows10, this seems to be no more warranted for the operating system Windows. If need be, you can come somewhat close to impeding phoning home by turning off the network interface or somewhat less rustic by changing device settings to the airplane mode.

## 2.2 Safe installation of software

The right to induce arbitrary changes to the system is not enough. The user must be able to initiate the changes, particularly the installation or update of installed software securely.

The files for new software or software updates nowadays typically arrive at your system via the Internet. During their travel through the web, they may be subject to reading access as well as manipulation by criminals, the marketing mafia, nation-state spying agencies, and other digital highwaymen. Malicious manipulation of such transmitted data is a way for attackers to gain covert or open access to their victim's systems. Among other administrations also the German government decided to engage in such digital highway robbery and founded the new agency ZITiS<sup>3</sup> for such and similar purposes on April 6th, 2017. The task to protect files against malicious manipulation in transit, which became necessary in recent times to protect against government misconduct, is not a new necessity. Many years ago, software vendors had to establish a technique to prevent criminals from infecting their update files destined for their remotely serviced customer systems. Private computer users who want to self determinedly install software on their systems nowadays need to adopt a similar safeguarding technique.

In the desktop computer, we already have cryptographic mechanisms in place, which allow for checking an update package's authenticity and integrity before its execution. They allow detecting any manipulation of transferred files en route to the recipient. Microsoft's Authenticode<sup>4</sup> is an example of such a vendor tool. These tools usually operate automatically and hidden from the user in the background. Such measures are supposed to prevent criminal web activities. But they also serve commercial vendor interests and governments. Unfortunately, they probably also serve illegitimate interests of nation-state secret services. Proprietary operating systems factually deprive the user of access to and control of such means of authentication. Thus, for self-determined usage of such techniques on a proprietary operating system, we need to set up new such means for employment under full and unfettered user control.

The non-proprietary Linux desktop systems, by default, come with user-controllable, open-source cryptographic means<sup>5</sup> to ensure the authenticity and integrity of web-transferred data.

Tools for cryptographic self-protection may not be present yet on your system, such as on a standard Windows installation, for example. Or such protection may not be supported by the distributor of free software you want to install. In this case, I recommend relying on HTTPS<sup>6</sup> and the commercial Public Key Infrastructure (PKI) to secure website access. All browsers support HTTPS. It provides a certain degree of protection. But be cautious, it gives security only against garden-variety Internet criminals. Competent nation-state agencies regularly bypass these protective means. In a trove of leaked CIA documents called Vault7<sup>7</sup> a CIA instructor warns the agents to never rely on the security of HTTPS when exfiltrating data from a victim's computer. Many nation-states and numerous companies would routinely sideline HTTPS-safeguarding, for example,

<sup>3</sup>[https://www.zitis.bund.de/DE/Home/home\\_node.html](https://www.zitis.bund.de/DE/Home/home_node.html)

<sup>4</sup>see: <https://docs.microsoft.com/en-us/windows-hardware/drivers/install/authenticode>

<sup>5</sup>GnuPG, <https://en.wikipedia.org/wiki/GnuPG>

<sup>6</sup>see: <https://en.wikipedia.org/wiki/Https>

<sup>7</sup>see: [https://en.wikipedia.org/wiki/Vault\\_7](https://en.wikipedia.org/wiki/Vault_7)

by using proxies as men in the middle<sup>8</sup>. For exfiltrating captured data, CIA agents should view such standard safeguarding mechanisms and encryption layers only as "blender layers". Own efficient encryption that would need to be "completely opaque"<sup>9</sup> would be mandatory. It should be employed additionally, anchored in a deeper layer of the outflow of data. I couldn't have expressed this any better myself.

### 2.3 Use of the digital signature

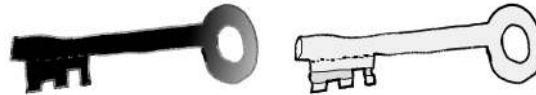


Figure 2.1: Private/public key pair

Left: Private key. Keep it secret. You need it to decipher ciphertexts and to sign documents digitally.

Right: Public key. Publish it in a way protected against tampering. Use it for targeted encryption and for verifying digital signatures issued by the holder of the private key.

Here I tightly sketch asymmetric cryptography and the digital signature. In contrast to common prejudice, the principles are simple and easy to understand. But, led by commercial interest, enterprises deform the simple pattern into a maze and bring that maze to the market. Those who want to get a firsthand impression should feel encouraged to read up on Microsoft's Authenticode procedures after completing this section.

In asymmetric cryptography, the participating computer user has a pair of keys (see figure 2.1). The pair consists of a private key and an associated public key. It is computationally easy to determine the public key from the private key. It is computationally near impossible, however, to calculate the private key from the public key or other data, which the owner of the key releases to the public. The owner must conceal the private key from the prying eye of an attacker and must distribute the corresponding public key. He makes it accessible to his communication partners in a tamper-proof way. A straightforward publication would be publishing it on an HTTPS-secured website, for example, where only the owner of the key has writing access.

Using the public key of the partner, you can encrypt a file for the partner. Only the partner can decrypt it using his private key. Using your own private key, you can create a valid digital signature of a file. This is impossible without access to the private key. Anyone who knows the corresponding public key can check the validity of the triple of the (signed) file, digital signature, and public key.

In case of a cryptographically self-protected software update or installation, the user has access to a copy of the public key of the issuer. Furthermore, the user has received the signed file and the digital signature via Internet downloads. If the signed file and its digital signature have been successfully checked against the public key, the user can be sure that the issuer of the software has signed the downloaded file in exactly the state, in which it is present on the user's system now. A single bit flip anywhere in the signed file would render the signature invalid. Manipulation by digital highwaymen, like, for example, conversion into a trojan on the way to the recipient, can be excluded then. After successful verification of the digital signature, the downloaded piece of software can be executed with confidence, if the signer is considered trustworthy.

After describing this simple manual procedure driven by the user, let me now comment on the Windows automatism Authenticode, that evades user involvement.

<sup>8</sup>see: [https://en.wikipedia.org/wiki/Man-in-the-middle\\_attack](https://en.wikipedia.org/wiki/Man-in-the-middle_attack)

<sup>9</sup>"completely opaque" seems to be the CIA slang expression for the well defined cryptographic term "negligible adversary advantage". Apparently, using proper scientific language is deemed inappropriate for agents of intelligence services and might intellectually overburden these people.

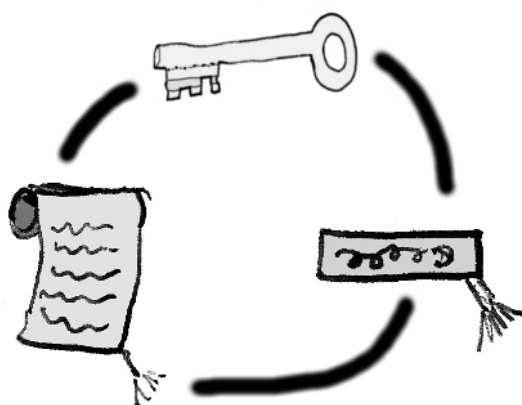


Figure 2.2: The triple consisting of the public key, file, and digital signature can be valid (coll. the signature is correct) or can be invalid. The signee sends the signed file and the digital signature to the recipient as a connected pair of files.

The verifier must retrieve the public key of the signee responsibly from an independently elected source at an independently selected and convenient time.

*In Microsoft's Authenticode procedure, a digital signature of the issuer is attached to the payload file, additionally an X509-like certificate. The latter contains a public key of the issuer, which had been signed by a Certification Authority. This signature of the public key by the authority had been verified at the beginning of this process automatically. On a positive outcome of this verification, the primary signature of the issuer will be checked against the downloaded payload file automatically using the previously validated issuer's public key. In a parallel query in an external database via the Internet the system checks, whether the certificate might have been retracted in the immediate past...I think you might get an impression of the maze caused by excluding the computer user receiving the update or installation file. And let's not forget a basic principle in IT: Complexity is the enemy of security.*

Please note that a hash value accompanying the installation or update file is barely security-relevant. It cannot substitute the issuer's digital signature. Any digital highwayman could calculate the new hash value of a manipulated payload file and replace the hash value of the correct payload file with the newly calculated one to evade this inherently broken attempt to secure file transfer with hash values. Unfortunately, some issuers of free software are still not aware of this commonplace and only offer hash values instead of a proper signature. In case you decide to install software from such issuers (this could happen in some of the following exercises), we will have to try securing such downloads employing a cumbersome and weaker replacement procedure to extract the most security possible from a given hash value.

## 2.4 Tools for reigning supreme on your computer

As with all other layers of the dissidents' pyramid (see figure 1.2 on page 16), we will introduce some open-source and free software to achieve the current layer's goal. Here we look at tools to come close to full system supremacy.

### 2.4.1 Open-source operating systems

The by far most widely distributed open-source computer operating system available for free is Linux. Thus I can restrict myself to only covering Linux here.

There are many free Linux distributions for private desktop computers or notebooks. Those distributions each include the Linux kernel and a collection of software modules needed for a usable



system, such as a mailer, browser, office packet, desktop organizer, and so forth. Furthermore, the distributions give access to a repository that contains a universe of all kinds of free software you can select for download and safe installation by a single mouse click. It is not hard to find the homepages of these Linux distributions on the Internet using your preferred search engine. One of the free distributions will surely suit your wishes for your software environment. For now or later, you should consider testing one or the other such distribution on your hardware. Once you have found your preferred one, you might want to migrate to your favorite Linux distribution, if you have still been working with a Windows or another proprietary closed source operating system.

I do not claim to be complete here with mentioning Linux distributions and will list some names only: Ubuntu, Kubuntu, Edubuntu, Lubuntu, Xubuntu, Debian, Mint, Knoppix, Trisquel, Elementary OS, Fedora, Mandriva, openSUSE, Raspbian, ...

You can find numerous links here<sup>10</sup>. I like Lubuntu<sup>11</sup> and Xubuntu<sup>12</sup>. These distributions' surfaces are similar to the old Windows version, on which I made my first attempts to walk in the digital realm many decades ago. If you also spent a lot of time with WindowsXP in the past and if you also loathe wiping on gay slats, you could like these distros too.

For hardware that is less than a few years old you can choose the 64 bit versions of the operating systems. Hardware that is around ten years or older might still need the 32 bit version. For the chosen Linux distribution you can prepare a Live-CD/DVD (or a Live-USB-Stick) and set the CD drive or USB port as the primary bootable device in your hardware settings. You can then use those to test if the distribution supports your hardware properly. And you can check if you like the respective look and feel of the distribution's desktop. You can do that without making any changes to your currently used system. The download pages of the respective distributions usually offer excellent step by step advice on how to prepare such live media. If you like the respective Linux distribution, you can start the installation with a mouse click from within the live system.

## 2.4.2 Dealing with digital signatures: GnuPG

Digital signatures compliant with the OpenPGP-standard<sup>13</sup> are by far the most widely used kind of freely and flexibly manageable digital signatures. We will use the well established free open-source software GnuPG (Gnu Privacy Guard) to handle such signatures. On Windows, you will have to install GnuPG. On Linux, GnuPG is usually included already in the distribution.

### Part 1, safe installation of GnuPG

Being a user of a Windows configured in a typical way, we encounter a dilemma at this point. Without the ability to use cryptographic self-protection yet, we have to install software for cryptographic self-protection in a secure way. I am reluctant to present you with the inelegant, convoluted description of makeshift security here. A digital signature, when implemented well, is simple, secure, and its use is transparent. The provisional procedure described here is neither simple nor is its application easy to understand, and safety is far from perfect. Nevertheless, I have devoted a lot of space and effort to describe the makeshift technique. Unfortunately, we need the process in practice. But I am reconciled that as you laboriously implement the makeshift procedure, you will expand your IT repertoire and practice its use. So let's get to it!

<sup>10</sup>[https://en.wikipedia.org/wiki/Linux\\_distribution](https://en.wikipedia.org/wiki/Linux_distribution) and [https://en.wikipedia.org/wiki/Lightweight\\_Linux\\_distribution](https://en.wikipedia.org/wiki/Lightweight_Linux_distribution)

<sup>11</sup><https://lubuntu.net/>

<sup>12</sup><https://xubuntu.org/getxubuntu/>

<sup>13</sup>The OpenPGP-standard is an encryption and digital signature standard maintained by the Internet Engineering Task Force (IETF) and can be found at the URL <https://www.openpgp.org/about/standard/> eingesehen werden.

**Dangers of unsafe software installation without self-protection**

An attacker could load the installation file with a virus or Trojan on its way through the Internet. During the subsequent installation, we would infect our system with it. Any station on the data link involved in the transfer could carry out such an attack. A resource-rich attacker could even break or bypass any HTTPS protection that might be in place for the data.

The attack could be like a shotgun blast aimed at any Internet user who downloads the installation file and poison every download with malware. Or it could be a targeted attack specifically on you and your computer. The threat of the shotgun attack may be bearable in your current situation. Many other Internet users can already use cryptographic self-protection. You cannot until you complete the installation of GnuPG. Checking the digital signature of the installation file (see section 2.3 on page 20), those other users would immediately detect the file manipulation. They will alert the publisher of the software then. That would expose the attack and elicit countermeasures that could compromise or expose the attacker. After a few days at the latest, the spook would be over.

More dangerous for you in this situation is the second variant, a targeted attack on you personally (or on a small group of targets) and thus on your computer accessing Internet resources under your IP address. The attacker inspects data packets passing by. In case these belong to installation files and are destined to your IP address, it poisons them with malware. The attacker could keep such an attack armed for months or even years without being generally noticed and fought.

You might object that an attacker can hardly know under which IP address you are on the net. But I assure you that there are many databases in which your IP address is assigned to your name and identity each day anew. Marketing businesses trade in such data. Every time you log in to a service, for example, when you log in to Google automatically, the service provider knows your identity from the account data and can link it to the IP address you use to log in<sup>14</sup>. If the marketing mafia can push personalized advertising to your system, a hostile organization can also force personalized attacks on you. The attacker could even buy all the data it needs on the gray advertising market.

We now want to make precisely this targeted attack on you more difficult by distributing your download activity to two IP addresses, the second of which will be newly assigned to you by your ISP. A targeted attack on you would then have to react very quickly to the new IP assignment to be successful. Merely up-to-date address data would no longer suffice for the attacker. It would need up-to-the-minute data. The portion of the download that occurs after the new IP address is assigned should be small and fast. Therefore, in the second part, you would not download the large installation file again, but only a digital fingerprint of this file, the so-called hash value (see also the corresponding entry in the glossary). This causes a much smaller "bug wave" in the data ocean of the Internet and is thus less clearly visible to monitors and attackers than a re-download of the large installation file.

At this point in the course, you now have the option to consciously tolerate the risk of an attack targeted at you, send a short push prayer to heaven, and without further ado, download the installation file and run it without checking. That would be the smoothest course of action and might even be reasonable given a moderate ambition and skill level on your side. In this case, however, please follow the next book sections carefully anyway, and practice the skills required in the process at least partially (hash calculation and handling your IP address), and skip only parts that are very difficult for you. The other option is to roll up your sleeves, gather concentration and tackle the admittedly complex, makeshift secured installation.

---

<sup>14</sup>Research "IP Targeting advertising" or visit this URL, for example: <https://www.dbswebsite.com/blog/ip-targeting-101-smart-display-advertising/>

### Preparing the system for makeshift protection

In the last section, I already mentioned the use of hash values. Perhaps you already have a tool for determining the hash values of files. Then the makeshift protected installation would be possible. You can think of a hash value as a digital fingerprint of a file. However, unlike a real fingerprint, the slightest change to the file, such as a single bit flip at any point, will change the hash value to a completely different number that bears no resemblance to the previous value.

Software for calculating hash values is probably already available on your system. If you are working with Linux, you can use the console command (see the glossary entry on page 163 and exercise 2.5.1 on page 30):

```
sha256sum <path_to_file> checksum.txt
```

The text file "checksum.txt" would then contain the calculated hash value. For other hash algorithms, you can change the command accordingly to `sha512sum`, `sha1sum`, or, e.g., `md5sum`. In the Windows command prompt, the corresponding console command would be:

```
certutil -hashfile <path_to_file> SHA256.
```

I recommend you directly install GnuPG as the first option mentioned in the last section if your system does not yet support the above console commands or you do not want to work with the console right now. Download the installation file of the software from its HTTPS-protected homepage and install it without securing yourself. With the HTTPS-protected GnuPG homepage, the risk of infection by malware is, in my estimation, lower than on a website for downloading any free Windows program for GUI-supported hash calculation.

If you still want to perform the makeshift procedure for self-security now but cannot calculate hash values yet, the search engine of your choice will surely provide you with a link to a GUI-based software suitable for this purpose. In this case, search for "calculate hash values on Windows", for example. The software must allow calculating at least the long outdated, already broken but popular hashes MD5 and SHA1 and the hashes SHA256 and SHA512, which are still considered secure. Furthermore, the publisher of the software should seem trustworthy to you. By the way, I think the naming convention of U.S. authorities coining the name "SHA" (= Secure Hash Algorithm) is somewhat pretentious. It has a similar value as the print "fresh eggs" on a packet of eggs in the supermarket - they were indeed fresh at some time in the past and might or might not be in such a state today.

Probably you do not have the TOR-Browser<sup>15</sup> installed yet on your Windows at this point. Thus, you better use all kinds of ad blockers, tracker blockers, and virus protectors from the panopticon available during your Internet search to keep the digital vermin of the Windows biotope at bay a bit.

### Downloading the GnuPG files

We will now proceed with the installation of GnuPG. If you have not already done so, please boot your computer now and open an Internet browser. You can find Werner Koch's GnuPG for Windows at the URL given in the footnote<sup>16</sup>. I find the request for payment or donation to be edited on the website a bit intrusive. Enter 0 Euro and click on "Donate & Download". If you will use GnuPG regularly later on and appreciate the software, you may return to this place and support Werner and his allies.

Those users who have opted for the unsecured option and can neither verify digital signatures nor calculate hash values at this point should take a deep breath, send a quick prayer to heaven and perform the installation unsecured.

Those other users who cannot verify digital signatures but can already compute hash values should continue the workaround and resort to hash values for security. For the GnuPG installation

<sup>15</sup>The TOR-Browser routes Internet access through the TOR Network, which currently is the most advanced anonymizing network. We will cover TOR in detail in section 3.3.3 on page 49.

<sup>16</sup><https://gpg4win.org/download.html>

file, also download the SHA256 hash value of the file. First, check with your self processed hash calculation whether the downloaded file truly yields the specified and downloaded hash value. If it does, the download was at least consistent. But a powerful attacker who monitors traffic and can also open HTTPS traffic, such as a government intelligence agency, could still maliciously replace both the download packet and the hash value with another matching pair that contains implanted malware.

### Workaround check of GnuPG files

You can do the following to make the previously described attacks aimed specifically at you more difficult. First, find out your current IP address that you used for the last downloads.

*Note: This IP address identifies you personally. Even if you do not have a static IP address but a frequently changing, dynamically assigned IP address, at least your Internet Service Provider (ISP) knows at what time your household used which of its IP addresses. The ISP might trade or share such information with authorities. If the agencies already have you in their sights, they may even have subscribed to the ISP's listings of the IP addresses assigned to your household.*

Grab your system's keyboard and get to work: your favorite search engine will surely respond with a rich list of links when you search for the keywords "find my IP address free". I just did that and found the URLs given in the footnotes<sup>17</sup> <sup>18</sup>, as two of the first three results. Go to such a web page and note the IP address that the page displays for your Internet access. Let some time pass, and then visit the GnuPG download page anew with a different IP address. You can read how to do this in the next paragraph.

Usually, you get a dynamically assigned IP address from your ISP. To get a different dynamically assigned IP address for the second time you access the website, you can turn off your home router, make a coffee, and take a coffee break. Sometime later, turn your router back on. Your ISP has now usually assigned a new address to your connection. You can check this by going to one of the websites you found earlier to get your new, now current IP address.

If you enter the Web with a fixed IP address, you may want to use a VPN to access gpg4win's website a second time. Careful and diligent users who want to get a little ahead of this course can, for example, get a list of free VPNs here<sup>19</sup> and use a VPN listed there. VPNs are not discussed in detail in this book, however, until later, in chapter 3 on page 39 on receptive anonymity.

With the new IP address for your account, you now quickly download the SHA256 hash value for the GnuPG installation file again without any detours. Check if it is identical to the previously downloaded value. In this case, if you had verified before that this value for the downloaded installation file is indeed correct, you can now run the installation file for GnuPG on your Windows system with tolerable risk.

If the control procedure described in the last paragraph fails, you should be alarmed. In the vast majority of cases, the failure is due to a random error in the data transmission. However, it could also be an indication of an ongoing attack on your computer. Never ignore such observations. In this case, cancel the installation and check your system for malware infection. Perform the same procedure again a few days later with increased vigilance.

Insist on a positive test! If the procedure fails repeatedly, do not install the software. That applies to any makeshift securing of a software installation. A repeatedly failed security test and subsequently aborted software installation would be less damaging than a system compromise. Often, an attacker might not be able to hack into your system directly, but it may well be able to disrupt your traffic and thus prevent proper software installation. People usually react to a malfunction by dropping security measures only to achieve the intended goal after all. Thus by

---

<sup>17</sup><https://whatismyipaddress.com/ip-lookup>

<sup>18</sup><https://www.whatismyip.com>

<sup>19</sup><https://www.vpngate.net/en/>

disrupting the function, an attacker could trick you into lowering security to the point that your system becomes vulnerable to its attack after all.

**Makeshift procedure for signatureless verification solely based on a given hash value:**

1. Download the installation file and claimed hash value from the download site.
2. Check the identity of the downloaded value with the self calculated hash value.
3. Wait for some time. Start a new session with a new IP address, possibly via a VPN.
4. Download the hash value again and check for the identity with the previously stored hash value.
5. If the hash values are identical, you may execute the installer.

**Concluding remarks and outlook for securing software installations**

If you, later on, download software issuers' public keys, you should also split downloads. Get them in one session and the corresponding installation and signature files in another one. Again split the task into two sessions, preferably using different IP addresses. In that case, doing it for the public key just once will suffice. You can then safely store it on your system, for example, in the repository of GnuPG or Academic Signature. There will be some practices and tutorials on using GnuPG in the exercise section.

Some readers might have just downloaded the GnuPG installer and executed it without an integrity check. I recommended this in this guidebook if there was no possibility yet to calculate hash values. Maybe you were used to doing it like that, and perhaps you indeed never noticed any unwanted manipulation of your system so far. But exactly now, using GnuPG, you got a powerful tool for manual cryptographic self-protection and independently secured software installations. Now you can abandon the hazardous habit of the past and discontinue running unauthenticated software installers.

Unwanted malware infection by running unauthenticated installers is rare. But over many months and years of using a private Windows system in risky ways, a covert or open malware infection might eventually happen. A study by Kasperski from 2013 showed that about 5% of all inspected Windows computers had picked up active malware, despite running antivirus protection software. About 13% of the systems without antivirus software harbored active malware. Kasperski screened for classical Windows viruses only, so I suspect that a substantial dark figure of further malware is present on the systems<sup>20</sup>.

*Please note: The vast majority of malware infections do not result from the targeted attack type described on page 23. The majority belong to two other categories. The first of them is to lure the victim to a maliciously programmed website that exploits bugs in a browser's script interpreter. The second one consists of tricking naive users into opening specially prepared emails in Outlook. Attackers may have peppered such emails with a maliciously crafted Office document as a mail attachment. Such Office documents usually contain macros exploiting MS Office bugs to take control of the victim's computer.*

*The sophisticated tailored attacks on the computers of selected targets mentioned earlier in this section do not occur frequently. But those who, like us, engage in encryption and anonymization to maintain control over our data can readily become the focus of government intelligence agencies. Carelessly checked or even unchecked installation files would be an invitation for attacks by such services.*

**Part 2, first steps to use GnuPG**

We have dealt with installing GnuPG in the subsection before. This installation had been necessary for users of Windows only. Now that we have installed GnuPG, we have to import or create keys in GnuPG's repository to use them. Both Windows and Linux users need to do this.

<sup>20</sup><https://eugene.kaspersky.com/2013/03/25/one-in-twenty-is-the-sad-truth/>

Now please go through a premature exercise. Create an OpenPGP public/private key pair linked to your email address using GnuPG! Using your preferred search engine, you will readily find helpful instructions for this task. Nowadays, a secure RSA key pair should have at least a length of 3072 bit. There are some graphical user interfaces for GnuPG to facilitate the RSA key pair creation and other user interactions. But many experts don't use the GUIs but prefer to run GnuPG directly via the text-oriented console instead.

In case you don't feel ready yet, to create the key pair on your own and according to your findings on the Internet, you may now prematurely work through introductory exercises RSx1\_I-1 and RSx1\_I-2 on page 30 and then generate the key according to RSx1\_I-5 by issuing the command `gpg2 --gen-key` in the console.

Later on, you will need my public key, the key of the author of this guidebook. In the next step, we will import this public key into the repository of your GnuPG implementation. That is what the key looks like:

```

---BEGIN PGP PUBLIC KEY BLOCK---
Version:  GnuPG v1.4.11 (GNU/Linux)

mQINBE8FgGEBEAClIAF1ZZbngJcJbLxCsD/CvBIsd032rdUoew/SI9S+4sp+Dl6uzWFER63c6ImwL
3Nr2lgw3Q9F7tZfPQcolvIFhjnvQ6Zp2yHe1ZyJkAZ8f4PjglwuMt2Ra979SDRSYPpEmWhvRkG+3o
rBFeqhUfNAGrZw/SdcgMNjbzsQqW2qresxY4wtEyDCY6ADBUMi8SCiW8EpWno i2IZP+KPwvVAOL3
Jv4BXOLImWjGwYtGoV10ByQv2HTmrKUbx7M9GZWENPsUBLLhRTFOhDt4wuXZrSmNhzu2tCqATEe0
sB7nyZz30u00s3uS6o9hE2ZEqtrVkdTAwazQ9szPkAFIQRxtFpsmum0lSP8bjHntDamht2bCe97lu
4PFEnghFibXnQioT5d4SVfR8e5av989mpmeKwLzm3wulrXpY6x+PjJ0hayfYC1Ph3+C7qK8jrx0Wu
PsLGH6HHyNoys+/3cnHBRMDxla8NxzbLW0yUmdHGE9R99wXzJEqNE3VeGhT2t41XQkKWHR4TjSoQ
/2F0Gu7ghgPIF6NDef42gh/n8J1uowu8AYEf v f qzJbCSy9wISWP4eGJYpSgFnvwJRvZk3foUvTXMW
FnhVfJ/X6BmYbEjQ/ZKH7ob5 0q49wV89wMjCaSQ6pFmJr3l6/bDhkuWMJ/KVpYMKrvGqMPOFQ/ZA
+/w9owARAQABtDJQcm9mLkRyLk1pY2hhZWwgQW5kZXJzIChrZXl f mJAxMikgPGFuQGZ0LXdlZGVsL
mRlP0kCPgQTAQIAKAUCtWAYQIbLWUJESwDAAYLCQgHAWIGFQgCCQoLBBYCAwECHgECF4AAcGkQcM
XXdBxtaFrp2w+/IIS6ihldFyUmCQaN63dV/8vh0PRL4n1HL+D1tq8QiW5kptL5pcrpYWwQvX+f4
H9xn03y/S0t2b/+VE6P1xYYxyiYc+s0lqUFnWqp0tdt4uwFHRUAh8uPQVURY7aVcZ3K8H5opR4A3H
GIVxXPArZbphPQNDL/OEsfGloAujCfBav/QeW4h2+LljaDslt/F3p+VDBPrwHtRBd4cCroZ326Y/X
NYfByopXWvQXVKtTL8J5vzt/UTqhbKYAlho4Z4Hu9MYmcX/VOQYS3s/ODCeQvhrJv/Fb0vA+1mA78
bNDA/fDG5ByQ8GhRkVMxv1M0yH9TWZnPDW7T+gpPyhfU+f qH3UQ4yr3ShmkSaLffvD9nlf cGHvNB
vI+U853v19s+YGaTKi9or6Y2YZ81AQ1rJ6Rh4MPHnD7qkQfCR+ciE54J7fNmWgEXOhED1MKYwt4dN
UPUzkV0K0l7hHxCL/Bt+taDOB6il98dAJ0dB7xlyi4+F2z+BmQE5wLwwkPeMYJiSy96uCZ65wpPjC
DUJIfnUEqc/dYzVrgJz6TBTZAiCI97OMDUj89/JlHWX0oPeTWDDvStOWNFSELUHXIb7vFnV+nPGs0
PhZANiKX7H1Pajb1PT5ypMhmsMJ0culJuTcl c583BPKT+oGsL7Khd5h50L1Q44yl+8vtLhUSVw3uy
KX1E= =pw2U
-----END PGP PUBLIC KEY BLOCK-----

```

Public keys in OpenPGP format, as the one shown above, are usually expressed in Radix-64 encoding. This encoding exclusively employs printable characters, such that we can easily represent them as prints on paper or transport them via text-oriented digital transmission channels. Importing the key printed above via typewriting would be unreasonable, of course. You can obtain the public key printed above in electronic form from the HTTPS secured homepage of Academic Signature<sup>21</sup>, ready for copy and paste. It is located down below on the web page referenced in the last footnote.

You may use your preferred search engine again to suggest a good website explaining how to import a public key into GnuPG's internal repository. Search for "import GnuPG public key", for example. Using a tool with a graphical user interface can facilitate this import. In case you don't feel ready yet to do this according to the result of your Internet inquiries, you may proceed as follows:

<sup>21</sup>see: [https://www.academic-signature.org/academic\\_signature\\_key.html](https://www.academic-signature.org/academic_signature_key.html)

1. Copy the key, as printed in the box above, from the presenting website and paste it into a new text file named "anders\_key.asc", for example.
2. Save this text file in a folder of your choice.
3. Open a console (see 2.5.1) and navigate to this folder in the console.
4. Type the following command in the console:

```
gpg2 --import anders_key.asc .
```

If you use a different name for the text file containing the Radix-64-represented key, of course, you use that name instead of the placeholder in the sample console command.

5. Watch out for gpg's reply and close the console again. If the response had not been an error message, the public key should now be in GnuPG's key repository.

An alternative way of doing this, albeit a less secure one, would be to employ a public key server. Your search engine fed with, for example, "public key server" will surely provide you amongst other hits the web page indicated in the footnote<sup>22</sup>. This web page presents and distributes published public keys. You can find my public key by searching the database linked to the website (along with some zombie keys I created in the past and lost access to). Alternatively, you may retrieve the public key via the console directly from a key server using the so-called "fingerprint" of the key as the reference. The console command for this would be

```
gpg2 --keyserver keys.gnupg.net --recv 0x1C6D685A .
```

The linkage of the key to the person claimed as the owner of the key is somewhat shaky, though, if retrieved from a public key server. You could generate a key, easily assign the name King Arthur, Josef Stalin, or Mother Teresa, for example, and upload it to public key servers. Experts may point to the so-called Web of Trust (WOT) to alleviate this identity problem and securely link the physical person to the key. The WOT is helpful in a benign environment where anonymous use of the web is unnecessary. I won't go into detail here regarding the WOT, however, because its use will publicly reveal who your friends and communication partners are. It would serve the knowledge of your social relations on a silver platter to an attacker and would permanently burn your chances to communicate anonymously in the future.

### 2.4.3 Dealing with digital signatures: Academic Signature

*Disclaimer: The author of this guidebook developed, maintains, and distributes the open-source tool Academic Signature free of charge. The author has no commercial interest in connection with this software.*

Academic Signature uses the most advanced of the well-established cryptosystems, the elliptic curve cryptosystem (ECC). The program is intended for manual use via a graphical user interface and allows producing so-called negligible-adversary-advantage ciphertexts. Such ciphertexts cannot be distinguished from random noise by an attacker, are fully compatible with anonymity, and thus can be ultimately secure. To my knowledge, there are no other free, open-source tools available, which allow producing such ciphertexts, employ public-key encryption, and which are fully compliant with anonymous use (and no commercial tools either, for that matter).

In the last section, we introduced the open-source tool GnuPG. GnuPG is currently by far the most widely used free tool for public-key encryption. However, GnuPG can be a little cumbersome for manual usage. Furthermore, it allows ECC encryption in the Linux version only. It creates ciphertexts that an attacker can readily identify as being GnuPG ciphertexts. Thus, using GnuPG, you will break or endanger anonymity. Hence, there is at least a need for another encryption tool to fill this gap.

Academic Signature's executable installation files for the Windows operating system and archives containing the source code are available on the web. From a so-called tarball, you can build Academic Signature from the source. Search, for example, for "download Academic Signature" in

---

<sup>22</sup><https://pgp.mit.edu/>

your favorite search engine (Duckduckgo, Startpage, Baidu, Yandex, bing, google,...) or find it directly here<sup>23</sup>.

Please retrieve the installer of your choice from the web page and also its corresponding GnuPG signature file. You may download the ECDSA signature also, which is to be checked later by Academic Signature itself. At this point, you should already have a copy of my public key from a trustworthy source on your system. Now you should use your favorite search engine to find instructions on how to check OpenPGP digital signatures using GnuPG, for example, with the search string "verify digital signature using GnuPG". A good search engine will probably present the following link with a high ranking<sup>24</sup>. If your current search engine buries this link under bought commercial hits, you might consider changing your favorite search engine, by the way.

Now follow the instructions given on the website, which you found most appropriate among the search results. Verify the triple consisting of either installation file or source code tarball, the corresponding OpenPGP signature file, and the public key of the program maintainer (in this case, mine). On a slow system, the verification might take a few seconds. If the triple is verified, you can install the software.

Installation of the binary on Windows should take no more than 2-3 seconds. If you are working with Linux and are compiling from the source code, you need some minutes to read the readme.txt file and install the necessary support packages from the distribution's repository. Furthermore, you will need some single-digit number of minutes, depending on the speed of your system, to compile and build the executable file. The final installation of the executable happens within the blink of an eye on Linux.

On the first execution of Academic Signature, you pass through an initializing sequence of events. At first, the software securely initializes the pseudo random number generator (PRNG). Secondly, it generates your first public/private key pair. Finally, it sets your access password and encrypts all data worth protecting using this password. The software's protected repository already includes my public ECC key. You can also call some procedures of GnuPG from Academic Signature's built-in GUI.

Now, I encourage you to perform a lagging double-check of the integrity and authenticity of Academic Signature's downloaded installation file. Please check against my public ECC key with Academic Signature. The check is easy since the software already includes my public key. Checking would be a useful warm-up exercise for using the tool. It is a dummy check, though, since the key came with the software and was not retrieved independently from the software download.

Many users perceive the guidance by the GUI as self-explaining. The others may resort to the video tutorials I give on Academic Signature's homepage. The check should come out as a positive verification since you did already check the GnuPG signature.

You have two independent means now for digitally signing files and for verifying digital signatures. Thus, from now on, you can manually use cryptographic self-protection independent of the opaque automatisms of an operating system vendor.

## 2.5 Exercises for reigning supreme on your computer

In the last two subsections, you already encountered some informal exercise sections while reading this book. Now we enter into the first formal exercise block. Please have a running computer next to this book in such exercise sections when studying the text. That will allow you to search the web and practice while reading the text. You will find a set of formal exercises below to work out the base of the dissidents' pyramid. Later, you will find similar exercise blocks for all other layers of the pyramid.

---

<sup>23</sup><https://www.academic-signature.org/>

<sup>24</sup><https://www.gnupg.org/gph/en/manual/x135.html>



All formal exercises in this book begin with a framed element printed in *italic*. This element states the level of difficulty, the estimated time required for the workout, prerequisites, and, if deemed necessary, special notes regarding the exercise. You may occasionally find interspersed text elements printed in *italic* again. Those passages are not part of the tight exercise flow. They give additional information, helpful for troubleshooting, or of general interest to pigeonhole your findings. You will note that I attach an identifier to the exercise titles. In the identifiers of this exercise block, RS stands for "Reign Supreme" and x for "exercise".

### 2.5.1 RSx1, introductory exercises

*Difficulty: medium.*

*Time: in total about 30-90 min.*

*Prerequisites: a running computer at your side.*

**I1)**

Create a folder on your system for use in these introductory exercises.

**I2)**

Here is the first console<sup>25</sup> exercise. It is a dull but necessary chore. Using the console, you can call operating system routines and execute any program installed on your computer without touching the mouse and without using the graphical user interface (GUI). Unfortunately, we have to memorize the console commands and type them into a console window. This is not as bad as it sounds, though, since their names are usually unchanged for decades, unlike the permanently modernized access patterns via the GUI. In principle, the GUI is merely a mediator to prevent us lay users from having to memorize and type commands and their parameters, but it internally just calls these commands.

Computer nerds love the console. It is customary, for example, to run the crypto-software GnuPG via console commands. After all, Werner Koch initially developed it for that mode of use. Most of us have experienced accidental damage to our computer's operating system at some time in the past. The mishap often left the system's inner workings still operational but only shot the GUI to pieces. In such cases, we can usually repair the damage or, at least, rescue our data using console commands.

In this exercise, you should search on the Internet for a site that gives a list of the most useful console commands for your system, for example, by using the search string "console commands PowerShell" or "console command prompt commands" or "Linux console commands". Many console commands are the same on Windows and Linux computers.

After you have found a site you like, start a console from the GUI. The console will start with the home folder as the currently active folder. To give you an impression, I will list some useful Linux console commands here on paper.

**ls** The command `ls` prints a listing of the current folder's contents.

**cd** The command `cd` (change directory) followed by a space and the relative path to the target folder changes to this folder.

**cd ..** Using the double period in the command `cd` changes into the parent folder.

**mkdir** The command `mkdir` (make directory) followed by a space and the name of a folder not yet existing creates this folder.

**rm** The command `rm` (remove) followed by a space and the name of a file or directory name deletes this folder or this file.

You will see many more such useful commands on the pages found with your Internet research. Now please navigate in the console to the folder created in I1). List the (not yet existing) contents of this folder. Then close this console again.

<sup>25</sup>(see the glossary entry on page 163)

**I3)**

Select a simple text editor you like for the next exercises. You will have to create and edit text files in ASCII format. Usually, you should find such editors on your system already.

**I4)**

Create a text file containing a handful of sentences using this simple editor. Store the text file in the folder you created for the exercises.

**I5)**

If you haven't done it so far, create private/public key pairs for OpenPGP/GnuPG using the cryptosystem RSA as well as for Academic Signature using the cryptosystem ECC. For GnuPG, you can do that by issuing the command: `gpg2 --gen-key`. For Academic Signature, you do that using the GUI of the program. In the following parts of the book, I assume the reader has installed Academic Signature and GnuPG and has worked through all the introductory exercises.

**2.5.2 RSx2, using Academic Signature's digital signatures**

*Difficulty: elementary.*

*Time: about 10 min.*

*Prerequisites: exercises RSx1\_I4, RSx1\_I5, and the installation of Academic Signature.*

*Note: The completion of this exercise is a prerequisite for many other exercises.*

1. Create a digital signature for the text file from the previous introductory exercise RSx1\_I4 with Academic Signature using your newly created private ECC key. If you need help, have a look at the instructions and tutorials on Academic Signature's homepage.
2. Subsequently, check the triple consisting of the text file, signature file, and your public key for consistency - colloquially: check the signature.
3. Flip a single bit of the text file. Do this, for example, by changing one uppercase letter into a lowercase letter or vice versa. Save the file and check the digital signature anew. Verification must fail now.
4. Revert the bit flip and restore the file to the original state, save it, and check the digital signature again.

**2.5.3 RSx3, using GnuPG's digital signatures**

*Difficulty: medium.*

*Time: about 20 min.*

*Prerequisites: all introductory exercises and the installation of GnuPG.*

*Note: The completion of this exercise is a prerequisite for many other exercises.*

1. Create a digital signature for the text file from the previous introductory exercise RSx I4 with GnuPG and your newly created private RSA key. You don't necessarily have to do it using the console. Surely you will find instructions and tutorials on how to achieve this on the Internet. Use your preferred search engine and select a good source for help. Subsequently, check the signature. I will not give explicit and detailed guidance here and encourage you to rely on your skill and practice searching the Internet. I would not be surprised, though, if your Internet search would refer you to the URL given in the footnote<sup>26</sup>, and if you would find the commands `--detach-sign` and `--verify` there for digitally signing and verifying signatures using GnuPG.

<sup>26</sup>see: <https://www.gnupg.org/documentation/manuals/gnupg/Operational-GPG-Commands.html>

2. Flip a single bit of your text file, for example, by changing one uppercase letter into a lowercase letter or vice versa, save the file and check the signature anew. Verification must fail now.
3. Revert the bit flip and restore the file to the original state, save it, and verify the validity of the signature again.

#### 2.5.4 RSx4, domain certificates

*Difficulty: elementary.*

*Time: about 15 min.*

*Prerequisites: none.*

Find a web page that you can access using HTTPS protection (Google.com or the Internet portal of your bank, for example). When accessing such a page, your browser calls some cryptographic background routines, which are already present in your system. These routines ensure that a bogus page cannot impersonate your home banking portal, trying to steal your login credentials.

Visit the website and inspect the website's certificate. A certificate in this context is a public key, amply garnished with more or less important additional information. This certificate and the site's domain name are linked. The domain is the web page typically ending in .org, .com, .gov, .de, or the like. During setting up an encrypted HTTPS communication, the website proves to the visitor that it has access to the private key belonging to the public key given in the website's certificate. That is its authentication.

If using Firefox, you would click on the small padlock symbol on the left side of the URL line to inspect the certificate. Then click into the field labeled "show connection details", click "more information" then and finally on "view certificate". It is a telltale sign that such important information is buried so much deeper than any buy-now button, for example. If you use a different browser, please search for this info on your own. Do not let up until you have at least found the public key in human-readable form and its bit length.

#### 2.5.5 RSx5, authenticate via the operating systems' automatisms

*Difficulty: elementary.*

*Time: about 15 min.*

*Prerequisites: none.*

*Note: The completion of this exercise is a prerequisite for many other exercises.*

Please install an additional useful free Software for your system, using the builtin automatic cryptographic safeguard means of the operating system during the installation procedure. I suggest installing the free chat client Pidgin in this exercise.

##### Variant for Windows users

1. Use a search engine of your choice to find the download website of the software. Visit the download page of the software using a browser of your choice.
2. Download the installation file and store it on your disk. Do not execute it yet. You may have to change the system settings to prevent automatic execution of downloaded installers and allow for a more finely grained control of installation processes.
3. Now start the execution of the installer. Jot down all system messages and inquiries during this installation, including your input. Think about which of those might be related to cryptographic safeguarding means. Gather all information you can get hold of about the cryptographic procedures (key length, used hash algorithms, used cryptosystem). Concoct explanations: What was the proper sense of each security-relevant inquiry or message?

**Variant for Linux users**

1. Select either the command line packet manager or a graphical packet manager for the installation. Use the manager to install Pidgin or whatever other software you picked for this exercise. Then start the download and the installation.
2. Jot down all system messages and inquiries during this installation, including your input. Think about which of those might be related to cryptographic safeguarding means. Gather all information you can get hold of about the cryptographic procedures (key length, used hash algorithms, used cryptosystem). Concoct explanations: What was the proper sense of each security-relevant inquiry or message? Is one of the messages unneeded, in your opinion? Is a specific message or inquiry lacking, in your opinion?

**2.5.6 RSx6, authenticate an installer using GnuPG**

*Difficulty: elementary.*

*Time: about 20 min.*

*Prerequisites: installation of GnuPG, all introductory exercises, and RSx3.*

*Note: The completion of this exercise is a prerequisite for many other exercises.*

At a later stage, you will need the OTR-plugin for encrypting chats in Pidgin. The Canadian Cypherpunks developed the OTR-plugin and are maintaining it. On a Windows system, you install it running an installation file downloadable from its website. On a Linux, the OTR-plugin is usually included in the repositories, and can be downloaded and securely installed using Linux's builtin automatic authentication by merely a mouse click. For this exercise, however, Linux users should manually download and manually authenticate, just like Windows users.

Using the primary source of a software package, signed by the issuer with the issuer's private key, is always a tighter and thus more secure interaction than employing the operating system's security mechanism. The latter would add an intermediary on the side of the operating system, which might be an unnecessary additional target for attack or bribery by malicious entities.

1. Browse to the Cypherpunks' official download web page and download the installer and its digital signature to your storage device. The developers of this crypto-software avoid the Windows automatism and thus allow for manual verification of the given GnuPG signature without having to rely on the Microsoft Corporation.
2. In a separate session, seek and find the public part of the key, which the developers used to sign the OTR-plugin's installation file. Import this public key into GnuPG's key repository.
3. Verify the OTR-plugin's signature against the installer and the developer's public key. On successful verification, run the installer.

*Note: On failure, repeat the procedure. On repeated failure, tell the developers about the authentication problem. Reputable developers are always keen to find out if users encounter such difficulties as they might indicate current malicious interference.*

**2.5.7 RSx7, authenticate an installer using Academic Signature**

*Difficulty: elementary.*

*Time: about 10 min.*

*Prerequisites: installation of Academic Signature.*

*Note: The completion of this exercise is a prerequisite for some other exercises.*

I assume you did already download and install Academic Signature (see 2.4.3 on page 28). I already encouraged you to verify the digital signature of the installer at the end of the installation instructions in subsection 2.4.3. If you did that already, you may skip this exercise. It could be a good idea, though, to repeat this task as part of this formal exercise block.

If you didn't download my ECDSA digital signature of Academic Signature's installer yet, please make up the backlog now.

- Please verify laggingly, using Academic Signature itself, that downloaded file, my digital signature of the latter, and my public ECDSA-key form a valid triple. Colloquially, we would then call this a valid digital signature.

Surely the validation of the digital signature will succeed. If it does not, some entity might have changed data on your system without your hand in the matter. That would be worrisome.

### 2.5.8 RSx8, install a free hex-editor with best security possible

*Difficulty: elementary.*

*Time: about 10 min.*

*Prerequisites: none.*

A hex-editor is a program to view and edit files as strings of raw hexadecimal numbers. Hexadecimal numbers are numbers expressed in base sixteen. Regrettably, the people maintaining and developing such programs often have limited awareness and expertise in matters of IT security. Thus you have hardly a chance to find a hex-editor installer from a developer's web page, secured by the developer's digital signature.

1. Please use the search engine of your choice to find links to such hex-editors for Windows. Despite their only being able to use such programs in a Windows emulator, Linux users should seek a Windows hex-editor as well. Linux users could readily find many good hex-editors from their respective repositories, secured by the systems automated security means. But that would not be the point of this exercise.
2. Select your favorite free and preferably open-source hex-editor for Windows. In the ideal case, the installer would come with a digital signature. At least it would use the Windows system's security automatisms. At the very least it would come with a checksum.

Make the most of the options accessible to you, to secure the install. When the security check does not indicate any fraud, Windows users should run the installer and install the hex-editor.

### 2.5.9 RSx9, encryption using Academic Signature and using GnuPG

*Difficulty: moderate.*

*Time: about 30 min.*

*Prerequisites: installation of Academic Signature and GnuPG, exercises RSx6, RSx7 (2.5.6 and 2.5.7).*

*Note: This exercise includes the task of sending a ciphertext via email to one of your accounts. If you have to live under a repressive regime, however, it could be advisable to omit mailing the ciphertext using your nonymous (= not anonymous) mailing accounts for now. In this case, only copy it to another folder on your system before deciphering it. If your government analyzes email traffic, you may want to avoid attracting unwanted attention.*

Pretend your authorities were respecting your constitutional rights, and pretend acting according to the pyramid of the free (fig. 1.1) would suffice.

1. Please select a document with innocuous content.
2. If you did not create private-public key pairs for GnuPG (RSA) and Academic Signature (ECC) yet, please do it now.
3. Encrypt the selected document asymmetrically (as a matter of fact using hybrid encryption) and create a GnuPG and an Academic Signature ciphertext of the innocuous document. You do that using your respective public keys.

4. If you can risk doing so in your country, send the GnuPG ciphertext as an email attachment from one of your email accounts to another email account controlled by you.
5. If you can risk doing so in your country, send the Academic Signature ciphertext as email attachment in the opposite direction between your email accounts.
6. Download the email attachments to a folder of your choice and decrypt the different ciphertexts using your private keys.

There are numerous good tutorials and instruction pages for these tasks on the Internet. Find them using your favorite search engine. Such sources might facilitate this exercise considerably. Spend some time investigating on your own to find the instructional web pages that suit you best. Act according to your findings.

Regarding Academic Signature, the tutorials on its homepage should suffice, and the labels in the GUI should adequately explain the operation. Regarding GnuPG, this might not be the case. But if you feed your preferred search engine with an appropriate search string, you will find GnuPG tutorials. I would not be surprised if your search engine would guide you to the explanatory web page given in the footnote<sup>27</sup>.

You might find the command for hybrid encryption there:

```
gpg --encrypt 04F028F8 test.txt
```

and for decryption:

```
gpg --decrypt --output test.txt test.txt.gpg .
```

In this case, 04F028F8 stands for the fingerprint of the key used by you. It has to be replaced by your key's fingerprint sequence, of course. "test.txt" stands for the file name of your innocuous plaintext file, and "test.txt.gpg" stands for the ciphertext file's name. In both command lines, you would have to replace the string "test.txt" by the name of the file chosen by you for this exercise. Before issuing the console commands mentioned before, you should have navigated in the console to the folder harboring the respective files. In GnuPG's decryption command, you need not give the key's fingerprint since a reference to it is included in the ciphertext file in the plain. Please note that this may break anonymity if used in an anonymous setting.

*Note: Some providers of free accounts discard emails with attachments of formats they cannot recognize. I found a few who even modified attachments. Yet well-encrypted ciphertexts cannot be analyzed or classified by external parties. If a provider blocked your email or damaged the attached ciphertext, you can wrap the ciphertext as a zip archive. Providers will usually transport emails with attached zip archives without difficulty.*

Congratulations, you completed the first exercise block.

## 2.6 Attacks on your reigning supreme

Right after the exercise section, I will present some thoughts about weaknesses and possibilities for an attack. I will do this for all other layers of the pyramid of dissident protection (see fig. 1.2) as well in the further sections of this book. Here it will be all about security threats and outright attacks on your reigning supreme on the computer.

I cannot claim, of course, that my thoughts penned here would be in any way complete. Please take into account that there is a universe of many more possibilities for attacks that I cannot discuss here.

### 2.6.1 Systemd (on Linux)

Rather than being an outright attack on a Linux, the problem described here can be seen as a hidden security risk.

---

<sup>27</sup>see: <https://www.gnupg.org/documentation/manuals/gnupg/>

Modularity, inherited from the predecessor Unix, is a fundamental principle of Linux. In comparison with operating systems of the monolithic type, like Windows, this is a distinctive feature. Recently, however, the maintainers of many Linux distributions have introduced a large system component called Systemd<sup>28</sup>. Introducing Systemd allows a more efficient boot process. While Systemd is still modular nominally, due to many internal dependencies, it is factually a monolithic block.

Traditional Linux developers highly value modularity. Nevertheless, many distributions forced them to support the monolithic block Systemd. This advancement of Systemd deprives the community of free Linux developers of access to an ever-growing part of this operating system. The maintenance of this monolithic block remains in the hands of a few. This promotes changes in the Linux support base that make Linux vulnerable to takeover by a central authority. Many Linux enthusiasts see such a concentration of power with deep suspicion.

There is criticism, Systemd's expansion would erode the modularity inherited from Unix by Feature Creep and Software Bloat, causing ever-growing code size and computing time. Linux would be pushed in an unfavorable direction towards an endpoint only too well known from the Windows madness.

In recent times the introduction of so-called snaps<sup>29</sup> and snapd was criticized for related reasons. The releasing company Canonical kept part of the code used for snaps closed source.

### 2.6.2 Trends among vendors and retailers

Depriving customers of control over their computers allows for numerous business models. They benefit different players in the value-added chain regarding computer sales. The most obvious benefit of those business models is generating revenue from ample license fees for software that is maintained, distributed, and hopefully improved for relatively little expense.

A more recent business model is selling exfiltrated user data supposed to improve advertisers' marksmanship. Vendors can easily nick user data from the systems of private customers unable to migrate to another operating system without unreasonable effort. Customer lock-in is a multifaceted jackpot for vendors.

Let me mention a further deplorable custom. Big retailers and computer vendors charge fees from IT-businesses for installing their crapware on customer systems. We already touched that point in 2.1 on page 18. Crapware is software unwanted by the customer, which is hard or impossible to remove from the customers' freshly bought computers. Businesses use this crapware as a marketing tool to stimulate customers to purchase further software or to purchase extended duration of a primarily limited software's usability period.

Such business models flourish, if vendors and retailers succeed in bedeviling their customers struggle to reach self-controlled computer use with free and open-source software.

Reports that vendors delivered freshly bought systems infected already with clearly identifiable malware, viruses, or trojans, did not come as a surprise. Presumably, this occurs without malicious intent. It may be due to sloppiness and neglect of customers' safety needs on the retailer side. I experienced that myself when I bought a new Windows10 notebook a couple of years ago. It arrived by mail from the retailer with malware installed in its hardware drivers out of the box. Alliances at the expense of the incapacitated customer are ubiquitous.

Nowadays, you can hardly find a local retailer in Germany offering computers without a preinstalled proprietary operating system. If you want a new computer with a clean hard disk, no obstacles in place to prevent installing a Linux on it, and without having to pay for an unnecessary Windows, you have to buy at specialized online retailers.

---

<sup>28</sup>see: <https://systemd.io/> and <https://en.wikipedia.org/wiki/Systemd>

<sup>29</sup>see: [https://en.wikipedia.org/wiki/Snap\\_\(package\\_manager\)](https://en.wikipedia.org/wiki/Snap_(package_manager))

Because of software vendors' and hardware manufacturers' benefit, interlocking between hardware and software components are brought forward at the expense of modularity and customer choice. The deterrent example for private users trying to avoid digital tutelage is the deplorable state of affairs already established for mobile systems. At the same time, this very example seems to be the vision of stakeholders in the computer-related value-added chain.

### 2.6.3 Microsoft reigns supreme on your Windows10

Microsoft's operating system Windows used to be relatively open in the past and has set few limits to the user. Throughout introducing new versions of this operating system, Windows has been locked down more and more towards the private user. Installing alternative operating systems alongside Windows, for example, is aggravated more and more.

Furthermore, private users can hardly prohibit steady messaging of the system to Microsoft's servers other than by a hard shutdown of the communication interfaces. Manually changing settings to limit the Windows garrulity often will be overridden completely after a system update. Incessantly calling home seems to be a constant nuisance. It is indeed also a security threat, even if the user trusts Microsoft because it would inadvertently cover up newly occurring malicious traffic after a hack.

Regrettably, migration to Linux should be suggested to privacy and security-aware private users nowadays.







### 3. Receptive Anonymity

We will focus on the second layer of the pyramid of dissident protection now. The overall structure will be the same as has been for the base layer. First, I will introduce the term receptive anonymity and have a glimpse at theory. Then, we will look at free and open-source tools useful for achieving the goal of this layer. What follows are hands-on exercises that will need your running computer next to the text. And finally, let's get aware of ways to attack and exploit vulnerabilities so we can better avoid them.

#### 3.1 What is receptive anonymity?

Introducing the terms receptive and expressive anonymity in this book is inspired by the differences in the types of data streams whose endpoints need protection. We use the term receptive anonymity to denote the ability for reading access to web resources, without revealing any information to the powerful net chaperone, that might link the data flow at hand to the person receiving the data.

We deal with techniques that might, for example, protect a teenager researching the web for information on contraception, abortion, or homosexual coming-out. If he or she does such research in a demure and oppressive society, the teenager might risk a stoning sentence. Of course, we also might have to protect citizens of a dictatorship with other interests. They may be interested in the kind of art, comedy, literature, or religious content, which was put on the index by the dictator's authorities. Disobeying citizens must fear being thrown into a torture prison when caught accessing such content.

In purely technical terms, even when just surfing the web, there is always outgoing data traffic in addition to the telltale inbound traffic. This might be, for example, handshaking traffic to establish an HTTPS connection. This technical traffic, however, does not contain individually shaped thoughts, opinions, formats, or language snippets. It will, by itself, not give any information allowing an authoritarian administration to zoom in on the individual in front of the screen.

A person sitting in front of the screen, sending individually shaped thoughts, opinions, formats, or political documents out into the world may also need protection. The kind of protection this person would need is called expressive anonymity in this guidebook. Expressive anonymity will be covered in the next section, and corresponds to the next higher stage in the pyramid of dissident

protection.

If we solely go for and only use the means for securing receptive anonymity, we cannot protect meaningful interaction via the web and have to avoid it. There is a reward for this restriction, however. Receptive anonymity is significantly harder to attack than expressive anonymity. The attacker cannot distinguish the individual's data flow from other data flows occurring under receptive anonymity by other individuals. The data flows do not bear any individual fingerprints. Thus, the attacker is unable to define a fictive victim identity and correctly assign other data flows to the same fictive victim identity. Therefore the attacker cannot extract and analyze behavioral patterns. The attacker cannot, for example, analyze statistics on activity times, uncovering information about the time zone the victim lives in. Similarly, the attacker cannot find other potential victims with correlated activity patterns that might hint to the victim being a member of a group and might point to other members of the group already. The attacker cannot use statistics to narrow down the group of real-world suspects in its physical sphere of control, to finally identify and arrest the victims.

Politically, I consider citizens free access to receptive anonymity as indispensable for the stability of a democracy. Influential members of the society (clergymen, politicians, military brass, scientists, unionists, captains of industry or money, etc.) should not become susceptible to blackmail by foreign or domestic intelligence services. They, like most people, might enjoy more or less condemnable guilty pleasures using read access to the Internet, often with a sexual connotation, and should not fall victim to snooping agencies.

In contrast to confidentiality, we cannot achieve anonymity on our own account. We need infrastructure backing anonymous network use and need a community of people who also seek anonymity. Members of that community have to adhere to specific rules. For example, they might have to use the anonymizing TOR-Browser properly. This way, they would inseparably mix their data traffic with the traffic of other members of the community. The overwhelming majority of this community should not need anonymity for protection, merely seek privacy, and increase the user base of the network for political and ethical reasons. Thus, we could dilute the pool of vulnerable users with many empathetic users just giving cover to the few vulnerable ones.

In the past, it was occasionally said (also by me) that only a tiny fraction of anonymity seekers wanted to anonymize politically, morally, culturally, liberally, or criminally motivated data traffic. The overwhelming majority of data traffic anonymized by TOR<sup>1</sup> would consist of ordinary pornography. However, new data (see: 4.5 on page 101) suggest that the vast majority is routed through TOR for protection against politically motivated surveillance by police and encroaching domestic authorities.

For example, the number of daily TOR users in the U.S. tripled from the long-term average of 300,000 to about 900,000 in the week following the murder of African-American George Floyd<sup>2</sup> by police officers and the start of large demonstrations of the Black Lives Matter movement<sup>3</sup> accompanied by surveillance measures by the authorities<sup>4</sup>. It was not until the beginning of the intensive Trump/Biden election campaign that the number of daily TOR users in the U.S. fell back to the previous base value.

Governments often justify attacks on net anonymity by the need to defend public chastity or to fight ostracized forms of pornography. However, political dissidents may be the real targets of such attacks. Let me speculate about the motivation for repressive governments and their intelligence services for gnawing at citizens' receptive anonymity. I believe they want to get a hand on material usable for blackmail and to better be able to pigeonhole citizens as friend or foe regarding their political opinion.

<sup>1</sup>We will look at the anonymizing network TOR in 3.3.3 on page 49 in this guidebook.

<sup>2</sup>[https://en.wikipedia.org/wiki/George\\_Floyd](https://en.wikipedia.org/wiki/George_Floyd)

<sup>3</sup>[https://en.wikipedia.org/wiki/Black\\_Lives\\_Matter](https://en.wikipedia.org/wiki/Black_Lives_Matter)

<sup>4</sup>see, e.g., <https://www.cnet.com/news/house-dems-ask-fbi-others-to-stop-spying-on-black-lives-matter-protesters/>

### 3.2 Geopolitical aspects of anonymous digital communication

Many naive computer users overestimate the power of attackers from the government's intelligence services. They presume services' almightiness and get locked into a fatalistic state of mind. Others see themselves as extraordinarily versed users and become consumed with the illusion, a nation-state attacker could not launch a specific attack on them, because they would have no chance to attack in this way on their own account. However, it is critical, and at the same time, it is hard to tune ourselves into the mindset of agents of a powerful intelligence service who can access mass data. Only then can we, the opponents of ubiquitous state surveillance, assess with some confidence which defensive means may be effective and which ones may not.

Those who can access, view and analyze the complete data traffic within a nation and may even be able to modulate the transmission rate of arbitrary data streams, can attack anonymity with much higher penetrating power than a small unit with a limited view on a small fraction of the data traffic only and access to maybe only a handful of routers on the Internet.

In the German language, we use the term "Datenkrake" (=Data Kraken) for corporations like Google, Alibaba, Facebook, Amazon, Microsoft, and for technically advanced nation-state intelligence services like the NSA or GCHQ, which are capable of such virtually unlimited real-time access. With such access to mass data of a nation or even a whole political domain, Five Eyes and allies, for example, theoretically each and any data stream, running within the governed domain, can first be marked by a modulation of the transmission rate and then be traced from the start point to the endpoint, and at last, be nonymized. It is unknown to the public if our western intelligence services are already capable of putting this theoretical possibility into practice.

As a consequence, you can only reliably achieve anonymity, if the data traffic you are an endpoint of exceeds the boundary of a domain, whose data streams are wholly at the disposal of a single attacker.

The selective collaboration of nation-state intelligence services belonging to different political domains is inevitable, might even be in our best interest, and is probably not a grave problem for our privacy and anonymity. Russian intelligence services and services from the U.S. sphere of influence will surely cooperate in the prosecution of one or the other terrorist, islamist, or drug trafficker. But it is unlikely that this could break the anonymization of domain crossing data traffic when the different political domains don't share mass data. I am quite confident that they will not share such mass data as supplying their mass data to another political domain would be tantamount to capitulation towards the competitor's intelligence agencies.

In my view, the intelligence agencies of the domains Five Eyes (US, UK, Aus, NZ, Can) plus close allies including Germany, Russia plus close allies, China and other Asian states with communistic heritage, Singapore, Indonesia and Malaysia, and US-critical Latin America would be suspicious enough towards each other not to share mass data of their respective territory. Western-oriented Asian states like Japan, South Korea, and Taiwan may comprise a further political domain, distinct from the Five Eyes. We may even have some single nation-states, setting value on their sovereignty, who may comprise own domains. France, Iran, Israel, Switzerland, or India, for example, might belong to this category and could be viewed as insulated mass data domains.

If critically dependent on anonymity in particularly sensitive data exchange, I would take care that the corresponding data stream would cross at least one additional political domain other than the Five Eyes sphere of influence. I would do that even if the communication partner lives as close as on the other side of the street. The path of the data within the second domain will hopefully be long enough then to inflict sufficient statistically distributed latency to the data packets in motion to obstruct correlation analysis based on a possible modulation of the data stream by a domestic spying agency.

Despite the international nature of the web, Internet cultures, privacy concerns, trust in the government, and web use seem to be quite different in different regions of the world. The by

far most employed anonymization network TOR limits the governments' power, but TOR needs volunteers to operate servers as TOR nodes. Thus, the supply of TOR nodes is quite unevenly distributed in the world's political domains. The local availability of technical expertise, private money, and the degree of suspicion towards authorities determine the supply of TOR nodes. We find a small number of nodes (almost none) in Asia. China blocks TOR use (users need special provisions to circumvent the blockade). TOR nodes are plentiful in Europe and North America, there is a reasonable number in Russia, and they are thin on the ground in South America.

Let me stress here that I sketched the political domains according to my gut feeling and draw conclusions from this mental domain assignment for my communication streams. You may have different experiences and assessments regarding the borders of political domains, which might be more reasonable than mine and map political circumstances better. Of course, you will base decisions regarding the anonymization of your critical communication streams on your mental map of political domains.

### 3.3 Achieving anonymity by mixing data flows inextricably

Most of us currently have a contract with an Internet Service Provider (ISP) and pay the ISP for giving us access to the Internet via an access point in our home, or via mobile Internet access. We then have a personalized business relationship. The ISP knows our name, our bank account, address, phone number, and other data on us. The ISP can ascribe each data packet we emit into the web, and each data packet we receive to our physical identity.

A well meaning competent ISP with a large customer base can safeguard our anonymity on the web if it would be willing to do so. Yet our ISP cannot generally be regarded as a discreet, trustworthy instance. Thus, we have to make sure our ISP cannot read the contents of our data packets and cannot get knowledge of the destination of outgoing information. Regarding inbound traffic, the ISP should not be able to learn about the source of the data packets or their content. Reticent straw men who act as intermediaries for our data exchange can protect us from a snoop ISP. Protection from a prying ISP is not enough, though. We also need to defend against powerful government agencies, having access to all data traffic in their territory. Computer users can do that by mixing their data traffic with the data traffic of other users inextricably at the reticent straw men mentioned above.

#### 3.3.1 Simple Internet access

At first, we will have a cursory view here on what happens when accessing a web page's content. Let us assume it is your favorite online newspaper's web page. In my case, this is the German news outlet called "die Zeit". As a regular reader, you know the URL of your preferred online newspaper, in my case <https://www.zeit.de/>. When we want to access the web page, we type the URL into the top line of our browser window. On our hitting the return key to send the request, our browser calls the Domain Name System (DNS). The DNS receives the human-readable URL, picks out the corresponding IP address of the hosting server, and returns this IP address to our browser. In case the server hosts multiple sites, the selected domain name will be transmitted to the hosting server in the pursuing Http traffic for an unambiguous request. At the time I wrote the first German edition of this guidebook (spring 2017), the IP address of [www.zeit.de](https://www.zeit.de/) had been 217.13.68.251.

When drafting the first English edition of this book, I repeated the request in April 2020. Now the DNS gave back 217.13.66.41, just as 2017 a server in Braunschweig (northern Germany) belonging to a company called Gaertner Datensysteme. Accidentally that same IP address was the last one configured as visible in spring 2017. So apparently, apart from a somewhat faster transmission, not much has changed in the meantime.

The server contacted by you must know whom to respond. Thus, your request necessarily

includes your IP address. In addition to that, according to the current settings of your browser, your request transmits a bunch of other additional data not directly necessary for the server response, for example, which website you visited before. In standard Internet communication, your browser is quite a blabbermouth.

In the so-called routing of your data packets through the Internet, your request, as well as the server's response, will be routed via flexibly selected intermediate stations called routers. If you transmit the data in plaintext (unencrypted), each of the participating routers could read the contents of your packages. It could read out your and your target server's IP addresses, or even selfishly manipulate your data in transmission.

Each computer connected to the web, like a router, for example, has an IP address assigned. Other connected systems address it by this IP address. Nowadays, in times of Industry 4.0 and the age of buzzwords, even each flowerpot seems to need an IP address to be a good flowerpot. You can use the command "tracert" ("tracert" in a Windows console), to compile a list of all intermediate routers your data pass on the way to your target host given their administrators configured them to be visible. There may be good reasons for stations to remain invisible. A covert station is not necessarily an exfiltration and surveillance machine.

Let's now have a look at the Internet route from my notebook to the server of the online newspaper. I opened a console on my Linux computer (in spring 2017) and typed the command `tracert zeit.de`. Below you can see the traceroute log on the path from my notebook to the newspaper's web server. I replaced some IP addresses close to my end of the data path by the string `xxxxx...` to protect my privacy.

```
tracert to zeit.de (217.13.68.251), 30 hops max, 60 byte packets
1  gateway (xxxxxxx) 0.238 ms 0.568 ms 0.553 ms
2  xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx 31.696 ms 32.962 ms 35.155 ms
3  62.214.61.201 (62.214.61.201) 36.903 ms 38.626 ms 40.223 ms
4  62.214.37.114 (62.214.37.114) 67.778 ms 67.847 ms 67.828 ms
5  qsc.bcix.de (193.178.185.60) 54.672 ms 74.653 ms 74.668 ms
6  scmber11-et-1.qsc.de (87.234.14.172) 67.612 ms 43.897 ms 44.027 ms
7  scmham11-cg-0-0-1-0.qsc.de (87.234.14.138) 45.833 ms 47.937 ms 49.631 ms
8  scmham12-cg-0-0-1-1.qsc.de (87.234.14.136) 51.623 ms 53.531 ms 55.805 ms
9  scmdus12-cg-0-0-1-0.qsc.de (87.234.14.134) 57.473 ms 59.141 ms 61.065 ms
10 crmdus11-cg-10-0-2-0.qsc.de (87.234.14.167) 63.064 ms 65.021 ms 66.924 ms
11 suhlu-v9.gaertner.de (217.13.66.41) 70.434 ms 71.774 ms 73.946 ms
* * *....
```

At station 11, the command's processing reached the newspaper's server. The target server's administrators consider further information on the internals of the subsequent processing of my request to be none of my business and withhold it. We can find the probable locations of the servers logged by the traceroute command using one of the many popular IP geolocation websites. In this case, I used the page given in the footnote<sup>5</sup>:

```
62.214.61.201 (Versatel Server Duesseldorf)
62.214.37.114 (Versatel Server Duesseldorf)
193.178.185.60 (bcix eV Berlin )
87.234.14.172 (qsc ag,Cologne)
```

<sup>5</sup><https://www.neustar.biz/resources/tools/ip-geolocation-lookup-tool>

```

87.234.14.138 (qsc ag,Cologne)
87.234.14.136 (qsc ag,Cologne)
87.234.14.134 (qsc ag,Cologne)
87.234.14.167 (qsc ag,Cologne)
217.13.66.41 (gaertner datensysteme gmbh & co. kg, Braunschweig)

```

I apologize for not having an explanation for the qsc ag's server ping pong in its routing within Cologne. The mnemonics in the server names situated in Cologne, like "scmber11-et-1.qsc.de", also with ham and dus in addition to ber, seem to point to names of larger German towns. But I should better refrain from further speculation on that. The latency<sup>6</sup> of a higher two-digit number of milliseconds, somewhat less than a tenth of a second, seemed normal for the data traffic of private Internet users in 2017. In fall 2020, three years later, the typical latency has about halved.

Let's put it on the record. In this case at least "Versatel", the "bcix eV", the "qsc ag" and the "gaertner gmbh" can log, which article in "die Zeit" the holder of my IP address (me!) looked at, when he did it, and for how long he did it. Each of these entities could have additionally manipulated the contents of the articles transmitted to my computer, holding this IP address.

Now assume that a government service or some lawyer would have access to the logged data traffic of my newspaper's server. They could ask my ISP to reveal which individual was holding this IP address at the time of the respective traffic to find out which articles I read and for how long. The lawyer might need to pretend to have a legal title, though.

*Note: At the time of writing the first German edition of this guidebook, accessing "www.zeit.de" had not been protected by HTTPS yet. "www.zeit.de" has introduced HTTPS protection for its data traffic by now. Thus, it substantially aggravated unwanted eavesdropping or manipulation of the data packets in transit. Only powerful entities able to open HTTPS traffic, government intelligence agencies, for example, could still do that.*

Central Internet nodes, like the Berlin bcix participating in the traffic at hand, are a strong temptation for government agencies. They seem to feel a hoggish urge to wiretap all the traffic passing through such nodes. A node similar to bcix, the much larger De-Cix in Frankfurt am Main, has gained notoriety as a large exfiltration node for the German intelligence service BND. You may search using your preferred search engine for "De-Cix BND surveillance" or look into one of the related online articles here<sup>7 8</sup>.

I don't want an unregulated community of eavesdroppers to learn about my political opinion by snooping around in my Internet traffic while I'm reading an online newspaper. Mind that there are much more sensitive Internet activities than reading an article in a respectable newspaper. It wouldn't come as a surprise to me, dear reader, if you assess the situation similarly.

### 3.3.2 The VPN-Server as discreet intermediary

#### The Virtual Private Network (VPN)

Companies frequently use Virtual Private Networks (VPNs) to enable employees at remote locations anywhere in the world to log into and access the company network. The VPN establishes a secure encrypted "tunnel" for data packets transferred over the Internet between an employee's computer and the company network. Digital highwaymen in a hostile environment which the data packages must pass through cannot read or manipulate the exchanged data. As used in this book, however,

<sup>6</sup>Latency in Internet communication is the time delay between a request and the response.

<sup>7</sup><https://netzpolitik.org/2015/how-the-german-foreign-intelligence-agency-bnd-tapped-the-internet-exchange-point-de-cix-in-frankfurt-since-2009>

<sup>8</sup><https://www.datacenterdynamics.com/en/news/de-cix-files-constitutional-complaint-over-state-surveillance-german-data-center/>

the Virtual Private Network will not be run as a network. We will use a truncated version reduced to the tunnel and one VPN server. This server acts as our straw man, only decrypting and forwarding our Internet requests in the plain, encrypting the incoming responses for our computer, and passing them on to us.

Of course, the VPN server knows our IP address. It is transmitted with and also in our encrypted requests. The VPN server, however, replaces our IP address in the requests by its IP address in the decoded packets and then forwards them in plaintext to the receiving website. In case we requested an HTTPS protected exchange, the payload of the data packets ideally is inaccessible to the VPN server because we additionally encrypted it in a way negotiated between our system and the receiving website.

Please note that the VPN server's administrator is in a prime position to try a man-in-the-middle attack (MITM) on our HTTPS data traffic. As MITM, he could eavesdrop on our HTTPS protected data traffic or even manipulate it. Thus we have to trust the VPN to some degree. The VPN server's administrator could try to open our HTTPS protected exchange, redirect our requests to maliciously prepared websites, or grass us up by telling the government which web pages we visited.

We will now employ a free VPN server to avoid revealing our IP address to the webmaster of the website we intend to visit. At the same time, we avoid that an eavesdropper monitoring our traffic nearby learns the visited website. As mentioned before, this double protection begins with contacting the VPN server and negotiating an encrypted connection, the tunnel. Then we can send our encrypted Internet request to the VPN server. The VPN server must know to which computer it should respond. So our query already contains our IP address in the metadata. In the payload, our data packets contain the URL of the site we want to visit. However, since the data packets are encrypted, an eavesdropper intercepting our data exchange with the VPN server can only learn who is contacting the VPN. But it cannot find out what website we are addressing. The VPN server decrypts our packets, reads the target website's URL, requests the data for the VPN server's IP address, receives the response, encrypts it for us, and passes it to our computer. An attacker who intercepts the Internet traffic between the VPN server and the target website can discover the final destination of the VPN server's data packets. But the eavesdropper cannot learn who requested the data initially. The data stream between the VPN server and the target website does not contain our IP address. Thus the target website cannot learn either that our computer (our IP address) is the final destination of the response.

The VPN server's log files link our request for the target website to our IP address. Thus we need a reticent and trustworthy VPN server administrator for protecting the secrecy of our IP address. We mix our traffic with other users' traffic at the VPN server, and only the VPN server has firsthand information on our target websites. But watch out, if few users utilize the VPN, a powerful adversary, surveilling the VPN server's web environment and eavesdropping on the VPN traffic, could easily blow our cover. The adversary could match up the VPN's inbound encrypted traffic originating from our IP address with outbound Internet requests by the VPN server. Thus, a highly frequented server is better for our anonymization than a rarely used one.

Besides, we should take care not to get our browser permanently tagged with cookies and keep the plethora of other abusive tracking crap at bay.

To sum it all up, when using a VPN, we are barely protected against government surveillance. But we have reliable protection against unwanted data collection by the visited websites or our ISP.

### Using a VPN

Now I will access my favorite online newspaper ([www.zeit.de](http://www.zeit.de)) via a VPN to make it difficult for government agencies to access knowledge about my political orientation and prevent my ISP, the website's, or integrated advertisement servers' administrators from learning anything about my interests and reading habits.



From a website<sup>9</sup> listing public, free, and informal amateur VPN servers, I pick a currently available VPN server in Vietnam (116.102.xxx.xxx), run all my traffic over this server for this session, and access the newspaper's web page via Vietnam. To protect the privacy of my buddy in Nam, I blanked the last fields in his/her IP address. The responses are a bit sluggish now, but I can still comfortably read my online newspaper. The web page doesn't object to access from a Vietnamese IP address.

I am interested in the routing stations between my system, the server in Vietnam, and the newspaper server in Germany and try the command "traceroute zeit.de". Oops, this time it failed. I receive no substantial information. But think about it, how could it come out differently? Firstly I don't get responses to my request about the route from my computer to Vietnam. The traffic's payload is encrypted, and how should the involved relay stations know I issued a traceroute command in the first place? But I don't get any sensible response about the route from Vietnam to the German website either. The reason is that the routers in this part of the data path don't know about me. They would respond to the Vietnamese VPN server, who wouldn't forward such information for its legitimate protection against unknown me. But undoubtedly, services having access to my web environment can detect my data packets' path from my computer to the Vietnamese server. It just cannot be shown to me if I issue the traceroute command. I perform an easy little maneuver. I deactivate the VPN connection and address the Vietnamese server directly and openly with "traceroute 116.102.xxx.xxx", the xxx replaced by the real IP address fields, of course. That will map out the path to the VPN server then.

```
traceroute to 116.102.xxx.xxx (116.102.xxx.xxx), 30 hops max, 60 byte packets
 1 gateway (xxxxxxxxxxxxxx) 0.308 ms 0.277 ms 0.646 ms
 2 xxxxxxxxxxxxxxxxxxxxxx 32.121 ms 33.674 ms 35.366 ms
 3 62.214.61.201 (62.214.61.201) 37.314 ms 39.322 ms 41.486 ms
 4 62.214.37.126 (62.214.37.126) 55.506 ms 55.893 ms 55.904 ms
 5 bei-b1-link.telial.net (213.155.129.190) 54.049 ms 56.238 ms 57.548 ms
 6 ffm-bb4-link.se.telial.net (62.115.133.8) 78.794 ms ffm-bb4-link.telial.net (62.115.112.157)
 54.668 ms ffm-bb4-link.se.telial.net (62.115.133.12) 56.767 ms
 7 mei-b1-link.telial.net (62.115.112.231) 75.842 ms 77.600 ms mei-b2-link.telial.net
 (62.115.133.181) 77.827 ms
 8 snge-b1-link.telial.net (62.115.114.37) 233.607 ms snge-b1-link.telial.net (62.115.122.245)
 234.749 ms snge-b1-link.telial.net (80.91.247.55) 265.033 ms
 9 viettel-ic-314019-snge-b1.c.telial.net (62.115.60.18) 276.640 ms 279.745 ms 258.692 ms
10 localhost (27.68.248.33) 338.684 ms 354.614 ms 354.623 ms
11 localhost (27.68.255.49) 375.056 ms localhost (27.68.255.85) 352.175 ms localhost
 (27.68.255.49) 337.342 ms
12 localhost (27.68.208.142) 367.102 ms 372.429 ms 373.022 ms
13 125.235.xxx.xxx.adsl.viettel.vn (125.235.xxx.xxx) 347.320 ms 340.164 ms 341.912 ms
14 * * *
15 * * *
```

The routers don't grant me any routing information beyond station 13. Now let me localize the routers as in the case of the unprotected web page access.

```
62.214.61.201 (Versatel Duesseldorf)
62.214.37.126 (Versatel Duesseldorf) 213.155.129.190 (telia company ab, Stockholm)
62.115.133.8 (telia company ab, Frankfurt)
62.115.112.231 (telia company ab, Vienna)
62.115.114.37 (telia company ab, Stockholm)
62.115.60.18 (telia company ab, Madrid)
27.68.248.33 (viettel corporation, HaNoi)
27.68.255.49 (viettel corporation, HaNoi)
125.235.xxx.xxx (viettel corporation, DaNang)
```

So I can conclude, that presumably the VPN server, which I used for reading the newspaper, is located in Da Nang. After a ping pong in Europe, the web sent my packets from Madrid directly

<sup>9</sup>see, e.g., <https://www.vpngate.net/en/>

to Vietnam and then from Hanoi to Da Nang. The server in Da Nang requested the web pages of "Zeit.de", encrypted the answers for me, and forwarded them to my IP address in Germany. My computer deciphered the arriving packets and showed the result to me in my browser. I like to express special thanks to my unknown buddy in Vietnam who helped protect my privacy here in Germany.

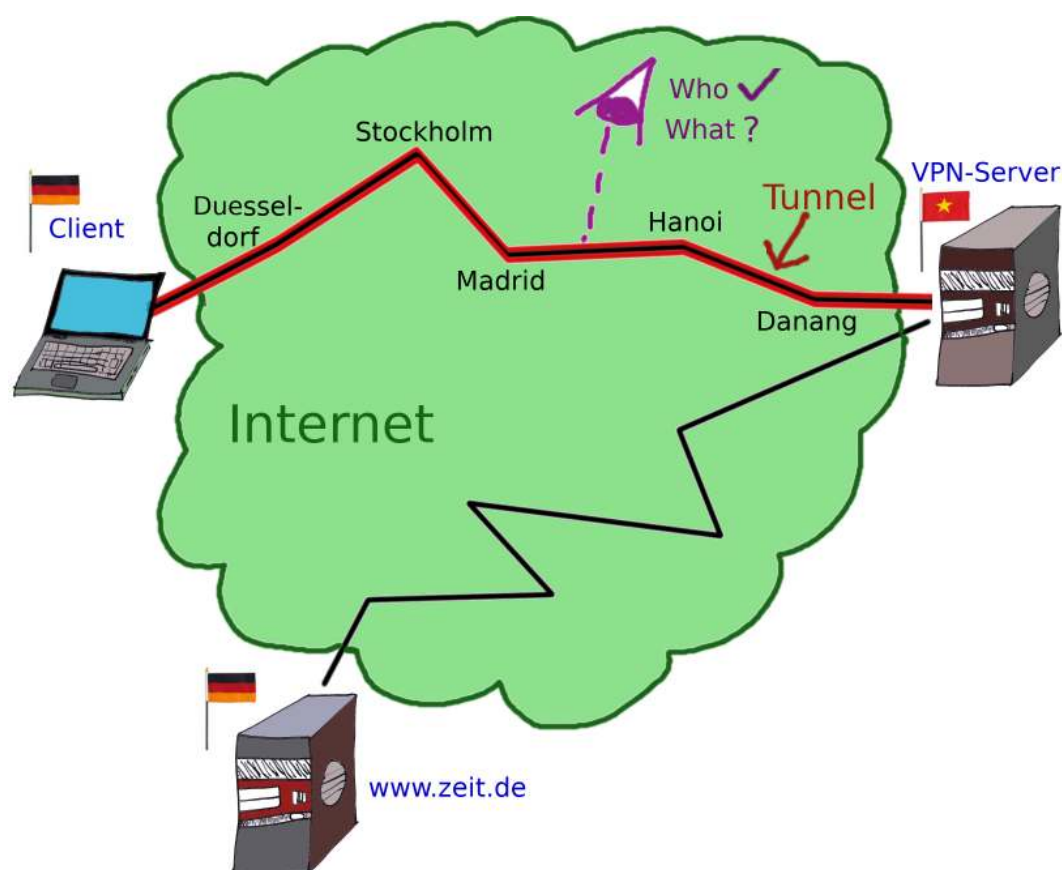


Figure 3.1: The figure illustrates a request for an article in an online newspaper via a Vietnamese VPN server. The illustration omits some routers for clarity. A snooper looking at the client's traffic to the VPN server in Da Nang can learn, **who** communicates with Da Nang, but cannot find out **what** the content of the communication is. An eavesdropper listening in on the traffic between Da Nang and the newspaper's website cannot learn anything about the client, the real cause of the traffic. But he can learn about the endpoints of this data stream (VPN server in Da Nang, Vietnam, and newspaper's website in Braunschweig, Germany). If the snooper is powerful enough to open standard HTTPS, he can learn about the communication's content, as well.

In this case, my service in return for the free use of the VPN service of my kind buddy in Nam was to give some "digital cover" or "dilution data" for Vietnamese data traffic to western Europe.

### Pitfalls

Let me sound a note of caution.

We should take it as a given that western government agencies exfiltrate any data traffic between Germany and Vietnam at a muted station (muted concerning the traceroute command). It will subsequently be evaluated and archived. The traceroute result shows a relatively large increase in latency after passing the second last European stations at telia (Stockholm-62.115.133.181, **77 ms** and Madrid-62.115.114.37, **234 ms**). This could indicate some "analytic activity" by our curious western intelligence services. Surveilling this Internet traffic is cheap and easy since there is a rather

limited amount of data traffic between Germany and Vietnam. Whatever kind of data collection is cheap, easy, and opportune will be done by our government's snooping agencies. We should assume that my communication with the Vietnamese VPN server and the closely correlated requests from Da Nang to the newspaper Zeit.de would stick out from the usual YouTube kitten video junk traffic like a checkered elephant in a shopping mall. The soup letter agencies would have no problem uncovering my newspaper reading session via the Vietnamese VPN server. The protection against a nosy ISP or an all too curious newspaper webmaster should have been operative, though.

Let me sound a note of outrage.

When writing the English edition of this book about three years after penning the first German edition, I repeated the analysis of the routing of data exchange with a Vietnamese VPN server. I was curious if there would be a European router again, at which the data traffic would suffer a substantial increase in latency, hinting at some "analysis" by snooping agencies. The result was distressing. In spring 2020, there was no longer a European ping pong. Instead, the web infrastructure routed the traffic destined to Vietnam through the UK first. In the UK, the data traffic suffered a telltale increase in latency. The UK then routed the data traffic to New York, New York routed back to the old world to Istanbul, Turkey, and from there, the traffic went to Asia and finally into Vietnam. I can imagine only one reason to route this traffic westwards first, through the UK and the U.S. east coast, and then finally back east into Turkey and only then finally towards Asia. I'll refrain from further commentary on this tell-all forward about Darth Vader chewing on dates in London and the Emperor's basement in New York to keep offensive language out of this book. When I did a similar inquest with a server in Laos, it got even more impertinent. The data en route from Germany to Laos had arrived far east in Singapore already. Yet Singapore didn't route it within its region to Laos. It gave it a round trip over the world and forwarded the traffic via the USA.

If we wish to strengthen VPN anonymization against nation-state actors, we can chain several VPNs. Even though this would be easy to implement technically, I don't know of any easily usable free tool to set up such a chain. The most viable workaround may be using a virtual machine. A virtual machine is an inner virtual system simulated in software by the outer physical computer. We can then configure the physical computer to use a VPN for accessing the Internet. The (inner) virtual system would inherit the Internet access via the outer VPN without being aware of the outer VPN. Then we could configure the virtual machine to use a VPN as well, which would comprise an inner layer VPN running inside the outer layer VPN of the physical computer. A browser running in the (inner) virtual machine then implements two chained VPNs for Internet access. I will not discuss this technique further. Readers interested in this method may get further advice and guidance from Internet resources.

Using Internet access via a browser running in a virtual machine brings advantages mainly on another battlefield. Surfing the web via a browser running inside a virtual machine offers higher resilience against external attackers trying to compromise and pwn the machine. These techniques, however, belong to the defensive strategy of building a digital fortress visible to powerful adversaries. This strategy is only advisable for computer users of high expertise, access to skillful experts, and substantial financial resources. The typical reader of this course will not meet these requirements and should employ the digital guerrilla tactics recommended in this book: avoid presenting a visible target to the attacker, stay anonymous, and blend in with the crowd. In this guidebook, we assume the reader uses a safe computer that is not compromised because it never drew attention from attackers, not because of exceptional digital fortifications.

Let me sound a last note of caution regarding the VPN.

Many providers of free commercial VPNs advertise easy installation by using their dedicated proprietary software that the user should download and install. In my opinion, however, the user should only use the VPN itself and not let the VPN provider make unchecked changes to the user's system. There is well-established open-source software for using a VPN. So if you need to

install proprietary software issued by the VPN provider, the provider has introduced that demand on purpose. You may then question the provider's motivation to create that demand. Please avoid installing proprietary software with questionable security from providers with questionable intentions. At best, they urge you to accept and install such software to serve their commercial interests. Doing so would entail a loss of control and foil the objective of this guidebook<sup>10</sup>. You should prefer using the VPN tools offered by your operating system, even if using them appears cumbersome. If you need third-party software, please use open-source tools applicable to publicly available standard VPNs, where you freely selected download source and download time. I will suggest some tools for VPN usage in the exercise section of this guidebook.

### 3.3.3 Onion routing, a network of reticent intermediaries

#### Chaining VPNs

In the last section, we mentioned chaining two VPNs. Chaining two VPN servers, whose administrators don't cooperate and which we are free to select, offers increased stability against anonymization attacks. The VPN server addressed first, which is the one providing the outer tunnel layer, doesn't know which target website we are addressing. The second VPN server, supplying the inner shell tunnel, is addressed via the first, outer shell VPN server. This second server does know the targeted website but doesn't know us, the protected client. Thus, the information on what we are accessing and who we are is separately present on two different servers. There is not one single log at one single place that carries the information to blow our cover. Let's assume the servers' administrators don't cooperate and don't cooperate with an attacker. Then the attacker needs access to the web environment of both VPN servers to successfully correlate our out- and inbound data traffic with the corresponding data traffic of the second VPN server. This is hard if we chose VPN servers in different political domains, one from France and one from Vietnam, for example.

If both VPN servers are located in the same political domain, however, our nation-state snooping service could learn, that we are communicating with the first VPN and that the VPN routes our data traffic to the second VPN. The second VPN would then be located in a web environment transparent to the attacker as well, enabling the attacker to successfully correlate all relevant data streams.

Chaining three VPN servers would again substantially improve our security. If all three VPN servers cooperate and join their logs, they could endanger our anonymity. Alternatively, only two need to cooperate, but then the third one must be embedded in an infrastructure transparent for the attacker. In the latter case, the attacker could successfully launch a correlation attack again. It could facilitate the correlation analysis if only a small number of data streams were running through the server, which is not cooperating. Yet this would still be a hard nut to crack.

#### The TOR network

There is a non-commercial network of servers, ready for us to chain. Volunteers run these servers, providing their service for free to fight censorship and to defend privacy and security for web users all over the world. These servers are not common VPN servers, though. The team distributing the server software hardened it against correlation analysis. They emit and receive normalized data packets, i.e., encrypted packets, that share size and structure and are hardly distinguishable for an outside viewer.

The network is called the TOR network. TOR is an acronym standing for The Onion Router. The majority of computers, which are only able to support average bandwidth, can serve as middle nodes for this anonymizing network. The demands regarding bandwidth and availability are somewhat higher for entry and exit nodes. Everyone can configure his system as a TOR server, if

---

<sup>10</sup>In the exercise section, I seem to suggest an action similar to such foolery for Windows systems and install such software for VPN Gate. In that case, however, each component is open-source and free. Furthermore, in the case of VPN Gate, the VPN suppliers and software issuer are completely different entities.

the minimum requirements regarding bandwidth and availability are fulfilled, and can offer the service for the greater public good. Operating TOR nodes is legal in most countries, yet, often enough, their operators are harassed by local authorities making up flimsy justifications for their harassment.

### Using the TOR network

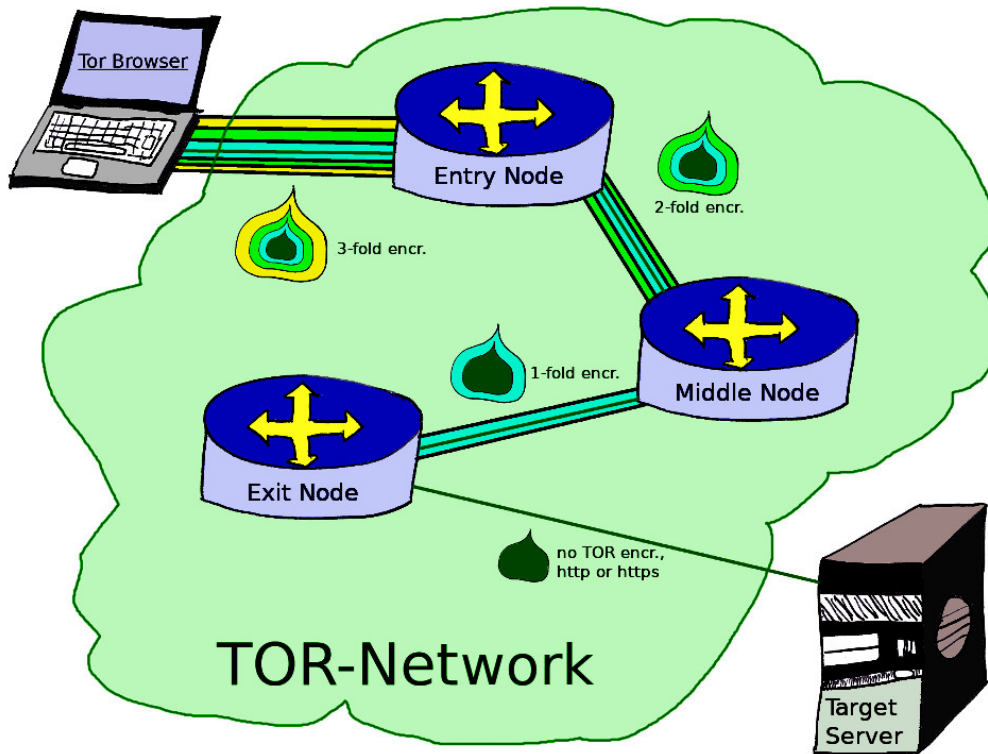


Figure 3.2: The figure illustrates web access via the TOR network. The client, here the TOR-Browser, selects an entry, middle, and exit node for the path from a published list of currently active TOR nodes. After a few minutes, or when the user encounters connectivity problems, the TOR-Browser will select a new TOR path with new nodes. For security reasons, the TOR-Browser will seldom change the entry node.

Now let us use TOR to access our favorite online newspaper. We need a dedicated browser called the TOR-Browser. When we run the TOR-Browser, it loads a list of all currently offered entry, middle and exit nodes and selects an entry node, a middle node and an exit node from that list. The TOR-Browser negotiates a shared AES<sup>11</sup> key with each TOR node in the selected path performing successive Diffie-Hellman key agreements<sup>12</sup>. Our data designated for being bounced around in the TOR network and then to be forwarded from the exit node to the target website will be encrypted threefold in our system.

The browser uses the respective AES keys for the nodes in reverse order to the sequence of the TOR nodes in the outgoing data path. First, the TOR-Browser encrypts the outbound data packets with the AES key negotiated with the exit node. Then it adds the exit node address in plaintext to the encrypted data and encrypts the data packets again using the key negotiated with the middle node. At last, the TOR-Browser inscribes the middle node address to the doubly encrypted data

<sup>11</sup>AES is the most widespread symmetric encryption algorithm nowadays. For symmetric encryption, see 5.1 on page 105.

<sup>12</sup>see: [https://en.wikipedia.org/wiki/Diffie-Hellman\\_key\\_agreement](https://en.wikipedia.org/wiki/Diffie-Hellman_key_agreement)

packets and encrypts once again using the key shared with the entry node. You can visualize the multiple encryptions of the data like layers in an onion.

Each TOR node passed removes its encryption layer (decrypts) with the shared AES key, learns the uncovered address of the next node, and forwards the data to the next node with one encryption layer peeled off, if our package is outbound. The exit node removes the last encryption layer, learns about the target website, and forwards the data in the clear or with remaining HTTPS protection.

The TOR nodes successively wrap the response data with the respective encryption layers on the way back. The exit node receives the response, adds the innermost layer of encryption using its shared AES key, and forwards it to the known previous node. Middle node and entry node add their encryption layers, respectively, and transmit the inbound data back into our direction. We receive the data packages with the triple encryption and can then peel off the encryption layers one after the other, using the keys shared with the respective TOR nodes. Then the response can be rendered in the TOR-Browser window. The protocol ensures that each node only knows its direct predecessor and successor nodes, and only the exit node can know the content of the data packets in case of an unprotected Http transfer. None of the nodes has access to the payload of the data packets, in case HTTPS protection is selected.

### Pitfalls

Please remember that active TOR nodes' addresses are published in openly accessible lists. Therefore we would need additional measures to hide the fact that we use TOR to access the Internet<sup>13</sup>. Our system's data exchange usually takes place with a publicly known TOR node. Thus our ISP or our government's network chaperones know we are using TOR. To hide using TOR from such snoopers, we could use a VPN, for example, preferably a free VPN situated in a country of limited cooperation with our government.

Since freely available lists of TOR nodes do exist, the web server contacted via TOR can also know when a TOR exit node addresses it. But the information the web page can nick from TOR visitors is sparse and can hardly be monetized. Thus some pages, like google.com, for example, reject to respond to requests transmitted via TOR. Their business model, based on violating the privacy of users by excessive data collection, is not supported by TOR visits. Almost all suppliers of free email accounts also belong to this category. They usually block requests to sign up for an account via TOR. Only using this account once at least via an unprotected, conventional IP address would allow them to link the account to a physical person. Only then can they sell grifted data from analysis of email content or satisfy requests by nosy government agencies.

#### 3.3.4 Anonymous access to the web

Our ISP knows a lot about us, including bank details, and usually has direct access to all our data streams running through the Internet. It is a strikingly simple idea to bypass our ISP, who might be nosy and is virtually sitting on our lap when we surf the web.

#### First attempt:

We put on a wig, hat, and sunglasses, glue a fake beard into our face, put on a long coat, and go to an Internet cafe. We pay in cash and sit down in front of a computer with our back towards the closest security cameras. Unfortunately, using the Internet tends to be abysmally restricted in this situation. The Internet cafe business will have locked down their systems and will have configured them restrictively. Probably we won't even be able to connect a thumb drive to the computer and take data home. We wouldn't be able to use TOR or a VPN. We couldn't encrypt anything, and at the latest, when we would need to log in at one of our accounts, our security would be at the mercy

---

<sup>13</sup>I won't describe other means here like using a TOR bridge or using obfuscation. If you want further information on these, please use your preferred search engine to find instructions in the Internet.

of the dicey Internet cafe operators. Safe Internet use is impossible if we cannot trust the hardware we use. So this doesn't work. At least we must be able to use a computer we control.

### **Second attempt:**

We again put on a coat, sunglasses, wig and so forth, avoid facing cameras and use our trustworthy notebook to connect to a public WLAN in a pub, in the train (not on a seat with a reservation on our true name), at an airport, in a shopping mall or the like. This attempt could succeed, given we are aware of the fact we would also have to conceal the MAC address of our device.

Each manufacturer of WLAN interfaces of digital devices ships the interfaces with an individually set MAC address (Media Access Control). Any WLAN we connect to, be it our private one at home or a free WLAN in the train or a pub, identifies and addresses our system by its individual MAC address. There is no technical need for the WLAN operator, however, to snitch and forward this MAC address anywhere, and our WLAN router at home will surely not do that. But any public WLAN access point communicating with our device, possibly even without us being aware of the transient contact, learns about our MAC address. Some department stores may track us in this way by the MAC address of our mobile phones when we stroll past the displayed goods. But let's get back to the computer. Logically, the MAC address of our system is the invisible analog to the license plate on our car.

Only if we bought our system wearing wig, sunglasses and false beard, paid in cash, never used the device to log in to one of our named accounts via a public WLAN, only then can we be sure that there is no database linking our name to the MAC address of our device. The operating system of our device controls communication with a WLAN access point. It would be easy to let the operating system transmit a fake MAC address different from the one inscribed by the manufacturer. That would be called MAC address spoofing. Yet, for the average user, it seems to be tricky in practice to reliably and stably spoof the MAC address. In summer 2017, I tested numerous GUI controlled, free tools, usable on open-source operating systems, to perform such MAC address spoofing<sup>14</sup>. I connected to a WLAN and looked up the data in the access point's logs. With one exception, none of the spoofing tools worked reliably. Right away or after a few minutes, they fell back to transmitting the real MAC address to the WLAN access point, in some cases even in contradiction to what the tool's GUI displayed. In case you depend on spoofing your MAC address with your life or your freedom, be on guard. Only believe what you tested thoroughly! There seems to be a lot of crap around. Only using the TAILS operating system (see 3.4.4 on page 54), I could achieve a reliable MAC address spoofing.

In conclusion, we can state that we can achieve Internet access with robust anonymity via a free, public WLAN if we reliably hide the MAC address of the device from the WLAN operator. As of fall 2017 and for Windows or Linux computers, in a way acceptable for laypeople, this seems to be possible with a TAILS operating system only.

## **3.4 Tools for receptive anonymity**

As for all other layers of the dissidents' pyramid (see figure 1.2), we will now discuss some open-source and free tools for achieving receptive anonymity.

### **3.4.1 VPN Gate**

The use of VPNs is quite common in Asian countries. People there use the VPNs to protect personal surfing habits from unwanted insight by the government, the advertising mafia, or to evade censorship. There is a project at the University of Tsukuba, Japan, aiming to encourage computer

---

<sup>14</sup>MAC address spoofing can in all operating systems also be achieved by direct, manual fiddling with the system configuration. I consider this unacceptable for laypeople, however, and will not treat that method in this guidebook.

users worldwide to configure their computers such that they can act as VPN servers and to offer their VPN service to the public. The project is called "VPN Gate"<sup>15</sup>.

Furthermore, there is an associated project called "Soft Ether"<sup>16</sup> offering free and open-source software for MS Windows that greatly facilitates selecting and using VPNs from VPN Gate. Soft Ether uses the builtin Microsoft Authenticode mechanism to verify the integrity and authenticity of the software during download and installation. As I mentioned in the prior subsection 3.3.2, using such VPNs does only provide security against nosy ISPs and overly curious webmasters of visited web resources. It is not necessarily safe against powerful government agencies. If used to enter into the TOR network via the VPN, it can serve as a security buffer to conceal your using TOR from your ISP and your local network environment. This may aggravate it for governments to find out you are using the anonymizing network TOR. VPN Gate and Soft Ether can act as confidential access to a TOR entry node.

### 3.4.2 TOR and the TOR-Browser

After introducing the concept of onion routing and mentioning the currently unrivaled implementation called the TOR network in the previous subsection 3.3.3, we will now dwell on using TOR.

TOR is a US-based project and gets the major part of its financial support from the U.S. State Department. For many computer users, this is suspicious. However, the software for TOR nodes is open-source, and technically TOR is by far the most advanced one of all open-source anonymizing tools.

TOR is much more than just the anonymizing TOR-Browser. It is a system for using the Internet anonymously in general. You can surf the Internet via TOR, you can chat, you can access emails, you can transfer files, or you can anonymously host Darknet websites using TOR.

On a Linux, you commonly employ a package from your distribution's repositories to install TOR system-wide. If installed from the repositories, the builtin automatic signature check ensures the integrity and authenticity of the download before installation. After installing TOR, TOR secured data streams can run in parallel and independently. The TOR-Browser is an additional independent software package. It comes with its own implementation of the TOR network access and opens its own TOR data streams. Under Linux, as well as under Windows, you should secure the installation of the TOR-Browser manually. You can get the TOR-Browser installation package as well as its digital signature from here<sup>17</sup>. At this particular time, we can identify the relevant public key of the issuer of the software by its fingerprint 0xEF6E286DDA85EA2A4BA7DE684E2C6E8793298290 or shorter and a little less secure by just giving the last eight hex digits 0x93298290. There is a description on how to import it into your GnuPG key repository via console commands on TOR's homepage. You should also be able to find the public key, for example, on MIT's public PGP keyserver<sup>18</sup>, which unfortunately is down every so often.

As mentioned earlier, cautious and suspicious computer users tend not to trust in the US-based vendor Microsoft. They avoid Microsoft's builtin automatic download verification. When dealing with anonymizing and encrypting communication tools, they prefer manual verification of signatures over builtin mechanisms even when working under a Linux. Automatism involving intermediaries, situated between users and developers, always introduce new weak points. They increase the authentication procedure's complexity and increase the number of instances involved in verifying the digital signature. The unneeded third parties water down the close and direct interaction between developer and user in creating and proving digital signatures.

---

<sup>15</sup>see: <https://www.vpngate.net/en/>

<sup>16</sup>see: <https://www.softether.org/>

<sup>17</sup><https://www.torproject.org/download/download>

<sup>18</sup><https://pgp.mit.edu/pks/lookup?search=0x93298290&op=index>



On a Windows, after the manually secured installation of the TOR-Browser, you customarily use the TOR implementation that came with the TOR-Browser also for other TOR applications. You do that by always opening the TOR-Browser first, let it establish the TOR path(s), then minimize the browser window, let the TOR-Browser run in the background and then start the other program(s) needing TOR access.

On a Linux, you customarily use the system-wide TOR installation. Most programs using TOR rely on the respective libraries being present. However, when you run the TOR application "OnionShare" on a Linux (see section 4.3.7 on page 89), you first get a dialog popping up, which will query you if you wish to use the system-wide TOR implementation or an implementation that came with OnionShare.

### 3.4.3 Email access and TorBirdy

Good email providers offer multiple options to access the accounts registered with them. In addition to the usual email clients like Thunderbird, Evolution, Sylpheed, Outlook, Claws, Icedove, Seamonkey, alpine, balsa, Geary, whatever..., they offer access also via a web interface. Better email providers offer better web interfaces that allow blocking client-side scripts in the browser and hence allow to favor security over unnecessary gimmicks.

If the web interface of your email provider only works when client-side scripts are allowed and executed, this widens the attack surface for hackers. In the worst case, hackers or hostile nation-state agencies could exploit bugs in the script interpreter of your browser (or of a bloated email client) to get administrative access to your system, to pwn your computer. If your favorite email provider's web interface prevents usage without client-side scripts, you might want to migrate to another email provider who does not coerce you to lower security.

If you want to use your email account without revealing your whereabouts to nosy snoopers, you should access its web interface using the TOR-Browser, preferably with client-side scripts blocked. For organized crime specialized in housebreaking (or burglary squads of an intrusive intelligence service), it may well be worthy information that today you are accessing your email account from Tijuana, Mexico. If you appreciate the comfort of your favorite email client, you could search the web for a plugin for your client that enables email access via TOR. For the popular email client Thunderbird, you could use the plugin TorBirdy, which allows just that comfortably. Please be aware of the plugin being normal software that requires the due safety means for download and installation. You should check the installer for integrity and authenticity before running it.

At this particular time, you can find the TorBirdy installer and its developer's OpenPGP digital signature here<sup>19</sup>, and instructions about its use and information on the source of the public key of Sukhbir Singh, who signed the installer, here<sup>20</sup>.

### 3.4.4 TAILS

Those of us who often surf the web anonymized by the TOR-Browser, and at times also surf recognizably (nonymously) will sooner or later suffer an anonymity accident. We might be distracted for a moment, mistakenly assume we were using the TOR-Browser, and log in to a forum or an account that would be owned by one of our fictive identities. Oops. Usually, we realize such mishaps right away. We irrevocably burnt the anonymity of our fictive identity. Because you and I don't do anything online that is illegal, this would not be dangerous for us. It could be embarrassing and irksome, however.

There is a powerful tool that can make such anonymity screw-ups less of an occurrence. I highly recommend its usage. It has quite a few other nice features too. The tool is called TAILS<sup>21</sup>.

<sup>19</sup><https://dist.torproject.org/torbirdy/>

<sup>20</sup><https://trac.torproject.org/projects/tor/wiki/torbirdy#VerifyingtheXPI>.

<sup>21</sup><https://tails.boum.org/>

TAILS is the acronym for The Amnesic Incognito Live System. TAILS is a complete operating system, based on a Debian Linux, that is installed on a thumb drive or a flashcard and completely and solely resides on this removable medium. To use it, you must configure your computer to allow booting from such removable media. If you have inserted a TAILS medium at power-on, the computer bypasses the built-in storage media, like the primary hard disk, and boots directly into TAILS. TAILS leaves the hard disk and other builtin storage media untouched. After you shut down the system and remove the TAILS stick or SD-card, TAILS has left no traces on the system that could point to your recent TAILS session, should authorities seize the system.

On a TAILS, any data traffic is routed through TOR unless you explicitly order it to use a plain browser and subsequently click through a red warning dialog. The TAILS operating system comes with a complete working environment like, for example, office software, email client, TOR-Browser, chat client, audio and video editors, and much more. Most privacy tools recommended in this guidebook are part of TAILS. TAILS has been conceived as a tool for dissidents and whistleblowers in a repressive environment and is perfectly usable by IT laypeople. TAILS is indeed perfectly workable; I am currently using it to type and edit this passage of this guidebook.

Generally, you can install any software package on a TAILS that is appropriate for a normal Debian Linux. The installation will be temporary, just for the current session only, unless you employ certain tricks. These tricks are straightforward, simple, and are accessible from the TAILS desktop, yet I will not delve into that here. It is the distinguishing feature of this operating system that a reboot will always lead to the initial state the system was in before the last session. Remember, the A in TAILS stands for Amnesic. This is a protective measure against hostile government action. In many repressively governed nations, domestic intelligence agencies systematically attack and pwn citizens' computers, should they be suspected of belonging to the political opposition. These agencies do this to be able to read citizens' emails and snoop on other forms of their digital communication.

Even in Germany, the government recently founded the agency ZITiS for enabling the government to hack citizens' devices. The government does not disclose to the public what the criteria for selecting the targets of such hostile actions are. I do not know if writing a book on how to avoid being hacked by the government qualifies me as a target. There is some indication it may have done so. Using TAILS is the easiest way to aggravate attacks that aim to covertly introduce changes to our systems.

Furthermore, TAILS allows for a reliable spoofing of the system's MAC address. You could pick up your notebook, a TAILS stick, sunglasses, a wig, and a fake mustache, go to a public access point and use the Internet anonymously. Leave away the stagy accessories for disguise ;-) and I have done so every so often on long train rides through Germany.

The price you have to pay is a more restricted software environment than in a regular Debian Linux, a somewhat slower boot process, and the necessity to abstain from opening too many windows in parallel. For purely technical reasons, data exchange between mainboard and thumb drive via a USB connector is substantially slower than data exchange between the mainboard and the primary hard disk. Additional problems may arise on mainboards with a gigabyte or less of memory because the TAILS system does not come with a swap partition (for good reasons) and cannot support more than a few applications in parallel on scarce mainboard memory. All in all, if you don't adopt what I would call window discipline, you might run into severe congestion of data streams and bring TAILS to a halt.

Let me close this section on TAILS by describing a typical use case. Let's assume an unknown creature, human being, or extraterrestrial reptiloid, had contacted us under a pseudonym. Let's further assume the creature had pointed to a website that seemed interesting to us. Probably our unknown communication partner is good-willed. Yet this could as well be an attack on us. When visiting the website without adequate protection, we would risk having our system pwned by the

attacker. But if we own a TAILS, we can relax and use it and its TOR-Browser to examine the page safely. Even if visiting a maliciously prepared website would compromise TAILS, despite its substantial resilience, the damage would be paper-tiger. A reboot would heal the assault's impact on the amnesic system as long as we don't face a highly sophisticated attack, planting malware into the firmware on one or more of the mainboard's chips.

### 3.4.5 I2P "The Invisible Internet Project"

TOR is the most advanced, the most comfortable, and the most widely used anonymizing network. It would be wise to assume, though, that U.S. government agencies have access to a TOR kill switch and can render TOR unusable in their own discretion, totally or for a selected subset of users. After all, TOR is funded mainly by the U.S. government<sup>2223</sup>.

The threshold will be high for such action because parts of the U.S. administration use TOR and would suffer from the downsides of the shutdown. We can further up the threshold by supporting other free anonymizing networks. By using them and increasing user base, we can improve them to allow for serious, albeit low-intensity use. They could then act as an instant replacement for TOR, should the U.S. government decide to take down or cripple TOR. This would instantly degrade the U.S. government's influence on the worldwide anonymizing habitat. Dear readers, you can personally support the stability and availability of anonymization means by the occasional use of an alternative service, which might still be of somewhat cumbersome usability. This could help to safeguard digital freedom for us all anywhere in the world.

Let me mention some alternative overlay networks for anonymized Internet use. There is Freenet (<https://freenetproject.org/>), Zeronet (<https://zeronet.io/>), IPFS (<https://ipfs.io/>), GNUnet (<https://gnunet.org/en/>), and The Invisible Internet Project I2P (<https://geti2p.net/en/>). Please excuse me when I missed some more. In my opinion, the tool I2P could be best fitted for intelligent non-nerds and stand in line next as a replacement, should TOR be rendered unusable. We can also employ I2P as a complement to TOR.

TOR transmits small, normalized faceless data packets, and commonly, the user doesn't act as a node. In contrast to TOR, all I2P users also act as nodes. I2P nodes always receive and submit large packets. The I2P packets are assemblies of many separate snippets belonging to different data streams. The nodes decrypt and unpack the received packets into the snippets, and retain the snippets addressed to the node. Then they reassemble new aggregates from received snippets destined for other nodes and forward the newly assembled large packets to other nodes. The large packets are encrypted, of course. They are completely opaque to eavesdroppers. I2P thus forwards the snippets of a single data stream simultaneously over several different paths. The idea is to aggravate separating the anonymized data streams for the attacker. The attacker has hardly a chance to tag a single data stream transmitted via I2P. Developers of anonymity networks seem to relish plants of the botanical family of Amaryllidaceae, so this kind of clumpy routing is called garlic routing.

A powerful surveilling agency, powerful enough to control the I2P node's web environment, could easily subtract the emitted volume of data from the received volume and calculate the amount of data retained at the node. Thus compared to TOR (the onion), I2P (the garlic) only offers a marginal advantage against Website Fingerprinting (see section 3.6.1 on page 76). The clumpy garlic routing of I2P is aggravating attacks, however, in which a data stream would be tagged by some method, data rate modulation, for example. The attacker would always tag several streams at a time and would do so incompletely. This aggravates so-called Confirmation Attacks. These are attacks in which exit and entry nodes, or services controlling their web environment at least,

<sup>22</sup><https://www.torproject.org/about/sponsors/>

<sup>23</sup><https://www.businessinsider.com.au/claims-tor-funded-by-us-government-agencies-2018-3?nojs=1>

cooperate with the attacker. The attacker tags the stream close to or at the entry node, and the tags are detected close to or at the exit node. In the end, the largest security advantage of I2P, in my view, is the large and erratically distributed latency of the anonymized data streams, which in return somewhat degrades comfort and usability.

Occasionally, you hear the statement, I2P would be a complement and not a rival of TOR, because its developers designed it for a direct one to one connection between Internet users. TOR, in contrast, would be designed for anonymized Internet access to websites. I don't share this opinion. TOR is well suited for one to one connections as well. This TOR mode is called rendezvous type interaction. On the other hand, I2P is only less well suited for anonymously surfing the web, because the number of users is much smaller than with TOR. When researching for this section of the book on a Sunday evening in December 2017, I found that about 200 I2P nodes were involved in routing my data stream, about 30 of which were residing in Germany. Furthermore, at the Sunday in December 2017, there was only one, slowly responding I2P standard exit proxy from Norway publicly available.

Nevertheless, user groups can set up speedy exit nodes or proxies themselves and share those exit nodes within the group. That is not easy enough, though, to describe and recommend it in this guidebook for laypeople. In case the U.S. government would take TOR down, we would soon get a similarly high user number in I2P, resulting in a massively increasing number of available nodes. Maintainers of I2P would presumably quickly address the higher need for fast exit nodes connecting to the plain Internet. Then, the transitional problems would dissolve rapidly.

I2P works. Please, dear readers, use I2P (or another open-source and free tool) occasionally to anonymize your web traffic. Only by such actions can we sustain the pressure on powerful snooping and surveillance lobbies, to leave TOR untouched. We need to retain the capability of other anonymizing networks to quickly absorb TOR refugees on day X and to ramp up their capacity.

### 3.5 Exercises for receptive anonymity

This exercise section relating to the second layer of the pyramid for dissident protection encompasses the largest number of exercises of all layers. Here you will practice receptive anonymity for protecting your read access to Internet sources. In itself, this is a respectable goal to achieve. Achieving this goal will open numerous new options already for your practical everyday Internet use.

#### 3.5.1 RECx1, trace your data packets on their way to the destination

*Difficulty: medium.*

*Time: about 30 min.*

*Prerequisites: all introductory exercises 2.5.1.*

*Note: The completion of this exercise is a prerequisite for another exercise.*

**For Windows:** Open a console. On newer Windows versions, the main console application is called somewhat boastfully PowerShell. There is also a simpler, more pristine console on Windows called command prompt. You will find the console versions, if installed, within the other applications installed on your system. On Windows, the console command we will be using is called "tracert", which is usually installed already out of the box.

**For Linux:** For the typical Linux user utilizing the console is daily business. On a Linux, however, the basic network analysis tool traceroute usually is not present by default and has to be installed from the repositories first (on a standard Debian, for example, enter "apt-get install traceroute" in the console).

1. Please open a Browser window now and surf to the homepage of your favorite online news outlet.
2. Copy the web address and paste<sup>24</sup> it directly after the respective tracing command (tracert or traceroute) into the console and finally execute the command by hitting the return key. In my case, for example, this would be `traceroute zeit.de`, or in Windows PowerShell or command prompt this would be `tracert zeit.de`. Watch for the responses piling up in the console and observe.
3. The first line of the tracing response reveals the IP address (received from the respective Domain Name System server) associated with the URL. In my case it was 217.13.68.251 from “traceroute to zeit.de (217.13.68.251), 30 hops max, 60 byte packets”.

If you like, you may geolocate one or the other - may be all - IP addresses in the path to find out the whereabouts of these servers. There are numerous pages on the web, which offer such IP geolocation for free. Your preferred search engine will readily suggest a suitable web page when responding to the search string "free IP geolocation".

### 3.5.2 RECx2, access the Internet via a free VPN

*Difficulty: medium to hard.*

*Time: Reserve at least one to two hours for doing the exercise for the first time.*

*Prerequisites: none.*

*Note: The completion of this exercise is a prerequisite for many other exercises.*

Here I suggest two versions of this exercise, one version for Windows and one version for Linux. We will use the prospects the free project VPN Gate offers. This is not an easy exercise, and we are just at the beginning of this course. Thus, as an exception, I will give quite explicit and finely grained instructions.

The instructions for Windows and Linux differ substantially. However, we will begin with a short common introduction, including a brief description in running text on how to download OpenVPN configuration files for manual usage.

Then we split the instructions into three separate parts. These parts are dealing with Linux and manual use, Windows and manual use, and Windows and software supported comfortable use. In all three parts, we will deal with instructions to adjust the software environment first. I will give these instructions as bulleted lists.

Then we will look at the second task, bringing the OpenVPN configuration files downloaded from VPN Gate into play and connecting to a selected VPN. I will describe this second task for the manual use versions in the form of numbered lists of instructions. The software supported comfortable usage in Windows is largely self-explaining and will be illustrated by screenshots. Putting a VPN to use in this version will be done by just double-clicking on an entry in a list offered by the supporting software.

In the end, the explanation paths will be joined again for a common exercise closure, including an optional practical use case. When reading the instructions for this exercise, you may skip the parts that don't apply to your computing environment.

#### **Common introductory part:**

Please use your browser to open the VPN Gate project site of the University of Tsukuba. Your preferred search engine will help you to find this site.

Please inquire about the nature of the project and the ambition of the people advancing and maintaining this project. Inquire also who is leading the project (Dr. Daiyuu Nobori).

<sup>24</sup>The IT experts have placed another little trap for laypeople. In most console applications, ctrl+c and ctrl+v for copy and paste don't work. You have to use shift+ctrl+c and shift+ctrl+v, respectively...

Download some currently offered (short-lived) OpenVPN configuration files from the VPN Gate project page<sup>25</sup>. As of today (Apr. 2020), you may click on the entry in the seventh column on "OpenVPN Config File". (In case the page's design has changed in the meantime, I am sure you will find the appropriate place to click then.) On the page appearing after this click, you select a specific configuration file by a right-click and download the configuration file. Store the downloaded configuration file in a folder of your choice. Do this for the configuration files of some other OpenVPN servers listed on the VPN Gate page as well.

### **For a Debian based Linux:**

If they are not yet installed on your system, please install the following three packages from the repositories:

- network-manager-openvpn
- network-manager-openvpn-gnome
- openvpn

Now let's make the VPNs usable on your Linux system.

1. Click on the symbol of the network manager and then select "Edit Connections" or the corresponding entry in your language version.
2. Add a new connection and select: "Import a saved VPN configuration".
3. Then select one of the recently downloaded configuration files from VPN Gate.
4. Now the selected VPN connection is selectable from the network manager's drop-down menu.
5. Do this for the other downloaded configuration files as well.
6. In the network manager, select one of the VPN connections just imported.

Your system will then try to establish the VPN connection and negotiate the key for the tunnel. A few seconds later, you will get a message stating either success or else failure regarding the task. From my experience connecting to some of the servers might not work. If you tried connecting to one of those, don't worry and try another one. The VPN servers from Japan or South Korea high up in VPN Gate's list have always worked for me.

If the VPN is up and running, visit one of the many free websites<sup>26</sup> that show you the IP address under which you are visiting the site. Verify that your system indeed routes your access through the chosen VPN server.

### **For Windows (manually set VPN connection):**

We will now prepare the software environment on your Windows to allow for using the downloaded VPN configuration files. Possibly you might have used VPNs on your system before, and there might still be running VPN related leftover processes in the background. Different tools for VPN usage tend to be incompatible and break each other. Thus, you should first activate the Windows Task Manager, search for VPN related processes currently running and end, stop, or kill all of those that might be competing.

Please continue by downloading and installing the free and open-source OpenVPN client of the company "OpenVPN Technologies, Inc.". You may search the web for "OpenVPN open-source free download" or something similar. Such a search gave me the first web address to dig further (April 2020). After an evasion slalom around some attempts to divert me to commercial services, I found the download site for the free and open-source software. (This is revenue territory after all.) Maybe you can still find it here<sup>27</sup>. If not and the site has moved, your search engine will surely help you.

---

<sup>25</sup><https://www.vpngate.net/en/>

<sup>26</sup>see, e.g., <https://www.iplocation.net/find-ip-address>

<sup>27</sup><https://openvpn.net/community-downloads/>

- Download the OpenVPN installer and its OpenPGP digital signature.
- Optional - Get the public key of the signer of the software and check the downloaded installer and its signature against the key. (When I double checked on April 24, 2020, I found they screwed up on correctly signing the binary in OpenPGP's mayhem of keys, subkeys, and expiration dates - GnuPG wouldn't pick the intended subkey and failed to verify. The Windows installer of OpenVPN will also use the builtin Windows authentication automatism during installation, so you could at least rely on that one and install.)
- Execute the installer.
- The icon of the OpenVPN GUI should now be visible on the desktop.

You will have to copy the downloaded OpenVPN configuration files to the folder "C:\Program Files\OpenVPN\config" later. The system should have created this folder during the successful installation of OpenVPN.

Now that we installed the software for using VPNs, we can use the downloaded configuration files. The step by step procedure given below may be subject to frequent change. I had to update the description from the German edition of this book in the translation to accommodate substantial changes. OpenVPN Technologies, Inc. introduced them during the last three years. The description below dates from April 24, 2020. If the company introduces new changes until you read this, you may have to adapt your course of action to the future pattern anew.

1. Copy the downloaded OpenVPN configuration files to the folder:  
C:\Program Files\OpenVPN\config.  
Windows will probably inform you that you need administrator privileges to write to this folder and ask you for some confirmation of that. It will then take your click on the OK-button for an authentication (goodness!) and perform the copy operation.
2. Now click on the icon on your Desktop labeled "OpenVPN GUI". A symbol for OpenVPN will appear in the lower right corner of the screen in the System Tray. A right-click on this icon in the System Tray will trigger a popup menu offering references to the VPN configuration files located in the folder "C:\Program Files\OpenVPN\config" for connection.
3. Select one and click on "connect". The system will then try to establish the VPN connection and negotiate the key for the tunnel.
4. If you completed the last step successfully, the system will activate the VPN connection and will indicate the connection status on the screen. In some rare cases, the VPN server will ask for a user name and a password. In this case, you should fill in "vpn" in both fields. On success, the system will explicitly inform you, and in the lower right corner of the screen in the System Tray, you will see a small screen icon with green filling.

From my experience connecting to some of the servers might not work. If you tried connecting to one of those, don't worry and try another one. The VPN servers from Japan or South Korea high up in VPN Gate's list have always worked for me.

If the VPN is up and running, visit one of the many free websites that show you the IP address under which you are visiting the site<sup>28</sup>. Verify that your system indeed routes your access through the chosen VPN server.

You can find an alternative, nicely illustrated albeit quite outdated description of the connection procedure on VPN Gate's pages here<sup>29</sup>.

#### **For Windows (integrated comfort solution for VPN Gate only):**

Download the current stable version of the "SoftEther VPN Client Software", as of today this would be the URL: <https://www.vpngate.net/en/download.aspx>.

<sup>28</sup>e.g., <https://www.iplocation.net/find-ip-address>

<sup>29</sup>[https://www.vpngate.net/en/howto\\_openvpn.aspx#windows](https://www.vpngate.net/en/howto_openvpn.aspx#windows)

- Select the version of the SoftEther client for download, which includes the VPN Gate client plugins. Expand the downloaded zip-archive.
- Regrettably, Dr. Nobori relies on and uses the builtin Windows authentication, trusts Microsoft, and does not supply a digital signature himself for his software. So you don't have to verify an OpenPGP signature. (When I asked him for one, he wouldn't answer. From here on, you could skip this instruction and rely on instructions, tutorials, and explanations on VPN Gate's pages instead.)
- Execute the installer and mind the Windows messages regarding the authentication of the installer. I suggest dropping an icon for SoftEther onto the desktop. The installer will automatically integrate the plugin that came with it.
- Execute SoftEther by clicking on its icon.

Possibly the window that is shown in figure 3.3 or a similar Window will open:

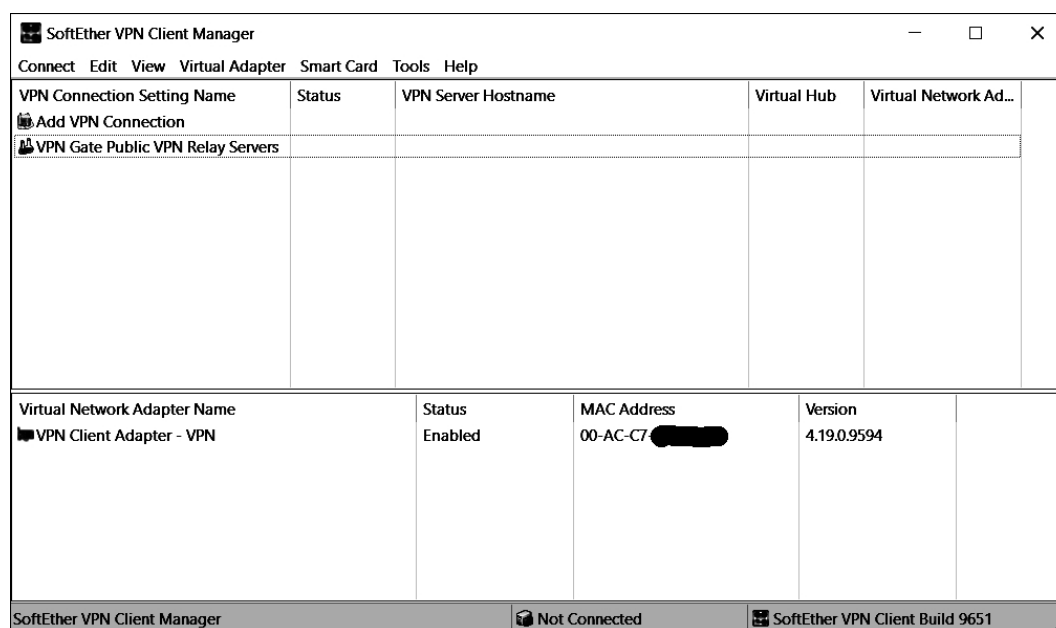


Figure 3.3: The figure shows the SoftEther client window after starting the program. For privacy reasons, I blackened parts of the MAC address.

The next window shown in figure 3.4 opens after a double click on "VPN Gate Public VPN Relay Servers". In this window, you can select one of the offered servers. The servers are offered by many volunteers, predominantly from Asia.

Select one server. Then click on the button labeled Connect to the VPN Server. Watch out, though, not all servers work as expected. Some servers were not accepting connections. In one case, a Russian server obtrusively redirected my requests to unwanted web pages. A server from Taiwan did not forward my TOR traffic. A Chinese server blocked all my HTTPS requests and, of course, TOR access and would only relay Http plaintext requests. Vietnamese servers were somewhat slow but surprisingly appeared uncensored. Those are my favorites. South Korean or Japanese VPN servers generally supply the fastest service. Following my observations, the servers are subject to the restrictions that regrettably apply to Internet traffic in their countries.

After contacting the VPN successfully, you get a message telling you the internal IP address on the selected server. Your relayed access operations to web resources appear as originating from the IP address of the VPN server. The SoftEther window shown in figure 3.5 displays this IP address in the second column of the list. In the case shown in the figure, this is the 184.22.xxx.xxx of the server in Bangkok I connected to in fall 2017.



VPN Gate Academic Experimental Project Plugin for SoftEther VPN Client

**VPN Gate Public VPN Relay Servers** Academic project at University of Tsukuba, Japan. 筑波大学 University of Tsukuba

Gain freedom access to Internet by using VPN connection via Public VPN Servers provided by volunteers around the world. Bypass your local malfunctioning firewall's packet blocking, and hide your IP address safely.

VPN Gate Academic Web Site

Refresh List

**133 Public VPN Relay Servers on the Earth! (Updated at 2017-11-27 10:02:36)**

| DDNS Hostname           | IP Address (Hostname)      | Region             | Uptime   | VPN Sessions | Line Speed |
|-------------------------|----------------------------|--------------------|----------|--------------|------------|
| vpn380772303.opengw.net | 59.10.██████               | Korea Republic of  | 0 mins   | 3 sessions   | 12.0 Mbps  |
| vpn859061045.opengw.net | 222.103.██████             | Korea Republic of  | 0 mins   | 1 sessions   | 10.7 Mbps  |
| vpn201040081.opengw.net | 124.121.██████             | Thailand           | 0 mins   | 0 sessions   | 20.8 Mbps  |
| vpn959922015.opengw.net | 120.26.██████              | China              | 47 days  | 15 sessions  | 8.7 Mbps   |
| vpn842616377.opengw.net | 211.185.██████             | Korea Republic of  | 0 mins   | 1 sessions   | 12.0 Mbps  |
| vpn786386026.opengw.net | 116.220.██████ (116-220... | Japan              | 3 days   | 12 sessions  | 35.4 Mbps  |
| vpn965536718.opengw.net | 46.39.██████               | Russian Federation | 18 days  | 24 sessions  | 2.0 Mbps   |
| vpn548531552.opengw.net | 203.241.██████             | Korea Republic of  | 3 days   | 4 sessions   | 21.9 Mbps  |
| vpn168611145.opengw.net | 69.164.██████              | United States      | 2 days   | 46 sessions  | 14.3 Mbps  |
| vpn272049352.opengw.net | 110.132.██████ (110-13...  | Japan              | 1 days   | 2 sessions   | 75.4 Mbps  |
| redfoxy.opengw.net      | 45.76.██████ (45.76.██...  | United States      | 14 hours | 88 sessions  | 47.5 Mbps  |
| vpn198278632.opengw.net | 59.16.██████               | Korea Republic of  | 4 hours  | 3 sessions   | 0.0 Mbps   |
| vpn546378679.opengw.net | 180.50.██████              | Japan              | 2 days   | 8 sessions   | 14.0 Mbps  |
| vpn352078826.opengw.net | 184.22.██████              | Thailand           | 4 hours  | 2 sessions   | 28.8 Mbps  |
| vpn587888113.opengw.net | 182.227.██████             | Korea Republic of  | 14 hours | 7 sessions   | 53.5 Mbps  |

A VPN Server with higher Line Speed (measured by Mbps) and smaller Ping result are usually more comfortable to use. You might be able to browse websites which are normally unreachable from your area if you use VPN servers that are not in your area.

Proxy Settings

**Connect to the VPN Server**

Implemented as a plug-in for SoftEther VPN. (c) VPN Gate Project at University of Tsukuba, Japan.

Figure 3.4: The figure shows the pool of available VPN servers. We may select one of those. To protect the privacy of the volunteers offering their computers, I wiped the last two blocks of their IP addresses in the screenshot.

SoftEther VPN Client Manager

Connect Edit View Virtual Adapter Smart Card Tools Help

| VPN Connection Setting Name       | Status    | VPN Server Hostname                      | Virtual Hub | Virtual Network Ad... |
|-----------------------------------|-----------|--|-------------|-----------------------|
| Add VPN Connection                |           |  |             |                       |
| VPN Gate Public VPN Relay Servers |           |  |             |                       |
| VPN Gate Connection               | Connected | 184.22.██████ (Direct TCP/IP Connection) | VPNGATE     | VPN                   |

| Virtual Network Adapter Name | Status  | MAC Address     | Version     |
|------------------------------|---------|-----------------|-------------|
| VPN Client Adapter - VPN     | Enabled | 00-AC-C7-██████ | 4.19.0.9594 |

SoftEther VPN Client Manager 1 VPN Sessions SoftEther VPN Client Build 9651

Figure 3.5: The figure shows the SoftEther window after successfully connecting with a VPN Gate server from Bangkok, Thailand.

Now please visit a web page for free geolocation<sup>30</sup> and verify that your web access indeed appears to come from the selected VPN server's IP address.

#### **Merged continuation for Linux and Windows:**

Go on to read your favorite online newspaper for a while via the VPN from VPN Gate.

Now please disconnect the VPN and use the command `tracert` with the IP address of the VPN server (`tracert` in Windows) from a console. The result will show you the routing from your system to the VPN server.

#### **Optional:**

Activate the VPN anew and try to register a new email account with an arbitrary email provider.

*Hint: Some providers block such attempts if they know you are accessing their service via a VPN. Yet VPN Gate's servers' services are usually available on a short time basis only, are often offered by private web users like you and me. Thus there is a good chance the servers may not be known as VPN servers to the chosen email provider.*

Use this account to send an email to your primary nonymous email address. If you decide to abandon this account, please delete the account to unburden the provider.

### **3.5.3 RECx3, safely install the TOR-Browser**

*Difficulty: simple.*

*Time: about 20 min.*

*Prerequisites: ITSx6, RECx2.*

*Note: The completion of this exercise is a prerequisite for many other exercises.*

Free citizens of current democracies have the right to walk the street without being checked by the police for no specific reason and have the right to move in cyberspace anonymously as well. Nevertheless, after the Snowden revelations, spokespeople of U.S. intelligence services publicly threatened to intensively surveil (hack) specifically those citizens who engage in encryption and anonymization techniques on the Internet. I have to honor, that my German home country's services refrained from such impertinent threats up to now.

Regardless of the substance of such threats, in this exercise, we will not perform our downloads in the open anymore. We will not present ourselves as a target to such barefaced intelligence agencies. A leak showed that a U.S. intelligence agency maintained a database of insubordinate Internet users. The agency had monitored them when researching on or when downloading TOR. The intelligence agency labeled them as extremists. Probably this applies as well to other protective tools mentioned in this book. Possibly, VPN Gate is excluded from this threat, since it is not directed at protecting against nation-state surveillance and would not be fit for this purpose anyways unless combined with other tools. Let us now use VPN Gate as an innocuous-looking entry port into the world of tools intelligence services label as extremists' tools in an attempt to intimidate us and make us renounce our civil rights.

1. First, activate a free VPN of your choice for your system.
2. Start your favorite browser, clear all cookies, and open a new private window. Google search may reject to serve you via a VPN. Find and use a good search engine, which works for you via a VPN. Go to the website of the search engine, preferentially another one, than you commonly use.
3. Search using the search string "Download TOR-Browser", for example, to find the TOR-Browser's download site. The search engine will probably refer you to the web location given in the footnote<sup>31</sup>.

<sup>30</sup>for example, <https://www.iplocation.net/find-ip-address>

<sup>31</sup><https://www.torproject.org/download/>

4. Download the installation packet for the TOR-Browser in the version you need as well as its digital signature and store both files in a folder of your choice.
5. Disconnect from the current VPN and connect to another VPN of your choice.
6. Now inquire which public OpenPGP key you need to check the signature with, download the public key and import it into your implementation of GnuPG.

*Note: Possibly, you were referred to the site <https://www.torproject.org/docs/signing-keys.html.en> presenting the public keys, specifically to the key with fingerprint “0x93298290”, if this is still up to date when you inquire.*

7. Check the triple of the public key, installation file, and signature file for consistency (=check the signature). On a positive outcome, you may continue. Else, if the verification repeatedly fails, double-check the procedure on your side. After double-checking and consistent failure, you may inform the owner of the respective key pair about the incident and politely ask for clarification.
8. **Windows only:**  
Run the installer.

**Linux only:**

- (a) Unpack the installation archive in a folder of your choice.
  - (b) Drop an icon onto the desktop or into the taskbar, which links to the executable script in the TOR-Browser’s folder Browser/start-tor-browser. You may use the emblem that came with the installation packet.
9. **both systems again:**  
Disconnect the free VPN and run the TOR-Browser.
  10. Visit your favorite online newspaper or another website of your choice anonymized by the TOR-Browser.

### 3.5.4 RECx4, monitor TOR-Browser’s operation

*Difficulty: simple.*

*Time: 10-20 min.*

*Prerequisites: RECx3.*

1. Please start the TOR-Browser and visit a web site showing the IP address accessing it.
2. Click on the little padlock symbol at the left of the URL field. As a response, the browser displays the IP addresses of the entry node, middle node, and exit node of the current TOR path. Please compare the exit node’s IP address as displayed from the TOR-Browser with the IP address, the visited website shows. Both should show the same IP address.
3. Surf to some websites you like and observe how often the TOR-Browser changes the nodes and if all of them are changed at the same time and with the same frequency.
4. Find out how you can trigger such changes yourself.
5. Optional 1: Using the TOR-Browser search the web for the developers’ explanations on why the entry node change frequency differs from the frequency of change of the other nodes. Find out how you can trigger an entry node change.
6. Optional 2: Using the TOR-Browser search on the Internet for a list of currently active TOR nodes and verify, that the three nodes currently used in your TOR path are on the list.

### 3.5.5 RECx5, combine VPN and TOR-Browser and block script execution

*Difficulty: medium.*

*Time: about 20 min.*

*Prerequisites: RECx2, RECx3.*

*Note: The completion of this exercise is a prerequisite for some other exercises.*

We will practice concealing our use of TOR anonymization from our nosy local web environment in this exercise. We will hide it from our ISP, operators of a free WLAN we may be using, operators of nearby routers we use, or from agencies wiretapping our Internet connection from a black SUV parked on the other side of the street.

1. Connect to a free VPN, for example, from VPN Gate, which would preferably have the server outside your political domain.
2. Start the TOR-Browser. (Your system will route all your in- and outgoing traffic, including the TOR traffic, through the VPN.)

*Hint: If the web environment of the VPN or the VPN server itself is censored or surveilled (when situated in China or Taiwan, for example), TOR may be inaccessible. For the majority of VPNs from VPN Gate, access to TOR is possible.*

3. Now read your favorite online newspaper without having your ISP or other local snoopers know you are using the TOR-Browser for anonymization.
4. To enhance your security, block the execution of client-side scripts pushed to your browser by visited websites.

*Hint: Maliciously prepared scripts can exploit bugs in the browser's script interpreter to pwn your system for mining bitcoins at your expense, for just spying on you, or for turning your system into a bot to attack innocent third-party web servers.*

Please open the global security settings in TOR-Browser now.

Open Browser Preferences by clicking on the icon with three parallel horizontal lines in the upper right corner of the window or select Preferences from the menu (Edit→ Preferences). In Preferences, select Privacy & Security, scroll down somewhat till Security, and set the level to safest. Study the explanations given in the settings. Do this so you can lower safety in an informed way should a web resource refuse to serve you in the most secure setup. Unfortunately, the user interface for browser settings appears to be subject to frequent change. Chances are, you will have to adapt to changes that may have occurred since I wrote these lines.

5. To control client-side script execution in a fine-grained manner, you may click on the small red symbol in the upper right corner of your browser window (see figure 3.6). It provides access to the settings of the NoScript plugin that came bundled with the TOR-Browser. Experiment with the settings and stick to the habit of enforcing the tightest possible restrictions for what visited websites can do on your computer. But watch out, global browser security settings and NoScript plugin's settings may interfere with one another in a somewhat intransparent way. The user interface for the NoScript settings may be subject to change. You may have to adapt to changes that may have occurred since I wrote these lines.

Check which of your favorite web pages still works with these strict security settings. Maintainers of these websites apparently respect their guests' desires for security and privacy when surfing the web.

You are capable of surfing the web with a formidable level of security now. There is still a long way to go with this book's training program. However, on completing this exercise, you reached the first substantial milestone.

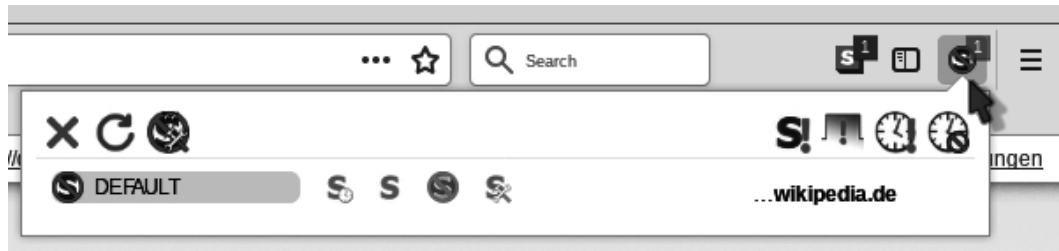


Figure 3.6: The figure shows NoScript's popup dialog accessible by clicking on the Interdiction sign. It allows controlling client-side scripts pushed to your browser by visited websites. Here you get access to fine-grained control in NoScript's plugin settings.

### 3.5.6 RECx6, check your mails without revealing your position

*Difficulty: simple.*

*Time: about 10 min.*

*Prerequisites: RECx3.*

*Note: The completion of this exercise is a prerequisite for a few other exercises.*

1. Start the TOR-Browser. If the TOR-Browser has been running already, please restart it or select New Identity from the File menu. That will effectively decouple prior website visits or previous logins from your next activity for snoopers' eyes.
2. Visit the website interface of your email provider and log in to your email account.
3. Read new emails from your inbox or write and send emails from your account. TOR helps you conceal your physical location.
4. Close the TOR-Browser or select New Identity from the File menu. By this, you make sure that you decouple future activity with the TOR-Browser from your login to the email account for the surveillant's eyes.

### 3.5.7 RECx7, install TorBirdy in Thunderbird

*Difficulty: simple.*

*Time: about 15 min.*

*Prerequisites: RSx6, RECx3 (on Windows).*

*Note: The completion of this exercise is a prerequisite for one other optional exercise.*

This exercise is optional. For this exercise on TorBirdy, you need the Thunderbird mailer. If you don't want to install Thunderbird, skip it.

1. Start the TOR-Browser and use a good search engine that doesn't block requests submitted via TOR and search for "download TorBirdy". Possibly you will be referred to this website<sup>32</sup>. In case you also get this URL<sup>33</sup> among the search results, you got a good search engine, and you should prefer this source. This source offers an OpenPGP digital signature by the developer along with the installer for the Thunderbird extension `torbirdy-current.xpi`.
2. Download the xpi-file and its digital signature.
3. Please get the public key of the key pair the developer used to digitally sign the installer with. Here and now, April 2020, this would be the key of Sukhbir Singh with fingerprint `0xB01C8B006DA77FAA`. Search for a trustworthy website posting it, or get it from a public keyserver.

<sup>32</sup><https://addons.thunderbird.net/en-us/thunderbird/addon/torbirdy/>

<sup>33</sup><https://dist.torproject.org/torbirdy/>

4. Verify the correctness of the digital signature. If it is correct, you may install the extension in Thunderbird. You will surely find your way through the menus of Thunderbird to the user interface for installing extensions.
5. Make sure the plugin is activated.
6. On a Windows, make sure the TOR-Browser is running in the background to be able to lend its TOR access to TorBirdy.
7. Access your email account in the client Thunderbird and send or receive emails without revealing your physical location.

*Note: Some email providers block access via TOR to accounts hosted by them. Good email providers don't. In case your email provider does not belong to the good ones and forces you to reveal your physical location to lord only knows who, you may wish to migrate to a good, free email provider.*

### 3.5.8 RECx8, force TOR-Browser to use an exit node in a country of your choice

*Difficulty: medium to hard*

*Time: about 45 min.*

*Prerequisites: RECx3.*

1. Inquire on the web using your TOR-Browser how you can induce TOR-Browser to select an exit node in a country of your choice (given there are appropriate nodes available in that country). If you found a suitable web page explaining how to do that, select a country for your exit node and do it. You might have to edit a TOR system file for that purpose.
2. Restart your TOR-Browser using the new settings and check that the current exit node is indeed located in the country you selected.
3. Have a look into the online news outlet "theintercept.com" via the newly reconfigured TOR path.

*Note: At this news outlet, a close confidant of Edward Snowden is a major contributor. Articles on certain topics in The Intercept may be somewhat arduous to read for people not sharing The Intercept's political orientation.*

4. Please release the choice for the location of the exit node again after finishing the look into The Intercept.

### 3.5.9 RECx9, find a TOR Darknet search engine for onion sites

*Difficulty: medium.*

*Time: minimum is 10 min.*

*Prerequisites: RECx3.*

1. Using the TOR-Browser research on the Internet what a ".onion" website is.
2. Use a good search engine to search for Darknet search engines. Try some of these for Darknet searches of your choice.

*Hint: In the Darknet, you will find many resources that are plainly broken, some might have been compiled without due diligence and some may be of poor quality or present false claims. (Some content on operational security made my hair stand up, some other was sound.) But please be forgiving and patient.*

*Websites of large corporations in the plain web may function well in your browser in comparison, but they usually only contain platitudinous PR-crap after all. You should be careful about what to believe from those sources as well.*

### 3.5.10 RECx10, practice browsing the TOR Darknet

*Difficulty: simple.*

*Time: from about 5 min.*

*Prerequisites: RECx3, RECx9.*

Using a Darknet search engine, find a forum of your choice and read some contributions. Please don't post any comments yourself yet.

### 3.5.11 RECx11, install and use the anonymizing tool I2P

*Difficulty: medium to high.*

*Time: about 60 min.*

*Prerequisites: access to your firewall settings.*

*Note: Behind a restrictively configured firewall that you cannot control (prevalent in workplace networks), I2P may not be accessible to you. The completion of this exercise is a prerequisite for a few other exercises.*

1. Use your preferred search engine for inquiries on I2P.
2. Download the Installer of I2P for your operating system and install it according to the instructions given on I2P's homepage.  
On a Linux, you can partially secure the installation using builtin automatisms. You can import the public key needed to verify the digital signature into the Linux key repositories, as described on I2P's web page. Then you can use it employing the builtin Linux authentication schemes. If working on a Windows, you authenticate the downloaded installer by verifying the OpenPGP digital signature with the key of activist "zzz".
3. Prepare a separate user profile in your favorite browser for using I2P. In this user profile, you configure the necessary proxy settings according to the description given on I2P's pages. If done correctly, this allows for anonymously surfing the web via I2P and for roaming through the I2P Darknet. Be patient. I2P's latency is substantially longer than TOR's.
4. Visit a website in the clearnet, which reports the IP address which you submit to websites to respond to. Make sure that this is the IP address of an I2P exit proxy and not your real IP address.
5. Take a look at the somewhat confusing so-called I2P router console and find the list of I2P nodes currently usable for your I2P traffic. Identify your system in the list of nodes.

When I tested the functions of I2P around the turn of the year 2017/2018, the Susi Mailer, the I2P torrents, and the anonymous Internet access worked nicely right away. With some fiddling, I got the plugin for the purely I2P internal, encrypted mailer I2P-Bote functional. Other plugins and accessories were dysfunctional or broken, pointed to broken links, or the module's source code wouldn't compile or link because of library incompatibilities. That is annoying. The plugins' and accessories' developers surely have their hearts in the right place, but the implementations frequently are not good.

To which extent you, dear reader, will be able to use I2P largely depends on your disposition. Those readers who are die-hard non-nerds will turn away in horror. Others may use I2P occasionally and keep an eye on the project with goodwill. But one or the other reader will immerse himself deeply in the world of the active, the chaotic, and the fascinating subculture of resistance against digital surveillance, digital paternalism, digital exploitation, digital coercion, and digital oppression. Who knows, a few of you may even offer an I2P exit proxy in the future.

### 3.5.12 RECx12, set up two TAILS USB sticks

*Difficulty: medium to difficult.*

*Time: at least 60 min.*

*Prerequisites: RSx7, RECx3, two USB memory sticks of at least 8GB capacity are required.*

*note 1: the completion of this exercise is a prerequisite for many other exercises.*

*note 2: as of version 3, TAILS is based on a 64 bit Debian Linux. The processors of older computers (> approx. 10 years) sometimes only support 32 bit operating systems. If this is the case with your computer, you have to choose a newer one for the operation of TAILS.*

Open the TOR-Browser for Internet research. Find out which website offers TAILS for download. Today and on my system, the following address is suggested to me by my search engine: <https://tails.boum.org/install/download/>. For further description, I refer to the English version. You can choose the version of the website in your language if available, in case you need that to facilitate the exercise.

1. Start the following download now in the TOR-Browser and save it in a suitable folder of your system:

Download Tails x.x. ISO image (1.x GB)

This download will take some time, especially via TOR, and should run in the background.

2. Select "New private window" in the File menu of the TOR-Browser and click on it to open a new browser window. First, type the URL of your search engine in this new browser window's URL field.
3. In this new window, click on padlock icon next to the URL line and select "New Circuit for this Site" from the menu popping up. Make sure that different exit nodes are active for the original download window and this new window.
4. Now please copy&paste the same TAILS address from the first window and request the page in the new window again via the different IP address of the new exit node. Then scroll down to "Verify using OpenPGP (optional)". Now save the digital signature of the download "Tails 3.x OpenPGP signature" and the Tails signing key in the folder that you used before to save the downloaded TAILS ISO image. It will take some time before you can verify because the gigabyte size download of the ISO image will take a long time to complete.  
*Note: It is safe to split the download of the triple consisting of key, signature, and payload files here in a different way than before. The main objective is splitting downloading the triple into two streams to two different IP addresses at all. We choose the different split here to avoid wasting time. We start the download of the huge file (the ISO image of TAILS) right away, and then let it finish in the background unattended while gathering the other two parts of the triple. Waiting for the huge download to complete will leave enough time for reading some info material presented on the TAILS web pages.*
5. You can now read the detailed instructions on how to verify the signature at the bottom of the page. Probably you can sit back, relax and smile a bit because by now, you have established your workflow for OpenPGP digital signature verification. You can verify on your own later after the download is complete. However, the detailed and explicit instructions show how carefully the publishers of TAILS write the instructions for users of their tool. The developers designed TAILS for intelligent IT laypeople.
6. Under "3 Continue installing or upgrading" on the page, first find out how the further installation proceeds on your system. It is, of course, advisable to keep a tab of the browser on this page so that you do not have to search for this page again later to follow the given instructions. If you had a quick overview of the process and the huge download is still in



progress, open a new browser tab and go to the TAILS homepage<sup>34</sup> there. Find information on the objectives of the TAILS developers there.

7. Until the huge background download of TAILS' ISO image finishes, remain on the TAILS pages and collect further information on the project. When the download is complete, discontinue studying the project information for now and return to the pages later. Soon verify the digital signature of the ISO image, i.e., verify the triple of ISO file, signature file, and the public key of the signer. You can only proceed if you have successfully verified. Otherwise, you should repeat the process. If the signature verification fails repeatedly and you have double-checked for errors on your side, contact the developers and inform them about the persistent error.

*Note: The developers of free security software are generally very interested in investigating genuine suspicions of sabotage or errors in the distribution of their tools.*

8. Now carry out the instructions for the installation described under "3 Continue installing or upgrading". You had a glimpse of them in advance in the second last item of this to-do list. I appreciate the quality of the descriptions on the website and regard them as commendable.
9. Create two TAILS sticks, each with a persistent storage domain. The presence of two TAILS sticks largely facilitates secure updates should a manual update become necessary and alleviates future troubleshooting.

*Note: As is customary with operating systems, we must carry out system updates sequentially. If you haven't used TAILS for a long time, you may have missed an update. In this case, you often cannot perform the next update automatically and will be referred to the manual procedure. We can facilitate the manual procedure if we downloaded the latest version of the new TAILS ISO image on a TAILS and authenticate the download by checking the digital signature. Then we insert the second TAILS stick and update its TAILS from the current system with the newly downloaded ISO image. Finally, we start the downloaded up-to-date TAILS of the second stick and update the first stick from there.*

10. Please start TAILS on your computer and familiarize yourself with the operating system. If TAILS does not start on power-on with the TAILS USB stick inserted, you must first configure the BIOS of your system such that the system can boot from USB media. Furthermore, please ensure that the boot order gives the USB ports priority over the built-in hard disk. You can find excellent troubleshooting instructions on the TAILS website.

*Note: Unfortunately, the computer vendors like to put annoying stickers onto the computers with a lot of unimportant pseudo information, for example, Intel inside. But on the other hand, they almost hide the crucially important information on how to enter the BIOS to adjust settings for the boot process. Usually, you have to hammer on some function key (often F2, F10, or F12) or even a combination of keys in a narrow time window after initializing the boot process. The magic keys are changed every so often when introducing new models. If necessary, research with your favorite search engine on how to enter the BIOS in your model.*

### 3.5.13 RECx13, browse the Darknet on a TAILS

*Difficulty: simple.*

*Time: at least 15 min.*

*Prerequisites: RECx2, RECx12.*

1. Use a Darknet search engine to find a Darknet forum of your choice for an innocuous legal gray area of your choice (music piracy, cannabis, or similar). Read a few contributions. Please do not post any comments yourself yet.

<sup>34</sup><https://tails.boum.org/>

2. Use a Darknet search engine to find a Darknet trade of your choice for an innocuous legal gray area of your choice (music piracy, counterfeit luxury goods, cannabis, or similar) and read some offers. Please definitely do not contact any trader yet.

#### 3.5.14 RECx14, setup access to your email account in TAILS

*Difficulty: simple.*

*Time: 5-10 min.*

*Prerequisites: RECx12.*

1. Jot down the access data you need for your "normal" email account.
2. Start TAILS.
3. Find out which standard email client comes with your TAILS.
4. Create access to your account in this client.
5. Write an email on your TAILS and send it to a buddy.
6. Receive an email response from your buddy on your TAILS.

You have successfully masked your geographical location during your access to the Internet.

#### 3.5.15 RECx15, read your favorite online news outlet using TAILS and TOR-Browser

*Difficulty: simple.*

*Time: from about 5 min.*

*Prerequisites: RECx12.*

1. Start your TAILS.
2. Open the TOR-Browser there.
3. Read your favorite online newspaper.

#### 3.5.16 RECx16, roundup evasion maneuver

*Difficulty: simple.*

*Time: from approx. 5 min.*

*Prerequisites: RECx12.*

1. Please envision the following situation: You are a dissident in a dictatorship and are currently working with your TAILS on a documentary on human rights violations by your regime. You intend to publish the documentation on your Darknet blog. Suddenly someone pounds vigorously against the door. Government officials demand access to your apartment and threaten to kick in the door. You shout that you are in the toilet and promise to open the door within 30 seconds.
2. Find the fastest way to put your computer into an innocuous state and to hide your TAILS stick. Research the options on the TAILS website for a swift shutdown of the TAILS operating system.
3. Try out different options. Measure the time needed for each option and jot down the recorded times.
4. Identify the fastest method.

Reminder: This is just an exercise! Don't get carried away! You do not yet need to swallow your mini-TAILS stick or flush it down the toilet.

### 3.5.17 RECx17, transfer your GnuPG key to your TAILS

*Difficulty: medium.*

*Time: from approx. 15 min.*

*Prerequisites: RSx6, RECx12.*

1. Start your standard computer using the plain operating system.
2. Find out how to export a GnuPG key pair. If you need to research on the Internet, do so using the TOR-Browser.
3. Export your standard key pair to a USB thumb drive.
4. Shut down your system and boot into your TAILS.
5. Connect the thumb drive to your TAILS and import the key pair into the GnuPG implementation of your TAILS.
6. Verify the availability of the key pair in your TAILS by sending an encrypted email from your TAILS to your default email account.
7. Receive and decrypt it on your plain system.

### 3.5.18 RECx18, install and use Academic Signature on your TAILS

*Difficulty: medium to hard.*

*Time: from approx. 45 min.*

*Prerequisites: RECx12 and the installation of Academic Signature on your normal system.*

Using your preferred search engine, inquire how to install and use Academic Signature from the persistent memory area of your TAILS thumb drive. Submit the search string "install Academic Signature on TAILS", for example. You will probably find the page given in the footnote<sup>35</sup>. The author of this book posted this web page.

Please follow the instructions on the page you found, and place the executable file and required subdirectories in a folder in the persistent area of TAILS. Because a reboot of TAILS would wipe an installation in the standard location for programs, you must place the source code in the persistent area. Compile it, leave the resulting executable binary there and call it, for example, using the File Browser.

Create an ECC private/public key pair with Academic Signature on your TAILS. Transfer the key pair to the Academic Signature installation of your plain operating system via a thumb drive. Now create some short text in TAILS and encrypt it using the public key generated there. Transmit the cipher to your plain operating system as an email attachment and decrypt the ciphertext there.

*Note: Some email providers may remove unclassifiable email attachments, block emails with such attachments completely, or even make unsolicited changes to the attachments. If your email provider does that to your emails, pack the ciphertext in an attached zip archive and try again. Providers commonly transport attached archives or other compressed files of common formats without touching them.*

<sup>35</sup>[https://academic-signature.org/Using\\_aca\\_sig\\_on\\_TAILS.html](https://academic-signature.org/Using_aca_sig_on_TAILS.html)

**3.5.19 RECx19, spot the MAC address of your device's WLAN interface**

*Difficulty: somewhere between easy and difficult, depends on the accessibility of your WLAN equipment.*

*Time: from about 20 minutes.*

*Prerequisites: control over a WLAN infrastructure.*

1. Find a way to read out the MAC addresses of your systems' WI-Fi interfaces. Whenever a device connects to a WLAN, the operator of the WLAN can find out the individually assigned permanent MAC address of the connecting device. The marketing mafia, as well as government agencies, surely possess databases that link this MAC address to your device and your name.
2. You can probably view the MAC addresses of connected devices via the maintenance interface of your home router. There is also free software, installable on your computer, that shows the MAC address of the computer's WLAN interface.
3. Make a list of MAC addresses of devices (smartphones, notebooks, tablets, printers, children's toys,...) going online via your home router.

*Note: There are many tools to show the MAC address directly on your system. But beware, I have found dysfunctional MAC address spoofing tools that only change what the tools display themselves. Contrary to the displayed, seemingly spoofed MAC address, the computer often instead transmitted the true MAC address to the WLAN router.*

**3.5.20 RECx20, find free software to spoof your device's MAC address**

*Difficulty: simple to medium.*

*Time: from about 10 minutes.*

*Prerequisites: none.*

1. Find free software that allows you to obfuscate the inscribed MAC address of your interface and to substitute it temporarily with another MAC address.
2. Use one or more of the programs you found and connect to your home router with hopefully actually obfuscated MAC address on the WLAN.
3. Now check if the individual, real MAC address is indeed held back and replaced by the newly selected, spoofed address and that this replacement remains stable for a long time.

*Note: At the time of writing this section, I was unable to find a reliable tool for this task except TAILS.*

**3.5.21 RECx21, spoof your device's MAC address using TAILS**

*Difficulty: easy to medium.*

*Time: from about 10 min.*

*prerequisite: RECx12.*

1. Boot into your TAILS in the default settings and check which MAC address your TAILS uses to connect to your home router.
2. Find out how to turn MAC Address Spoofing on or off when booting TAILS. You will quickly find out. Dear reader, such searches are not wasted time, so I do not give the result here not to spare you the search!
3. Boot TAILS with MAC Address spoofing turned off. Check again which MAC address your TAILS uses to connect this time.

### 3.6 Attacks on receptive anonymity

Today, TOR is the most common and powerful anonymization tool. Thus, in this section, I will only describe attacks against TOR. As far as I know, there are very few academic publications about attacks on other anonymization tools. I will distinguish three types of attacks on TOR-made anonymity. (Literature usually categorizes in different ways.)

#### **Type 1 attacks**

Attacks from the immediate web environment of the person seeking anonymity comprise the first category. Your ISP might be interested in the endpoints of your TORed data traffic, for example, to generate advertising revenue. But perhaps your government has also asked it to tell them what you are up to when you surf the Internet via TOR.

If you are lucky enough, your employer will not block TOR access at the workplace. Still, your employer or a curious IT administrator in your company may be interested in your anonymized activities on the web. Or perhaps you live in a police and snooper state, and a modern analog of the former Nazi Blockwart wants to be able to transmit information about your surfing habits to his case officer at the secret police.

Even research groups with limited resources can reproduce and investigate such first category attacks. That is why there is a wealth of information about them. The TOR developers can analyze the vulnerabilities and harden TOR against such known first category attacks after the respective research groups published them.

#### **Type 2 attacks**

In the second category, I pigeonhole attacks from the web environment of the target servers of your TOR traffic (possibly with cooperation of the direct web environment of the person seeking anonymity). Attackers can launch these, of course, from anywhere in the world. Such attacks are also easily re-enactable with limited means. However, we should think that, for example, an Australian or Russian intelligence service should hardly be interested in the political attitude we have as individual American, British or German citizens. They would have little interest in what diseases we search for on the web or what media we consume anonymously on the Internet. If we work in a place of interest for the visited server's home state, such attacks may serve to collect extortion material against us. Besides, there may be purely financially motivated quests for extortion material by ordinary criminals.

#### **Type 3 attacks**

The third category of attacks is the most dangerous one for democratic societies, but at the same time, unfortunately, is hardly subject to any public investigation. These attacks can only be carried out by the conglomerate of IT companies and government services (the Data Kraken), as it has enormous technical resources and virtually unlimited access to the network infrastructure of its political domain. Such attackers have the mass data of their political domain at their disposition, possibly also the cooperation of the target server and its immediate web environment. They can, therefore, attack the anonymization of their political domain's citizens simultaneously, en masse, and in parallel using powerful big data techniques. Security researchers with limited power and budgets do not have these options. Only theoretical considerations can help us identify weak points facilitating such attacks without the cooperation of the Data Kraken, and only theory can lead us to countermeasures against those attacks.

The developers and maintainers of current anonymization networks concede that the networks may not be secure against anonymization attacks by an adversary with global access to the infrastructure. We do not know to what extent government services are already capable of practically pinpointing the two endpoints of anonymized data streams that their access domain encompasses completely.

Now, sorted by the three categories introduced above, I'll describe some attack possibilities. In doing so, I will also point out possible countermeasures that impede the attacks.

### 3.6.1 Attack type 1, website fingerprinting

This attack belongs to the first category and would fit into the following scenario: You are a dissident in Despotistan and would like to inquire about international law on the website of a civil rights organization. The dictator in Despotistan does not like this and would throw you into jail if you could be proven to be doing such research. Because the members of the despot's family want to be able to move anonymously on the Internet at any time, access to TOR in Despotistan is not blocked.

Because you are on the political watch list, authorities closely monitor your outbound and incoming traffic. The despotic Ministry of the Interior compiles a list containing characteristics of traffic through the TOR-Browser with 1000 different disliked web pages. These include Amnesty International, the Doctors Without Borders (msf), the web interface of an email provider which allows anonymously created accounts and operation via TOR, Wiki Leaks, etc. etc. Each of these websites has unique timing and number of TOR data packets sent to its visitors. We can view these characteristics like a fingerprint of the website visit. The attackers compare the pattern of the TOR stream you receive with these fingerprints. If the traffic pattern matches, they have unmasked you as a visitor to the Wiki Leaks site, for example. Under Waterboarding, you confess to visiting all other 1000 prohibited sites several times a day, and you and your family members will be sentenced to 1000 years of labor camp or get a 23 fold life sentence.

We call this simple passive attack website fingerprinting. Because developers of TOR have already effectively limited the amount of information that attackers can draw from TOR traffic, this attack currently has a practical upper limit of resolving perhaps a few thousand target websites. Thus the despotic authorities may unmask the visit of about a thousand banned websites via TOR by this attack. Despotistan's citizens may achieve a certain amount of protection by simultaneously downloading the boring news outlet of the despotic government in another window of the TOR-Browser, or by playing a government propaganda video there during the TOR query of the banned website. In this way, you can shuffle the TOR packets yourself, hampering the attack.

You can find an investigation into this attack from the TOR developers here<sup>36</sup>.

### 3.6.2 Attack type 1, modulating the data rate with nearby end point

User A wants to access the blog of the human rights organization B, whose seat is in Despotistan, as is the residence of A. The despotic Ministry of the Interior wants to ban human rights organization B and to let all sympathizers "disappear". They want to do this as thoroughly as possible. That is why the Despotic Ministry of the Interior intends to draw up a list of all citizens who have accessed B's blog.

Naive citizens who have done so openly are, of course, immediately on the list. The Despotic Ministry of the Interior instructs the Agency of State Security (ASS) to nonymize the visitors protected by TOR. They want to add the less naive citizens to the list as well.

The ASS complains that it cannot follow the data path via TOR because the nodes of the anonymization network are mostly outside its access range. However, because both A and B are in the despotic sphere of influence, the attack described below will be promising: The ASS will modulate all data streams supplied by the web server from B to TOR. For example, they could delay one in three outgoing data packets in the first despotic router (waltz-time modulation). At the same time, the ASS analyzes the time interval of the data packets received from TOR for all TOR users in Despotistan. Users where they can find traces of the characteristic waltz-time modulation

<sup>36</sup><https://blog.torproject.org/critique-website-traffic-fingerprinting-attacks>

are blacklisted and will be arrested by ASS at due time. There are many ways for the attackers to mark the TOR data stream they have access to at the source. In addition to the rate modulation, they could also, for example, cut out every thousandth data packet and detect characteristic reactions of the browser on the receiver side. This type of attack is referred to as a "Confirmation Attack" in the literature.

Both the entry and exit nodes of the TOR network cooperate with the attacker in the worst case for the TOR user. But the simultaneous cooperation of the network environment (provider, first router, etc.) of A and B with the attacker would suffice. If A would simultaneously receive an obfuscating TOR stream from another source, this would not hamper the attack as hard as it does with website fingerprinting. The TOR user could attenuate traces of stream modulation or marking with this technique but could hardly completely blur them when the attack is well done. A higher, somewhat fluctuating latency of the anonymization network would also help.

The TOR user could best counteract the marking of the data stream by rate modulation with intended network congestion outside the despotic sphere of influence. For this purpose, for example, the user could access his TOR input node via a VPN that is slow compared to TOR and lies in a political domain that is not an ally of Despotistan. However, users would need additional countermeasures against having the attacker mark individual data packets in other ways. Unfortunately, as far as I know, no simple procedures or free tools are available to counteract such a TOR packet or stream marking in general at the moment. All the countermeasures mentioned so far would lessen user comfort and would not completely solve the problem of the confirmation attack. However, in the present case, from Despotistan, they might still be able to make the difference between life and death for vulnerable dissidents.

### 3.6.3 Attack type 1, physical attack on known TOR users

This attack is primitive. Concealing using TOR is very difficult. Thus it is sound to assume that the snooper state's intelligence agencies know all TOR users located in their territory and cataloged their names in secret blacklists. I presume that this is also the case for my home country, the Federal Republic of Germany.

An unidentified source leaked a document that cast a spotlight on the inner mindset of a western intelligence service, shortly after Snowden came forward with his exposures in 2013. The text revealed that the NSA labeled citizens who had only googled for TOR or TAILS or read a particular Linux magazine as extremists in an internal database. (We cursorily touched that point already in 3.5.3.) A search with your favorite search engine (extremist Linux magazine NSA search TAILS TOR) will provide many references to the incident.

If a nation-state intelligence service considers a domestic TOR user a worthwhile target, it might send a specialized burglary force to his office. Conventional analog locks are not an obstacle. The mission might be to copy all sorts of paper documents and the content of the computers' hard disks, plant bugging devices, spy cameras, and install spyware like, for example, a keylogger on the target's computers. Private communication is no longer possible from such an office. We private citizens have little way to prevent this at most perhaps, we may notice it.

Maintaining burglary teams, however, is a considerable effort for the intelligence services of our western democracies. The burglars are expensive to maintain, and the risk of bad press is substantial should their actions be uncovered. Thus we can expect that in democratic states, such burglary squads would only exist in relatively small numbers. Such that state-organized burglary squads should rarely pose a danger to the unruly citizen in an environment such as the USA, the Federal Republic of Germany, or other countries of the European Union. In obvious dictatorships, however, such burglary squads can be as common as they were in former East Germany.

*Note: You won't believe it, in former East Germany, they went down so far as to repeatedly flatten tires of bicycles belonging to older Ladies of undesired political opinion. In the German*

*language, they officially called this "Zersetzung" (disintegration) of the political opposition. Don't be too sure we will never see such state agent behavior (again) in our countries. It would be wise to expect all kinds of outlandish action from intelligence services enjoying unlimited funding and lacking adult supervision.*

#### 3.6.4 Attack type 2, NSA attack, hacking the TOR-Browser

From the Snowden revelations, we know that the U.S. intelligence agency NSA also attacked TOR users from the side of the website the users intended to visit. Initially, the intelligence service intercepts website requests from users' TOR-Browsers. The NSA crookedly responded from maliciously configured, strategically placed servers (code name QUANTUM) that could react faster than the legitimate servers. In IT security talk, we call this exploiting a race condition. On the web, the user's Browser processes the earliest seemingly proper response to a request. It discards responses arriving later. The poisonous NSA response exploited a then-existing vulnerability in the Firefox browser, on which the TOR-Browser is based, to pwn the user's system (code name FOXACID). The program is code-named EGOTISTICALGIRAFFE. You can find a link to this attack here<sup>37</sup>.

Ordinary criminals could, in principle, also carry out similar attacks trying to satisfy a financial goal. However, without the NSA's billion-dollar budget, they will hardly be able to buy the expensive but necessary Zero-Day Exploits. A Zero-Day Exploit is an exploit based on a bug in a software that is usable for pwning the attacked computer and is not currently known to the public, to any trusted institution, or to the software publisher itself and against which there cannot be any protection yet.

#### 3.6.5 Attack type 2, MITM attack at an exit node

An alert TOR user reported that the operator of a TOR exit node launched a so-called man-in-the-middle attack (MITM attack) on his TOR data traffic<sup>38</sup>. The user administered an HTTPS-protected website, accessed his website via TOR, and compared the presented site certificate with what he knew was his site's. Much to his surprise, in a TOR exit node hosted by Deutsche Telekom, the website certificate of the user's website had been exchanged for a different but accepted one of an attacker controlling the TOR exit node. Only the webmaster of the legitimate website, knowing the original certificate's correct public key, can detect such a fraud. The attacker's system thus successfully impersonated the target website for communicating TOR users. Note that the crook must have either hacked a certification authority to get the fraudulent certificate, or else he must have been a government crook. The attacker, the man-in-the-middle (MITM), was then able to decrypt and possibly manipulate the out- or ingoing traffic from/to the requesting computer, then re-encrypt outgoing traffic for the real web page and forward it to the web page. In principle, the attacker controlling the TOR exit node could also have redirected the user's request to a maliciously prepared, fake target website in an attempt to hijack the requesting user computer. However, the TOR Browser is quite resilient in the most stringent security settings.

The attacked user (and webmaster) saw himself unable to make a reliable statement about the factual purpose of the attack and the identity of the attacker. State espionage services can circulate counterfeit certificates past the standard security procedures and thus falsely impersonate websites. The German intelligence service BND, for example, has cooperated with the agencies of the Five Eyes to nonymize TOR users. So it would be a safe bet to suspect the BND of launching this attack. You can find an article in the German language on the cooperation of the German intelligence service with Five Eyes services to unmask TOR users here<sup>39</sup>.

<sup>37</sup>[https://www.eff.org/files/2014/04/09/20131004-guard-egotistical\\_giraffe.pdf](https://www.eff.org/files/2014/04/09/20131004-guard-egotistical_giraffe.pdf)

<sup>38</sup>see:<http://www.teamfurry.com/wordpress/2007/11/20/tor-exit-node-doing-mitm-attacks>

<sup>39</sup><https://netzpolitik.org/2017/geheime-dokumente-der-bnd-hat-das-anonymisierungs-netzwerk>



### 3.6.6 Attack type 3, big data attack on TOR users by statistical analysis

Each Internet user has his habits. For example, you'll often access a favorite site that I might never visit and vice versa. A target's visits to a favorite site are particularly promising markers for attackers if other people rarely visit that site. Let's assume, for example, a user frequently accesses the exotic hypothetical gardening website [http://www.hydrangea\\_and\\_hornets.html](http://www.hydrangea_and_hornets.html). This site may have two or three visitors per day. Then, the user's anonymity when using TOR is at risk.

The attacker may know from traffic observation that one or more TOR users are accessing the hydrangea site. The user has likely visited this website before, without any protection by TOR. The attacker now only needs to filter out the 500 or so unprotected IP addresses that have ever accessed the hydrangea site and then intersect this set with the set of all known TOR users. In the example given, there is probably exactly one user left.

Now let's assume there were 400 users in the intersection because the hydrangea page would be as popular as the online magazine Wired, for example. The attacker would have to put in a tiny bit more of an effort and see which of the 400 users is currently communicating via TOR if there is TOR access to the hydrangea&hornet page, now assumed to be quite frequently visited. Still a piece of cake.

Gradually, this gets more and more difficult when choosing an ever more requested website. Maybe the site is now the search engine Bing and 10 000 TOR users have also accessed Bing recently via their unprotected IP address. The attacker might exploit the time patterns of all suspects' access to Bing and still unmask the user with a little more effort. There will be a reasonable number of citizens who have already used TOR and Bing and who are currently maintaining traffic in the TOR network when someone starts a Bing search via TOR at this very moment. Perhaps the attacker now needs to evaluate several more such events and include thematic characteristics of the Bing search for its analysis. This modern form of dragnet search in different variants is very suitable for a service having access to mass data.

A countermeasure under high persecution pressure could be to train different behavior patterns for unprotected Internet use and anonymous surfing. You might even learn several distinct roles with individual habits and interests for the anonymized use of the Internet. Besides, it could help to cover up the use of TOR through an upstream VPN.

### 3.6.7 Attack type 3, big data attack by modulating data rate

A powerful global adversary can monitor, control, and modulate the rate of all Internet traffic, including all relevant TOR traffic. Such an attacker can easily recognize the modulations in all sections of the TOR data path it can access. Subsequently, it can uncover the complete TOR path in unfavorable circumstances. The TOR path should cross the boundaries of political domains and partially also run outside the reach of one nation's intelligence agencies to achieve enhanced security. If possible, TOR traffic should have a bottleneck outside of the potentially most dangerous nation-state attackers' political domain. Then such modulation would be somewhat blurred.

### 3.6.8 Attack type 3, trojan nodes

The two attacks described before are easier when the attacker can not only control the network infrastructure but even cooperates with one or more nodes' administrators. A significantly smaller amount of data and less specific data will then suffice. The assumption is plausible that nation-state intelligence services operate numerous TOR nodes to make it easier to anonymize TOR users. In the literature, the cost-effective, covert flooding of the infrastructure of an anonymization network by hostile servers of the attacker is called a Sybil Attack.

The collaboration of TOR nodes' administrators with the attacker opens up many attack

---

possibilities and, for example, facilitates the manipulation of the data rate and detecting the modulation or another marking of the data packets of a particular TOR data stream.



The Emperor  
is nude!



## 4. Expressive Anonymity

Receptive anonymity, treated in the last section, corresponds to real-world observation, viewing, or listening in without being noticed. In the real world, analog protection could be, for example, concealing what you are looking at by wearing dark sunglasses or driving in a car with dark tinted windows. We can easily understand that, and it is simple to achieve. Our digital equivalent of dark sunglasses was mainly the TOR-Browser.

We will now drop the restriction to purely receptive anonymous Internet use. Let's turn from a lone wolf into a social being in the digital realm. We want to speak our mind on the Internet without being caught and exposed by the state watchdog. The corresponding action in the analog real-world could be painting graffiti on the concrete surfaces in your town. It could be painting a political statement against the dictator onto the walls of the secret service building or, most commonly, to get together and exchange information or talk about politics in secret in the basement of a friend's house. We have higher security challenges now. The TOR-Browser will no longer be sufficient as our primary tool.

### 4.1 Fictive identities

This level of the pyramid of dissidents' protection is about expressive anonymity. That type of anonymity often requires the establishment of a fictive identity on the Internet that no surveillant can attribute to your person. You send messages from such fictive identities to communication partners, and you use these fictive identities to receive your partner's responses. The expressive anonymity of a fictive identity is more vulnerable than anonymity limited to read-only Internet access.

The fictive identity can be the owner of a chat or email account. Or it could even be the author of a website in the Darknet. It is usually associated with login processes that lead to actions an attacker can predict. That always reveals some information to the attacker. The operator of the used chat or email service, at least, knows precisely when this identity is active and can differentiate the fictive identity's data traffic regarding the service from all other TOR data traffic to and from its servers.

Besides, unencrypted utterances via the initially anonymous email or chat accounts allow

the services' operators to analyze the language style and content, allowing it to unmask the fictive identity or at least determine a characteristic fingerprint of the user's language. A cautious user should always assume that the operators of the services used anonymously cooperate with government agencies and kindly support their access to all available data on the fictive identity.

Other literature often refers to such accounts of fictive identities as pseudonymous accounts rather than anonymous accounts. I avoid the term "pseudonymous account" without additional attributes in this book. An account is already pseudonymous, according to common language usage, if you have given, for example, Winnie the Pooh or Sir Lancelot as owner name without systematically concealing your IP address. Of course, a surveillant could easily attribute such a pseudonymous account to your real person. The term pseudonymous account is vaguely defined, in my opinion. The language used in this book is following German juridical use (§3 Abs.6 and 6a BDSG, the old version before May 2018). In this book, I refer to the account of a fictive identity, which the user robustly decoupled from his true identity as a pseudonymous account with secondary anonymity. You can decouple this account from your true identity only if you diligently anonymize your traffic when creating and using the account.

On the other hand, I use the term primary anonymity when an attacker cannot assign the corresponding elements of data traffic to a fictive identity at all. This occurs when the attacker cannot distinguish the fictive identities' anonymized traffic in any way from all other anonymized data traffic. With TOR anonymity, this would apply to traffic organized as rendezvous type and to purely receptive activity.

## 4.2 Hazards of payments

As a private individual, you are usually dependent on the help of the service provider when creating an anonymous account for a fictive identity. The service provider must allow you to access storage space and fetch computing time on its systems without being able to obtain information about your true identity, consumer behavior, or communication content.

That is not in the interest of a commercial service provider who does not charge any fees. Exploiting and selling your data is the business model of such providers. Hence, you will only be able to anonymously use commercial service providers as long as they do not verify being able to assign your true identity to the account. Some providers, however, won't do that to save money and effort. You can use those for hosting your fictive identities' accounts.

For-profit companies may present as a business model to allow or provide anonymous access for you for a fee. Many commercial VPN providers do that. Even if these companies indeed put the security concerns of their customers above the interests of the government of their registered seat (dream on), the companies would lay a clear data trail to your real identity through the payment process.

You cannot take your anonymization into your own hands, for example, by purchasing server capacity, which you then configure to promote anonymity. Snoopers would be able to link these servers to you via the payment flow, even if your business partner would not cooperate with them. However, you could support anonymization for the public by providing the appropriate infrastructure to a large community of (hopefully) like-minded people. That is the basis of the TOR network. I think this is very commendable, but it would still hardly bring any progress for your anonymity.

Let me come back to the data traces of payment transactions. I have a morally motivated interest in helping dissidents, environmentalists, human rights activists, critics of globalization, fur-animal liberators, etc., and intellectually capable democratic politicians in their quest for secure communication. This motivation made me write this book. In my opinion, commercial interests are currently by far the greatest threat to freedom and privacy on the Internet. I dislike the sleaze of greed, cryptocurrencies, big money, aggressive marketing, and the web.

Nevertheless, I acknowledge that it is difficult and fascinating to make payment flows more secure and anonymous by technical means. However, the word Bitcoin is used only in two places in this book, namely here and later in one section on attacking password encryption. I have no interest in making it easier for drug dealers, tax evaders, mafia cartels, religious extremists, dubious financial institutions, or arms dealers to train the next generation in money laundering. Hence, I will not discuss the anonymization of payment flows here. In my opinion, there are enough opportunities for intelligent citizens in our present-day Western economy to secure their livelihood in a legally and morally clean way. Securing this economic livelihood for fellow citizens who, for whatever reason, are less financially successful is a task for society. I am afraid that I can contribute to that task with this guide at most only indirectly.

Let me finish this section with a piece of advice. Never trust paid services, take advantage of surveillance gaps in commercial but free services, and, wherever possible, use morally motivated providers who deliberately support free, anonymous web use.

### 4.3 Tools for expressive anonymity

Now we will deal with free and open-source tools for various forms of anonymous communication. We will do this in descending order regarding the size of the attack surface of the communication modes. First, we will look at anonymized chat, then at anonymized email traffic. At last, we will look at the safest communication mode, which is anonymized direct file transfer.

#### 4.3.1 Chat accounts

Communication via chat services is modern. I mainly see two problems with the security of chats:

1. The user is led to the loss of control by commercial providers.
2. She/he typically logs in to the chat account for long periods and, thus, offers a long-lasting attack surface for nonymization attempts.

##### **To the first problem:**

Most people, unfortunately, select a chat service along these lines:

- Which chat service do my friends use?
- Does this chat service provide explanations for why I can trust it?
- If so, where do I have to click so that the app of this chat service installs on my system, prepares my system for providing access, scans my contacts and everything else on my system, sends the data "home" and will work automatically from now on?

If I would honestly comment and evaluate this widespread practice, I would sound rude. Of course, as a security-conscious user, you have to stay in control. The security-conscious user chooses a server that allows him to anonymously set up a chat account via a web interface of the service in the TOR-Browser. It would be reasonable for a German user to choose a server located in Russia, Hong Kong, or, at least, abroad.

The user chooses a suitable chat client for his computer, with which the conditions for the chat data traffic between client and server can be set and installs this client. (The client is the local program on the user's computer, which we run to carry out the chats.)

Later on, we will add an encryption module for the chats. But this step is not covered in this section yet.

##### **To the second problem:**

Long login times are typical for a chat. Also, all kinds of further information are usually forwarded to the chat server and the buddies as well: The user is typing, the user has not touched the keyboard for two minutes, etc... The chat account is a gossipy beast. That's why we should consider

anonymous communication via chats as perhaps the most comfortable but also the most vulnerable form of anonymous communication.

I am well aware that numerous experts recommend encrypted chat for secure communication to the community of Internet users. Some even recommend commercial chat services bound to central servers and bash generally applicable encryption tools like, for example, the widely used GnuPG. That worries me. Anyhow, we'll make it possible to securely and anonymously chat using free and open-source tools and decentralized infrastructure.

#### 4.3.2 Jabber chat accounts

There have been chat services from large and formerly large corporations for a long time, each of which had established its own format and were incompatible with each other. Early 1998 Jeremie Miller started Jabber as an open-source project. He defined a free data exchange format and developed the first implementation of it. Later Jabber was renamed XMPP. As of today, the XMPP Standards Foundation maintains the XMPP standard. The name Jabber is still in use unofficially.

With open-source and free software, we can use the XMPP chat as independently as email. We use decentralized servers that communicate with chat users and with each other in a protected manner. These servers can also run XMPP chats between accounts on different servers, of course.

Many volunteers run such Jabber servers where anyone can anonymously create a free chat account. Using your preferred search engine, you will readily find a free server that suits you with the search string "List of free public XMPP servers". The availability of these amateur servers is not as high as that of professional servers. You should have two or three active Jabber accounts on different servers so you can choose an alternative anonymous carrier when a server happens to be down for whatever reason. Besides, a certain amount of rotation of activity also increases the security against nonymization attacks.

#### 4.3.3 Jabber client Pidgin and TOR

There are many suitable Jabber clients. You do not need to use Pidgin. However, in the exercise part, I will assume you use Pidgin. Your search engine will readily offer you many alternatives when employed with the search string "free Jabber XMPP clients". But please make sure that your client is open-source and free like Pidgin, that you can install an OTR plugin for end-to-end encryption (for later use), and that routing traffic through TOR is supported. Please use a client that is free of charge. Being stingy is not the main reason. Publishers of non-free paid programs will generally control, regulate, and personalize the release of their software. Using such software breaks your anonymity. It would allow them, for example, to provide you with a backdoored version if your government doesn't like you.

Windows users can find Pidgin here<sup>1</sup>, Linux users can find Pidgin in the respective repositories of their distribution. Mark Spencer originally developed the program and named it Gaim. It has been around for a long time and was renamed Pidgin in 2007. It is actively maintained and developed and can be augmented using various plugins.

Anonymized Jabber chats via Pidgin use TOR as an anonymization buffer. TOR merely serves to decouple the user's IP address from the chat (or an email) account. The usage of the chat account itself is conventional. Thus we classify the anonymity of this communication as secondary anonymity.

---

<sup>1</sup><https://www.pidgin.im/>



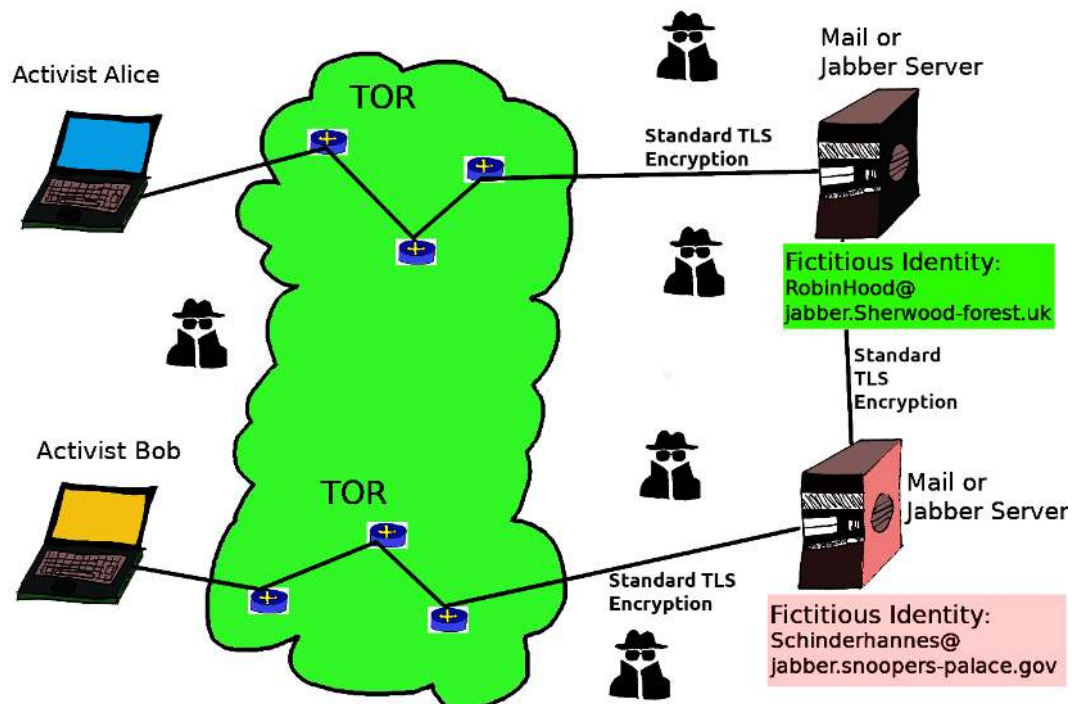


Figure 4.1: The figure illustrates the use of TOR as an anonymization buffer. The personally identifying IP addresses of users Bob and Alice are decoupled from the fictive identities "RobinHood" and "Schinderhannes". Apart from the anonymization buffer, this is normal chat or email traffic. The pictures of the gentlemen in hats and sunglasses stand for government surveillance. In this book, we call anonymity generated in this way secondary anonymity.

#### 4.3.4 Rendezvous type TOR routing

We can find a superior pattern avoiding the conventional servers. The TOR network itself may establish the connection between the two halves of the TOR path, starting and ending at the users' systems. We call this a TOR path of rendezvous type. The respective user computers take the role of the conventional server and provide a so-called Onion Service.

The TOR web pages<sup>2</sup> excellently explain this way of contacting the partner, even for clever IT laypeople. In this case of primary anonymity, the attack surface is much smaller than in the case of secondary anonymity discussed so far. However, we cannot store data at external resources, so synchronous TOR online time of communication partners is required.

Finally, I would like to point out that there is already the possibility to chat with free and open-source clients directly via TOR in the rendezvous type in a serverless fashion. One or both partners open an Onion Service, similar to a Darknet website. The Onion Service's TOR path connects to the receiver's TOR path via a TOR node acting as the rendezvous point. The respective TOR input nodes communicate directly through the TOR network and do not need conventional servers in the nymous net as intermediates. In principle, this offers increased security compared to server-bound operation. The problem of long, synchronous online times at the endpoints remains. Although these are not directly server login times, the typical security risk of concurrent activity time is still present, albeit somewhat mitigated. Fortunately, we don't have a server here that would centrally store activity time. But we still have an extended concurrent TOR online time. In this period, the users are accessible via a fixed TOR address. During this time, an attacker could launch a nonymization attack.

<sup>2</sup><https://www.torproject.org/docs/onion-services>



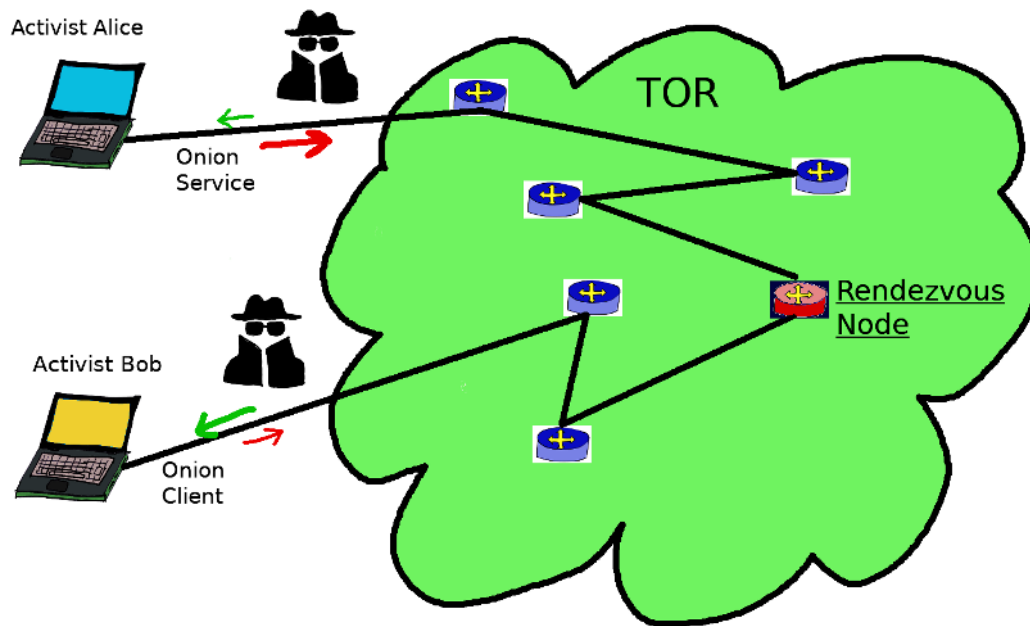


Figure 4.2: The figure depicts direct one on one contact via TOR in rendezvous type. We don't use any resources outside the anonymizing network for our communication. Sender and receiver connect directly via the TOR network. In this document, we refer to anonymity generated in this way as primary anonymity. The images of the shady characters with hats and sunglasses again symbolize government surveillance.

An innovative tool called Ricochet-im<sup>3</sup> allows such serverless chats. Ricochet is operational. You can chat with it, i.e., exchange short text messages. Ricochet works alternately as Onion Service when sending and as Onion Client when receiving. But there are still some problems with Ricochet: Ricochet uses the end-to-end encryption of TOR(v2) without bringing along independent crypto functions. Unfortunately, TOR(v2) cryptography is currently weak in places (1024 bit RSA, a hash function SHA1 cut to half bit length for authentication of the Onion Service. Cryptologists have been considering both to be insecure for several years). When I looked at it last time in spring 2020, Ricochet could still transmit text messages only. Ricochet was not yet able to transfer files then. It is, therefore, hardly practical to manually insert potent end-to-end encryption in Ricochet Instant Messaging. But this would be necessary as long as the own TOR(v2) encryption is weak at some points. So we can say that this approach is promising, but the implementation is practically still immature. Ricochet should be kept in mind, though; maybe Ricochet (or another tool adhering to the same principle) is already safely usable for file transfer when you will read this text. At the time of writing this book, this is not yet the case. Regrettably, developers have not updated or extended Ricochet for about three years by now.

#### 4.3.5 Anonymous email account, using a web interface and the TOR-Browser

Email communication can provide a more robust level of anonymity than anonymized chat. In anonymous chatting, the participants' gravest anonymity weakness is the long synchronous account login time. We know new chat protocols<sup>4</sup> that store chat messages on the servers. Thus they can do without simultaneous login of the participants to their respective accounts. Yet, I do not want to discuss this smooth transition from the chat towards the email communication in this book. Instead, in this section, we will deal with the pure and asynchronous communication pattern of

<sup>3</sup>see: <https://ricochet.im/>

<sup>4</sup>see, e.g., OMEMO, <https://en.wikipedia.org/wiki/OMEMO>

email communication.

Why is anonymity more secure for email communication than for chat? The contact of the individual user with the email provider's server who collects or sends user messages can be limited to rare, short, and intensive data transfer bursts. The login times are brief, and the contact with the communication partner via the email provider is asynchronous. For particularly sensitive communication, the email user can, at a time of his choice, withdraw to a specially secured system that he does not use for everyday work (e.g., a TAILS) and, if necessary, also to a physically protected private place.

However, the opponents of anonymous and private communication are also aware of this. We live in a network environment that, over the years, has made the anonymous creation of email accounts difficult. Honesty mandates admitting that email stalkers, mobbers, phishers, and spammers have, unfortunately, provided surveillance advocates with plenty of excuses for establishing anonymity barriers. It is, therefore, hard nowadays to create and operate a free email account anonymously. (Paid offers can hardly be anonymous.)

The providers of free email accounts generate profits by selling user data and placing advertisements. They have no economic interest in the anonymous use of their services. Most providers bluntly refuse to register an email account when users request it through a TOR exit node. But the subsequent operation via TOR is allowed by some providers. Creating the mail account, for example, via a Vietnamese VPN, could help in some cases. A few providers do not allow access via TOR but via free VPNs, others vice versa. Some providers initially allow to set up the account via TOR but restrict or block access after a few days when they cannot obtain personally-identifying information. I have ample experience in this tough fight to maintain anonymity against email providers. The situation is confusing and, in some ways, chaotic. At least this mayhem indicates that it is not the government that strictly mandates these unpleasant blockades.

You may have an anonymously registered email account with a large provider that you can use via TOR without the occasional captcha, security check, or inbox deletion harassment. Then you can assume that your anonymity, which may have existed at some point, has been successfully broken in the meantime, or at least the provider believes it has.

The only large email provider I currently know of that allows harassment-free anonymous operation of the web email interface here and now (spring 2020) without client-side scripts, and which tolerates the creation and usage of the account via TOR, is an Israeli provider<sup>5</sup>. The overall situation is changing and could be different tomorrow.

The alternative anonymization network I2P offers a valuable opportunity to set up an email account (SusiMail) that connects to the standard email world and is not confined to I2P. It works quite well except for an annoying, easily circumventable error in handling email attachments. However, the receiving partner should have his account with a somewhat relaxed email provider. With some mainstream email providers (gmx, for example), my emails from the I2P network occasionally did not reach the inbox of my mainstream recipient account when they carried encrypted email attachments. They disappeared without a trace, without any error message, lost in the digital abyss of the Internet. It is unlikely that the problem arose on the outbox side of I2P.

We could try to trick standard providers with non-standard procedures (via Asian amateur VPNs, in a free WLAN with sunglasses, hat, and MAC address spoofing, with email account exchange networks, or whatever else). Even when this may succeed, using a free service in a way that is contrary to the provider's will should ultimately not be considered fair. I would, therefore, advise against it outside of genuine self-defense cases.

Now we will discuss the process of creating an anonymous email account employing the TOR-Browser.

1. Invent a complete alias identity with a fictive name and nickname, an address that is as real

---

<sup>5</sup><https://www.Safe-mail.net/>

as possible, a fictive phone number, age, gender, hobby, name of a pet, etc., all fictive. Make a mental picture of this person. Write down details on a piece of paper, a kind of profile of this fictive person, and also create a digital version of this profile as a text file.

The data should bear no reference to your real person. When you live in Hamburg, for example, you could perhaps enter Constance as your place of residence, search for a pastry shop there on the Internet - of course using the TOR-Browser with blocked scripts -, and select the address of the neighboring house. Also, choose and jot down an access password for the account. You will often fail later with attempts to create an account via TOR with different providers. Exploit those failed tries to internalize the fictive data more and more. It is very convenient to be able to access the same fictive data again and again during such attempts. Keep both the paper and the digital version in a safe place.

2. Find a free email provider that allows you to create a new email account via the TOR-Browser, ideally for blocked scripts. Set up an email account for this fictive identity. (At the time of this writing, e.g., with <https://www.Safe-mail.net/>.)

If you cannot find such a provider, the provider should at least allow operation via TOR. In the latter case, you may be able to set up the account via an exotic VPN of a country that probably does not cooperate with your state's intelligence services. (Here and today, this worked at <https://yandex.com/>, <https://www.gmx.de/>, etc....) When creating the account, do not give out any information that might link to you or your circle of friends or acquaintances, no real phone numbers, no nonymous email addresses of you or your circle of acquaintances. These are all taboo. You can generously supply the fictive data, on the other hand.

3. Create a second fictive communication partner by repeating steps 1 and 2, preferably with another email provider. Even if you know other like-minded people with whom you want to communicate anonymously, you need the second fictive identity. You can use it to check the communication possibilities on your own, without having to strain your partners' patience for such tests.
4. Use both email accounts exclusively via the web interfaces in the TOR-Browser with blocked scripts. Let the fictive identities exchange emails. As long as you do not encrypt the contents, include only short messages that are meaningless to external observers ("the kingfisher has landed.", for example). It must not be possible to fingerprint the language style. Never ever exchange emails with nonymous accounts of acquaintances.

The communication bubbles of the alias identities and the nonymous, real identities must never touch. If this happens accidentally, and it will happen at some point in time, this will burn the anonymity of the whole communication circle. The Home Secretary's valiant gang would have exposed you if they were good at their jobs, in Despotistan with grave consequences for those who were unmasked. But even in our western democracies, it would require a tiresome amount of work to rebuild the conspiratorial network against the vassals of the Home Secretary.

#### 4.3.6 Anonymous email account, using shared access to minimize the attack surface

Since the beginning of this course, we implicitly assumed the following trust model: All members of the communicating group trust each other completely. They are friends. In small circles threatened by powerful, hostile opponents, this is plausible. But the friends distrust government agencies and profit-oriented organizations. However, they rely on that government and profit-oriented agencies from different political domains do not cooperate or cooperate only to a tightly limited extent. They arrange interaction with non-profit organizations (TOR, for example) so they can keep the trust placed in them to the necessary minimum.

With this trust structure, we can further reduce the risk for attack, compared to the last section's communication pattern using anonymous email accounts. In addition to the email service, many

email providers also offer storage space with their mail service. When this applies, members of the group can use the storage space in a dead drop style. The group members know and share the access password. They can all log in to the same account and upload or retrieve messages or documents. Of course, the online storage provider could also view the content of unencrypted messages saved in the storage space.

Encrypting content will be discussed later. Nevertheless, dear readers, you will become increasingly aware that protecting message content is vitally important for protecting anonymity. Therefore I would like to anticipate encryption at this point: For the authentication and the later added encryption of this dead drop type communication, friends can share a private/public key pair and the access password for the account.

Sharing a private/public key pair is not the textbook procedure, but I have personal experience communicating in this way. In my university's works council, we have used a shared public/private key pair for many years and conducted sensitive electronic communication smoothly and with confidence. (In this case, we didn't communicate anonymously.) There was no suspicion whatsoever that our employer would try to eavesdrop on us. Still, we could only conduct confidential discussions in electronic form without restraint because we knew for sure that the owner of the infrastructure, who would also be our opponent in negotiations, could not have access to the content of our communication.

#### 4.3.7 Serverless message exchange using OnionShare

An attacker might exploit a dangerous point of attack in anonymous email communication. It could correlate TOR activity times of many IP addresses with the login times of an alias account. An anti-freedom email provider might provide account login times to the authorities, which can then attack in this way and correlate login periods with the TOR activity times of the surveilled citizens. We can neutralize this point of attack by excluding the external email server.

Your computer adopts the role of the server. It temporarily opens a simple TOR Darknet website and offers the file to be transferred for download on this website. Your communication partner (ideally no one else) knows the corresponding Onion Darknet address from a timely message over an "out of band" channel or because you could have given it to him conspiratorially in advance. Your partner visits this Darknet page using the TOR-Browser and downloads the file. Then, after the first download, you remove the Darknet page from the net. Your partner may complain if he did not receive the file because an attacker had pushed in front of him. Fortunately, you had encrypted your file using a method of your choice. You try it anew with a new Darknet address.

You can achieve such kind of data transfer using the tool OnionShare<sup>6</sup>, written by Micah Lee. The tool is available in TAILS by default, and you can find it in the repositories of major Linux distributions. You can download the Windows version from the web page mentioned in the last corresponding footnote, including its digital signature. After verifying the digital signature of OnionShare, you can install it safely on Windows.

I just now create such a Darknet page with OnionShare and the current state of this manuscript as payload. . . and get "http://pws2toqzi4udn2ck.onion/asia-tat" as onion address here and now<sup>7</sup>. The address component "pws2toqzi4udn2ck" is the first half of the sha1 hash value of the public part of a 1024 bit RSA key pair used by Onionshare. The software expresses it in Radix-32 encoding. The user proves his authorship of the page to the TOR network using this RSA key, which is something like the ID of the page. TOR sends a cryptographic Nonce (a number used only once) to the user who wants to publish the page. In the protocol response, the user's system must digitally sign the Nonce with the private part of the RSA key pair and send the result back. After successful verification of the signature, TOR can assume that TOR is communicating with the rightful owner

---

<sup>6</sup><https://onionshare.org/>

<sup>7</sup>At the time of writing, OnionShare did not yet offer the long TOR(v3) Darknet addresses.

and publishes this Darknet page at the address calculated from the corresponding public key. All this happens automatically without a need for user action. The URL addition "/asia-tat", called the slug, consists of random words and is not passed on to TOR. If an attacker gets to know newly appearing onion sites from close TOR monitoring or from operating a TOR node and tries to retrieve the content, the retrieval will not give meaningful results. Since the attacker can download the payload only via the subpage, in this case, the slug "/asia-tat", it cannot obtain the file before the legitimate addressee if it cannot guess this addition. After the first successful information retrieval from the site, OnionShare takes it off the net by default. In case the sender wishes to serve more receivers and changes settings to allow for multiple downloads, OnionShare uses the slug as a means of protection. After several downloading attempts using incorrectly guessed slugs, OnionShare takes the site off the TOR network.

A powerful attacker could, of course, log the data traffic during regular retrieval by the intended recipient and, if necessary, break the standard encryption used within TOR. Remember that in the trove of leaked CIA documents called Vault7<sup>8</sup>, the unknown CIA instructor called the standard encryption "merely a blender layer". For this reason, we should always manually encrypt the files for the recipient in sharp use. Of course, we should encrypt it in a manner compatible with anonymity.

A tremendous advantage of the pattern used in OnionShare is that a one-time unmasking, discovered, or undetected by the eavesdropping victim, is not a permanent problem in a nation under the rule of law. On the next transfer, the user starts the procedure anew with a different Darknet address and a different RSA key pair. The recipient could protect his true identity by using the TOR-Browser and needed only receptive anonymity anyway.

So, after a good half hour with a tea break, my above Darknet address, <http://pws2toqzi4udn2ck.onion/asia-tat>, has not been accessed (no attack ;-), and I will take it off the net again.

Unfortunately, the user cannot determine the Darknet address in advance in the currently offered version of OnionShare. But since version 1.2, there is at least the possibility to enforce a constant Darknet URL. I recommend using this mode of operation. Then you may inform your contact, secretly and in advance, about the upcoming offer for downloading and the scheduled time interval of availability. With a locally calculated, but not yet used Darknet URL, you could even arrange the time and especially the new URL in advance and tell your partner about that by transferring encrypted files in advance. This precalculation would be easily possible in principle. The function would only have to be coded and included in OnionShare.

## 4.4 Exercises for expressive anonymity

We will now begin to practice speaking out anonymously in the digital world.

### 4.4.1 EXPx1, Register anonymous chat accounts, use Pidgin to chat anonymously

*Difficulty: easy.*

*Time: about 30 min each.*

*Prerequisites: RSx6, RECx3.*

*Note: The completion of this exercise is a prerequisite for many other exercises.*

You may want to work without a partner in this exercise. In that case, you can anonymously create two accounts yourself, preferably on two different computers, and then conduct anonymous chats between them. Now let's start with two preparatory steps.

- You first have to find an idealistic provider for free Jabber accounts for this exercise. Start a search in the TOR-Browser using your preferred search engine. If your preferred search

<sup>8</sup>see: [https://en.wikipedia.org/wiki/Vault\\_7](https://en.wikipedia.org/wiki/Vault_7)

engine refuses to work in the TOR-Browser, it should have been your favorite search engine for the longest time. Probably, you'll find the following link to useful lists among other search results:

<https://list.jabber.at/>.

Make sure to visit the web interface of the providers exclusively via the TOR-Browser.

- Register two Jabber accounts. Of course, jot down the Jabber addresses and the access passwords in a safe place.

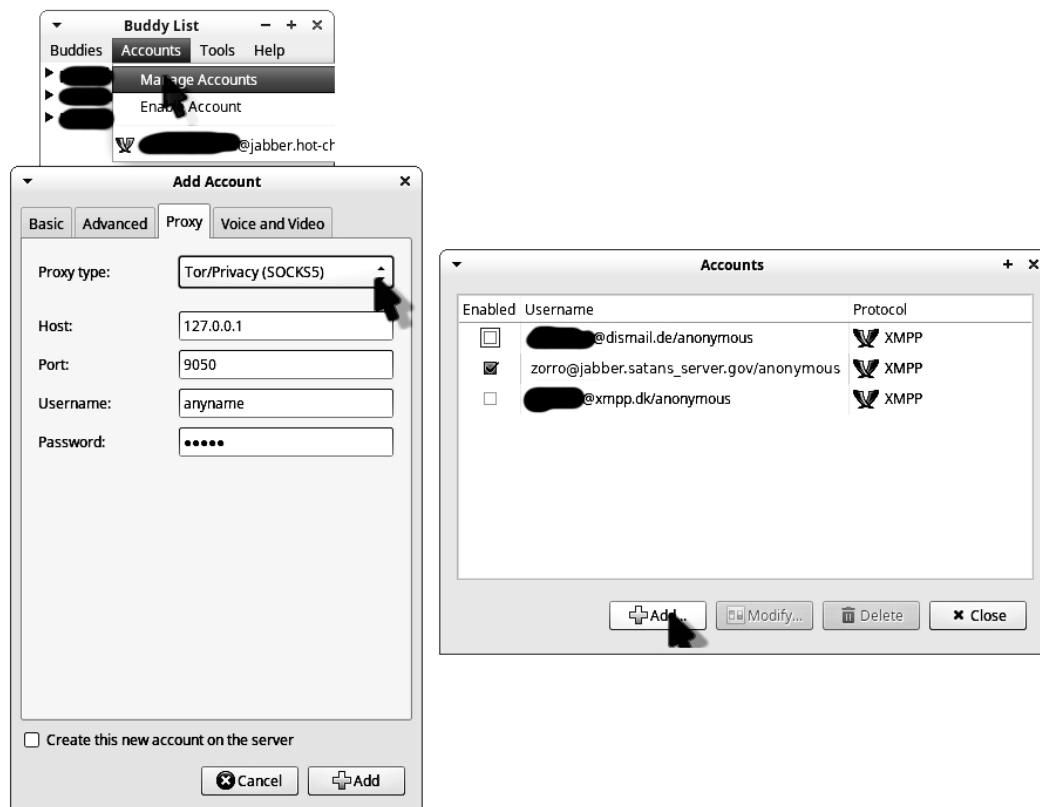


Figure 4.3: These are the Pidgin dialog windows for setting some account parameters. This is a screenshot of my Pidgin configuration. For privacy reasons, I blackened some parts of Jabber names.

When you then enter the account settings in Pidgin dialogs, make sure that the accounts are accessed exclusively via TOR. Since not everything is self-explanatory there, I'll give you a rather detailed instruction, which I based on a description by Micah Lee<sup>9</sup>:

1. Keep the Jabber chat address (e.g., "zorro@jabber.satans\_server.gov") and the password for the account (e.g., "Hellfire") handy.
2. Open the client software Pidgin and select the menu item "Accounts", as shown in figure 4.3. In the menu item, select the sub-item "Manage Accounts". Pidgin will open the window shown in landscape orientation. There you select "Add". The window shown on the left will open. This window has four tabs "Basic", "Advanced", "Proxy", and "Voice and Video". You will need to edit the first three of them. The Pidgin version I am working with now activates the tab "Basic" first. You may select the protocol XMPP in the "Basic" tab already. But please don't set anything more than that for now.

*Note: To avoid burning the anonymity of the account, edit the tab "Proxy" first and return to "Basic" later to enter more account data. That way, you force Pidgin to use TOR when it can*

<sup>9</sup><https://theintercept.com/2015/07/14/communicating-secret-watched/>

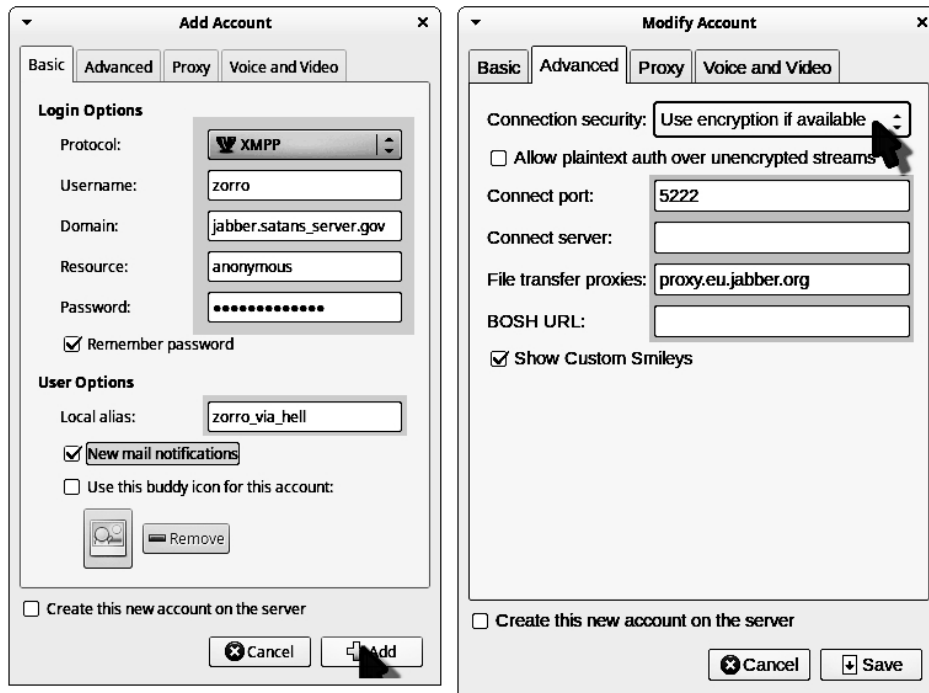


Figure 4.4: Left: Tab “Basic”, Right: Tab “Advanced”.

*log in to the chat server for the first time.*

3. As shown in the picture, select "TOR/Privacy (Socks5)" in the tab "Proxy". This routes the data traffic to the account through TOR. Enter 127.0.0.1 as Host (the IP address of "localhost"). Micah Lee recommends setting Port to 9150 for Windows and to 9050 for Linux.

*Note: I am passing on a recommendation here that was given on the web by Micah Lee and others. Let me point out that I was also able to work with port 9150 under Linux without any problems.*

4. In the following two fields, "Username" and "Password" in the "Proxy" tab, you do not need to enter anything. If you do - any name and password that is only used here - again according to explanations circulating on the net, Pidgin creates a separate TOR path, which differs from any other TOR data stream on the system. If this were indeed the case, it might improve security against attacks from a malicious TOR exit node or the contacted chat server.

*Note: Checking the outgoing packets for two accounts with two different name settings in the Proxy tab revealed to me that at least the TOR entry node selected by Pidgin (version 2.12) was constant and independent of any entries in these fields. In another later case, I simultaneously logged in to three anonymous chat accounts via Pidgin and chatted in three pairs of accounts concurrently. But I detected TOR traffic with two entry nodes only on my system. So we should be careful with such claims circulating on the web. There is some indication of TOR path reuse in Pidgin despite using different names in the Proxy tab when operating several accounts via TOR in parallel. Simultaneous TOR path reuse for different chat accounts could break anonymity.*

5. Now edit the "Advanced" tab (see figure 4.4, right side). The only setting you need to adjust here is "Connection security". Select "Use encryption if available", as shown in the picture. Later, after the introduction of encryption, you can upgrade to "require encryption" if necessary. You can leave the default entries of 5222 for the "Connect Port" and of

proxy.eu.jabber.org at "File transfer proxies" as they are.

*Note: I apologize for the small-scale instructions, unworthy of you as intelligent computer users, to enter cryptic magic words in cryptically titled fields sometimes without any meaningful explanation. Sometimes we have to go through such a digital thicket.*

6. We now return to the "Basic" tab (see figure 4.4, left side). You may have already entered the protocol XMPP at the beginning here. If not, please do so now.
7. Next, we get to the entries for the account data that the client pidgin needs to contact the server of your chat account.  
Enter the Jabber name, the part of your Jabber address in front of the "@", into the field labeled "Username:". In our screenshot, this is "zorro". In the field "Domain", you should fill in the part behind @ which stands for the server name of the chat service. In this example, it is "jabber.satans\_server.gov". Enter anonymous in the field "Resource" and the password of the chat account in the field "Password".  
*Note: The entry "anonymous" in the Resource field is an inner label only. If you like it better, you can readily enter "Hurray\_for\_Snowden" or "impeach\_Trump" instead.*
8. I recommend that you now tick the checkbox for "Remember password". Then, you might set a "Local alias", a nickname for this account. In this example I chose "zorro\_via\_hell".
9. Now click on Add to use the newly registered chat account. If everything went well, you should see a buddy list window and the chat status should change to Available.
10. Then start the chat with your partner or enter your second account for the chat exercise and start to chat between the two accounts of yours. No one but you and your chat partner will know that you own one or both accounts.

#### 4.4.2 EXPx2, Transmit a file anonymously using chat accounts

*Difficulty: medium.*

*Time: about 30 min.*

*Prerequisites: RSx6, RECx3, EXPx1.*

*Note: The completion of this exercise is a prerequisite for many other exercises.*

1. Prepare a short, innocuous file containing one sentence using your simple text editor. We will transfer it without encryption later. Make sure to avoid any details that might give an attacker any information about your identity.
2. You and (if involved) also your buddy should activate their anonymous XMPP accounts. Make sure that each one displays the respective buddy account as active and available for chat.
3. Find out how to transfer the file to your buddy account.

*Note: You will quickly find out how to technically manage file transfer in the anonymized chat via Pidgin. But the devil is in the details! You must search for anonymizing details in a nontrivial text or file to be transferred and remove them before you send the plaintext through digital enemy territory.*

*Just blacken identifying information in the scan of a text or replace telltale words by "\*\*\*\*\*". Intelligence services do that when they have to hand over files to parliamentary investigative committees. But please only conceal what is necessary to hide. Avoid being as ridiculous as the German intelligence services in their hearings concerning the NSA affair and the Snowden revelations. They impertinently handed over completely blackened pages to the supervising parliamentary committee. German news broadcasts showed this on television at prime time. With this public performance, the intelligence services had set themselves up as a prime target for comedians.*

*Also, remember to encumber linguistic analyses and cleanse texts of characteristic personal*



*stylistic features. Render your language featureless and plain.*

4. Transfer the file using the Pidgin chat. The buddy should acknowledge receipt in a chat message (or signal the failure and ask for a new attempt).
5. The buddy should open the file, introduce a small change, and return it in the same way.
6. You open the returned file, find the alteration, and tell your buddy in the chat about the change.
7. Your buddy takes this as a confirmation of successful transfer, says goodbye, closes the chat, and deactivates the account.
8. You also deactivate the account and close this exercise.

#### 4.4.3 EXPx3, Register email accounts anonymously, transfer emails with and without attachments

*Difficulty: medium to hard.*

*Time: about 60 min.*

*Prerequisites: RECx3.*

*Note: The completion of this exercise is a prerequisite for many other exercises.*

Please carry out the practice with a buddy. Alternatively, you can do the exercise by yourself, like the previous exercise.

Make sure to attempt registering the accounts exclusively via the TOR-Browser. If you consistently fail to register via TOR, you may try another mode as a workaround. Use a freshly and safely installed simple web browser like, for example, Midori<sup>10</sup> through a VPN. If successful, delete the used browser from the system afterward, including all its configuration data. Remember, this is only a workaround. Using TOR for registering is to be preferred. Subsequently, access the emails exclusively via TOR. If you managed to register anonymously, please think twice before you delete the account. An email account with truly unbroken anonymity is a treasure.

1. If you decide to do the exercise on your own, register two anonymous accounts yourself (otherwise one each), preferably with different providers. Even if you exercise with a partner, it might be wise to register two accounts yourself already at this point. You might need the second account later. Create and use these accounts via the web interface of the email provider exclusively in the TOR-Browser, preferably with blocked script execution. Please do this according to the instructions in 4.3.5.

*As a reminder: Here and now, <https://www.Safe-mail.net> worked. If you have already created anonymous Jabber accounts, it might be worthwhile to check the web interfaces of the Jabber servers for free email accounts. I could register an email account on direct personal request at the time of this writing at <https://dismail.de/>, for example. Let me express my thanks to the unknown friendly administrator at dismail.*

2. Exchange emails between the accounts. Please be careful not to convey any personally identifying information or personal language style elements in the email content.
3. Be fair to the email provider and delete your accounts if you no longer want to use them or if you have "burned" anonymity, for example, by accidentally touching your nonymous communication bubble.

<sup>10</sup><https://astian.org/en/midori-browser/download/>

**4.4.4 EXPx4, Chat using Ricochet via TOR in rendezvous mode (primary anonymity)**

*Difficulty: easy.*

*Time: about 30 min.*

*Prerequisites: RSx6.*

1. Install the chat program Ricochet safely. Begin by searching for the source of this program on the web using your preferred search engine. The search engine might refer you to the following address: <https://ricochet.im/>.  
*Note: Linux users should find the Ricochet-IM in the repositories of their distribution as well and can employ their Linux's automatism for securely installing the Ricochet instant messenger from the repositories.*
2. Download the installer of the Ricochet-IM.
3. As an optional, additional sub-exercise for users who can/could employ the builtin authentication of their operating system, please manually authenticate a manually downloaded installer of the Ricochet-IM. For manual authentication, the public key of John Brooks, who signed the installation files, can be found on the Ricochet homepage.
4. You and your partner should each start Ricochet and jot down your respective Ricochet addresses. Take care to transfer the Ricochet addresses conspiratorially if you work with a real communication partner.  
If you don't want to exercise with a partner, please create two accounts on two different computers yourself and conduct anonymous chats between them. It was not possible here and now without tricks to work on one system with two Ricochet Darknet addresses.
5. Start chatting.
6. Check if you can transfer files using Ricochet. Here and now, while writing these lines, it did not work yet.

*Note: Ricochet uses TOR encryption as end-to-end encryption. But the TOR key lengths are too small (RSA 1024 bit, at least 3072 bit would be required, also the ECC and AES key lengths are at the minimum limit for TOR). The encryption is, therefore, too weak when you are facing a resource-strong attacker such as a nation-state intelligence agency. This point has been openly criticized in a security audit by the NCC group<sup>11</sup>. As with task 4.4.2-EXPx2, you should take into consideration that the opponent can break 1024 bit RSA and might have access to the message content. Hence, you ought to clean the messages of personally-identifying information.*

*As soon as Ricochet offers the possibility for anonymized file transfer, you could transfer manually encrypted ciphertexts. Then you could use Ricochet for transmitting secrets safely and anonymously, as safe as TOR anonymity can be and as safe as your manually selected encryption method is.*

**4.4.5 EXPx5, Exchange files with your partner via OnionShare (primary anonymity)**

*Difficulty: easy.*

*Time: about 30 min.*

*Prerequisites: RECx3.*

*Note: The completion of this exercise is a prerequisite for many other exercises.*

1. Install the OnionShare file transfer program securely. Begin by searching for the Internet source for this program's installer in your favorite search engine. You may find the following address: <https://onionshare.org/>.  
Linux users will find OnionShare in the repositories of their distribution and can authenticate via the operating system's automatism. The TAILS OS already comes with OnionShare. You

<sup>11</sup><https://ricochet.im/files/ricochet-ncc-audit-2016-01.pdf>

can even authenticate the Windows version automatically employing Authenticode if you trust Microsoft.

2. Optional: To perform manual authentication of OnionShare's installation file, you have to do some research first. You need to find a trustworthy source for the public key of Micah Lee, who developed and maintains OnionShare and who signed the installation files. Please download the key, download the installation file's digital signature and check the triple of the installation file, Micah's digital signature of the file, and his public key for consistency. If you can verify the signature, you may install with the good feeling of having taken the direct route to security without the detour through Microsoft corporation's basement.
3. Please use OnionShare to transfer a file now.

If you don't work with a partner, please transmit the file to yourself. You send with OnionShare, and you receive with the TOR-Browser. For this exercise, we may use both programs on the same computer at the same time. The connection is of the rendezvous type. Transfer a file without personally-identifying content. As in task 4.4.2 on page 93, you should assume that a powerful attacker might have access to the file content. Please clean the file of personally identifying details (or encrypt it as described in the later sections of this book) before handing it over to OnionShare.

OnionShare is self-explanatory. If you have called OnionShare, you can drag and drop files and folders into the OnionShare window. OnionShare then packs the files to be transferred into a zip archive and opens a TOR Darknet web page. The page will have an onion address derived from the Darknet page's RSA key. The recipient can then download the archive from that address. Optionally, you may also configure the software to stick to a precalculated permanent onion address instead. After publishing the Darknet web page, OnionShare displays the onion Darknet address used to post the file. The sender of the file must then tell this address to the potential recipient conspiratorially and out of band.

Like Ricochet, OnionShare relies on the encryption provided by TOR itself, which is undoubtedly insufficient today when we still use TOR(v2). I am personally somewhat reluctant, however, to even fully trust TOR(v3), since it uses US-approved encryption. However, using an appropriate encryption tool, the file to be transferred can manually be encrypted in an anonymity-compatible manner, before sending it. After that, of course, only the ciphertext would be handed over to OnionShare for transfer.

#### 4.4.6 EXPx6, Publish a Darknet website using OnionShare

*Difficulty: easy.*

*Time: about 30 min.*

*Prerequisites: RECx3, EXPx5.*

1. With newer versions of Micah Lees OnionShare, you can publish regular Darknet websites. In this exercise, you will do just that. If you have not yet installed OnionShare, please do so now.
2. Create a text file on your computer containing the message you want to tell to the world anonymously via the TOR Darknet. For your first excursion as Darknet author, I suggest something along the lines of the fairy tale "The Emperor's New Clothes" by Hans Christian Andersen. Surely you will replace my suggestion with whatever you may find appropriate. Create a text file with your office suite, for example, containing the two lines "The emperor's new clothes?" and "The emperor is nude!". Save the document on your hard disk.
3. Use your office suite to export the document to HTML format. Create a folder on your hard disk, for example, called "my\_first\_darknet\_page" and save your HTML file there. Rename it to "index.html".

4. Open OnionShare and select the tab named “Publish Website”.
5. Drag and drop your file named index.html into OnionShare’s window area. Click on “start sharing”.
6. Copy the Darknet address and save it at an appropriate place.
7. Open your TOR-Browser, copy the onion address into the browser’s URL line, and access your Darknet website by hitting the return key. Verify that the site is working flawlessly.
8. You may not like the Darknet website’s title yet. Open the website’s file in your folder "my\_first\_darknet\_page" with a plain text editor, for example, the one you installed in exercise 2.5.1, and study the contents of the file. You will find lots of comments, lots of XML tags beginning with an opening `< . .` followed by some content and ending with finishing character `. . >`.  
Search for the XML starter tag `<title>` and the assigned finishing tag `</title>`. Read the text in between and change it to something like, for example, "enlighten the people!". Save the updated file.
9. Confirm that you can see the new title as the heading of the browser tab and as the heading of the browser window if you click the tab.
10. If you have a friend who is able to use the TOR-Browser, tell your friend your Darknet website’s onion address confidentially. Ask your friend to visit your Darknet website and verify that it is working.
11. After enjoying your authorship for a while, take your Darknet site off the TOR network by terminating OnionShare.

*Note: This is by far the easiest way to publish a website in the Darknet and works with any Internet connection that does not block TOR. Using a Raspberry Pi<sup>12</sup> as your system consumes little energy, about as much as a dim lamp. It certainly needs less energy than your home router. Currently, however, it takes some fiddling with the OnionShare software to get it running on a Raspbian (the standard Linux optimized for the Raspberry Pi) since the Raspbian repositories do not include OnionShare yet.*

#### 4.4.7 EXPx7, Search the web for information on TOR rendezvous paths, read and understand

*Difficulty: easy to medium.*

*Time: about 40 min.*

*Prerequisites: RECx3.*

1. Search the Internet for a document that explains the selection and construction procedure of the TOR rendezvous path. Use the TOR-Browser and your preferred search engine.  
You might find the web page given in the footnote<sup>13</sup>, among other helpful hits. Here and now, at the time of writing these lines, a valuable document was accessible at this site.
2. Please read and understand it.

<sup>12</sup>The Raspberry Pi is a fully-fledged minicomputer costing about 30-40 \$, having the size of a cigarette package. You could easily hide your Raspberry Pi Darknet server behind a flowerpot at an innocuous place. It is extremely energy efficient, runs using a multicore ARM processor, and doesn’t even get warm if not taxed heavily.

<sup>13</sup><https://www.torproject.org/docs/onion-services>

#### 4.4.8 EXPx8, Register an I2P email account using Susimail and exchange emails with a TOR-anonymized conventional email account

*Difficulty: medium to difficult.*

*Time: about 45 min.*

*Prerequisites: RECx11.*

*Note: The completion of this exercise is a prerequisite for other exercises.*

Please note: If your connection to the Internet is protected by a firewall that you are not allowed to configure yourself, this exercise will probably not be doable. I2P requires network access through ports that are rarely used and are usually blocked in an enterprise network.

1. In case you have not yet installed I2P, please do so now (RECx11). Contrary to I2P recommendations, do not configure I2P for permanent background operation. You should be able to easily switch the tool on and off and stay aware of I2P's state. If not needed, I2P should not be active. Security-conscious computer users always have an eye on the system's data traffic while working and compare it with what they expect from the current tasks. Constant background I2P-traffic could, in the worst case, provide cover for malicious data transfers triggered by hacker attacks. However, under certain conditions, background traffic can also be used as a countermeasure against big-data attacks.
2. Now start I2P and the browser configured for I2P use. From this browser, launch the "I2P Router Console". Select the "Email" field under the "I2P Services" category.
3. Register an email account there for one of the fictive identities you created in section 4.3.5 on page 86.
4. Then exchange messages and file attachments with a TOR-anonymized email account used in the previous exercise 4.4.3 on page 94.

*Note: There may be a long latency (hours) between sending the email and receiving it outside I2P.*

#### 4.4.9 EXPx9, Practice buying an electronic device anonymously

*Difficulty: medium to difficult.*

*Time: hours.*

*Prerequisites: none.*

This "analog exercise" is intended to make you aware of how the noose around the neck of our privacy is slowly tightening, unfortunately, without an outcry of protest. Imagine you had to discreetly contact a human rights activist or a family member outside the country without revealing your identity to the surveillant. Imagine you live under omnipresent government surveillance. But you have never been noticed personally by the authorities. So your communication, electronic and otherwise, is not being specifically targeted yet.

Every network-capable device identifies itself to a WLAN, which the device tries to connect to, by transmitting its unique MAC address. In the smartphone sector, this may happen without user awareness. That is a big privacy problem with WLANs operated by untrusted bodies (coffee shops, department stores, airport WIFI operators, etc.).

Let us assume the government knows your private IP address and the MAC addresses of all your computers or smartphones. Deplorably, these are realistic assumptions for all of us. Therefore you need a new device whose MAC address is not linked to your name in a government database yet. For this exercise, you may only simulate the purchase or buy a little something, like a memory stick. And you can do without the disguise mentioned below. However, you have to accept a loss of realism then and will have a less immersive experience.

1. So let's go shopping! Put on - at least in your mind - a disguise: hat, winter coat, dark glasses, maybe even a fake beard, and leave your apartment unobserved with the appropriate amount

of cash. Under no circumstances should you carry a mobile device (mobile phone, Barbie doll with IP address, etc.) with you. Minimize your track, for example, pay in cash for public transportation, don't use your credit card on the way, etc. . .

2. Go to an electronics store where you have not been a customer before. Observe the placement of the surveillance cameras and avoid looking into them head-on from a short distance. A powerful attacker might be able to determine the place and time of purchase based on the MAC address of the purchased device and view the corresponding stored video recordings.
3. Choose your purchase. For exercise purposes, a simple memory card or thumb drive might do the trick. Quickly proceed to checkout. Pay in cash and leave the store. In a secluded place, you can take off your disguise, at least in your mind. Then return to your home.
4. In sharp use, the difficult part of neither burning MAC-anonymity nor general online anonymity when you start using the device would still be pending. Contemplate how to make contact electronically, for example, how to set up the new device and how to send messages then anonymously and via a free third party WLAN without inadvertently exposing your identity.

An additional purpose of this exercise is to impart the experience that for resilient security, digital and physical measures must often go hand in hand. In the professional environment, they say: "When we talk real IT security, we also talk guard dogs, barbed wire and security men."

This is equally valid for private IT security, though, with somewhat different coloring and implementation: How well is your computer's site secured? Is it possible to spot from a publicly accessible location who works there and when? Could an innocuously looking SUV, parked outside your house, connect to your WLAN? How and where did you buy your electronic devices? Did you perhaps have a switched-on smartphone in your pocket when erroneously assuming an anonymous purchase?....

You may have noticed that beginning with anonymous digital communication is extremely difficult once you got into the focus of government surveillance. Using the possibilities of anonymous communication as early and as innocuously as possible is extremely important. The goal is to avoid getting into the surveillant's closer focus in the first place.

Fortunately, many Internet users already protect themselves against ubiquitous surveillance by using TOR. In early summer 2018, there were almost 200 000 daily TOR users in Germany. During the following two years, the number rose to about 300 000. Thus, it may be too costly in my country for even the most aggressive and the best-funded domestic state intelligence services to place all TOR users under close surveillance. Monitoring and archiving their Internet traffic just because they use this anonymization service would bind substantial resources. Surveilling all unsuspecting, normal Internet users, on the other hand, is technically feasible, easy, and cheap. The government can do that on a large scale and in an automated way. Snowden revealed that the U.S. intelligence agency NSA did that. It would be wise to assume that all the nation-state's intelligence services will do any surveillance that is technically feasible and inexpensive for them.

## 4.5 Attacks on expressive anonymity

Repressive governments have a deeper motivation for attacks on expressive anonymity than for attacks on receptive anonymity: They suspect conspiracies, coup attempts, and dissident networks everywhere. Expressive anonymity enables both underground agitation and whistleblowing as well as the covert leadership of a group of dissidents. Furthermore, it is easier to attack than receptive anonymity. That is why, unfortunately, western democracies' state authorities attack expressive anonymity with great effort. They claim to do that to prevent cybercrime.

All attack modes relevant in receptive anonymity also exist for expressive anonymity. I will not describe the scenarios here that we already looked at in section 3.6. We can find two new attack

options in the case of expressive anonymity, which I will describe below.

#### 4.5.1 Nonymizing information in transmitted data

This point has already been mentioned occasionally in the exercises. If unencrypted or insufficiently encrypted information is transmitted and attacked by resource-strong government organizations, the attacker may have access to this information. At the current status of TOR(v2), the encryption used by TOR(v2) is insufficient. The algorithms used in TOR(v3) are still considered safe. My gut feeling tells me, however, not to unconditionally rely on the security of a tool co-financed by the U.S. administration. So I do think TOR use should be supplemented by manually added reliable encryption from other sources.

Images or sound recordings intercepted in transmission may allow the identification of depicted persons or recorded voices. Many types of media files should, therefore, be taboo. Even in the case of texts, the so-called fingerprinting of the linguistic style can characterize the author. It can narrow down his or her social or cultural background or nonymize the author personally. We must thus carefully cleanse the files of characterizing or even personally-identifying information before transferring them through TOR. However, this is so costly and time-consuming that the practicability of expressive anonymity without an additional layer of reliable encryption is severely limited. It is, therefore, hardly used in this pure form. Nevertheless, logical and didactical reasons have led me to treat this mode of communication as a separate chapter in this book.

#### 4.5.2 Big data attacks

The self-description of TOR succinctly states that TOR cannot enforce anonymity against a global opponent. Let me explain this here in more detail. Put yourself in the position of a powerful attacker (the global opponent), who records all activities of all Internet users in its political domain: When had users X, Y, or Z which devices connected to the Internet and at what geographical location? Which Internet addresses did they visit? Which emails did X, Y, or Z send or receive, at what times did they maintain TOR traffic? What additional information will Internet providers supply and, if applicable, at which Internet portals were X, Y, or Z logged in?

It is a plausible assumption in our modern times that different snoopers document all unprotected Internet activities of all users in various databases. Only the contents and the accessed IP addresses of the TOR data traffic remains unknown to the surveillants. However, the fact that a user employs TOR and when the user does so is difficult to conceal. As a government-funded attacker, you might like to know, for example, whether X, Y, Z, or another Internet user from your sphere of influence is controlling the initially anonymous account "Johndoe@jabber.snoopers-palace.gov", whose government-critical messages you found unpleasant.

Now, unaffected by any encryption that may be present, Big Data comes into play: You know the last - let's say 10 - times at which the "Johndoe" account actively communicated. In figure 4.6 on page 102, you can see that the U.S. has about 400 000 TOR users per day, and my home country Germany somewhat less than 300 000 per day. So if the average U.S. TOR user is active every other day, we can estimate to have about one million different TOR users in the US. I would expect about the same number in Europe. So you may know, for example, that there are about one million TOR users in the political domain you monitor.

For the standard TOR user, let me assume that he/she has, on average, about half an hour of TOR traffic per day. So the probability that he/she is active on the Internet via TOR at a specific time is  $1/(2*24)$ , about one-fiftieth or 0.02 of this million. Your snooping-big-data-software (Xkeyscore from the NSA, for example) gets the engine fired now.

First known activity time of Johndoe: You can exclude the fraction 0.98 of all TOR users because they were inactive at the given time. This leaves a fraction of 0.02, i.e., 20 000 known TOR

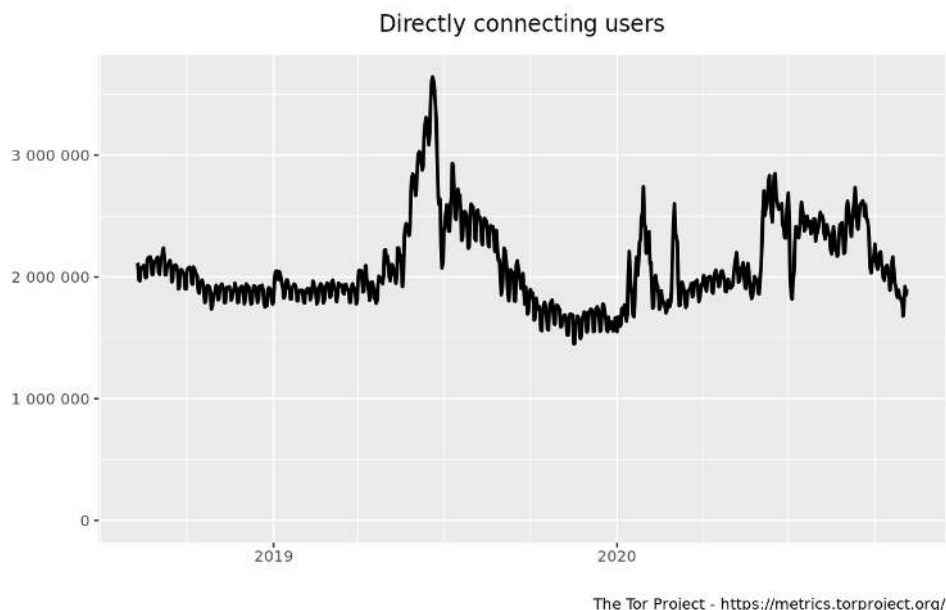


Figure 4.5: Worldwide numbers of users directly connecting to TOR. Excerpt from the website: <https://metrics.torproject.org/userstats-relay-country.html>, screenshot from 11/08/2020.

As with other privacy tools, the global political situation leaves its traces in the logs. TOR use peaked in Summer 2019. This peak correlates with the escalating political situation in the middle-east. There was a sharp million size spike of users connecting from Iran, leaving its signature in the worldwide numbers shown here. European countries showed a slightly enhanced activity then, and the U.S. numbers remained almost flat through this crisis. The increased activity in spring 2020 is probably due to the covid19 pandemic. Please feel encouraged to visit the website and experiment with the settings.

#### Optional Exercise:

Find out which nation's users caused a sudden increase in TOR access after May 31, 2020. The access numbers begin to sink in September, descend linearly, and reach the baseline again in November 2020.

*Note: The increase started a week after the murder of George Floyd. The sharp drop in the summer is around the 4th of July. An important nation had elections in November.*

Don't believe anyone who wants to discredit TOR usage and claims that TOR darknet users mainly use it for child porn sharing, money laundering, drug trafficking, and other criminal activities!

users you suspect of being Johndoe. Second activity time of Johndoe: Again, one-fiftieth of the previously suspected users remain, i.e.,  $20\,000/50 = 400$  suspects. Third time of activity: 8 suspects remain. And oops, after the fourth activity time recorded, only one suspect remains.

You can now reprimand this suspect with the full severity that your state's rules allow. If two or more are left, you have even caught Johndoe's chat partners along with him and can now arrest the subversive communication circle in its entirety. For you, being a member of a state intelligence service (or a member of the secret police), not everything always works out optimally. Suspect's activity times might eventually follow regular patterns. Or participants of the dissident network may sometimes be missing in the chat. So you might need eight instead of four activity times. But that is still less than the ten activity recordings that initially appeared to be very sparse. Homing in on the suspect depends exponentially on the number of activity times known to the powerful attacker.

We soberly slip back into the role of the spied upon Internet user and conclude that we should urgently move our account "Johndoe@jabber.snoopers-palace.gov" to a provider outside our political domain. We could then hope that our home country's intelligence services could at least not force the provider of the account to cooperate with them. Unfortunately, with primary anonymity in the rendezvous type (i.e., without using an external server), the situation is not much



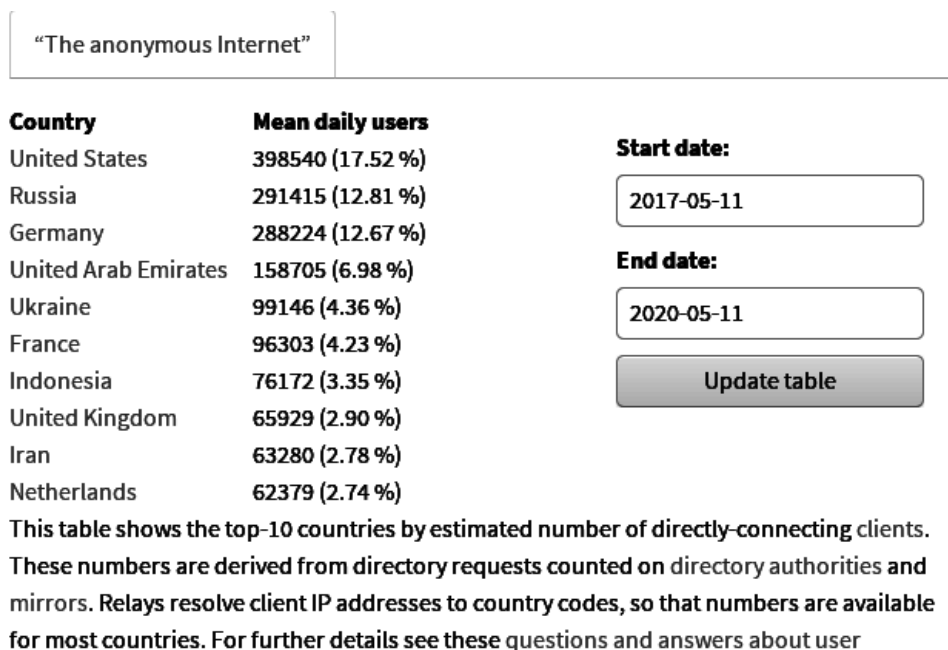


Figure 4.6: Average number of directly connecting TOR users from the top ten user nations during the previous two years.

Rank nine of Iran resulted from a sharp million-high spike lasting for about a couple of months during the peak of the political crisis, almost leading into a war with the US. Excerpt from the website <https://metrics.torproject.org/userstats-relay-table.html>, screenshot from 05/11/2020.

better if our communication partner at the other end of the TOR rendezvous path also lives in our political domain. In this case, the intelligence agencies surveilling the users could correlate their TOR activity times to unmask communicating buddies.

For rescuing our anonymity, we consider the following countermeasures:

1. You make sure that your TOR exit node lies outside your political domain (outside Five Eyes + Western Europe, for example). Besides, the endpoint of our data stream addressed by the TOR outbound node, the email, or chat server, for example, should also be located outside the domain. The attacker must not have access to the TOR outbound temporal activity pattern. Of course, anonymity gets shaky if your communication partner is in the same political domain as you are. This is one of the exceptional cases where well-designed secondary anonymity is superior to the primary anonymity of rendezvous type TOR communication. Better than a direct TOR rendezvous path from Alice in Antwerp to Bob in Berlin is email communication with a Russian TOR exit used by Alice, an Israeli email account of Alice's, an email account of Bob in Amsterdam and a TOR exit for Bob in Liberia, for example. The latency of the messages should be pleasantly high in this case. Bob can make the attack more difficult if he retrieves his emails from Amsterdam via Liberia only about every 1-2 days.
2. You may hide using TOR on your side of the TOR network. But TOR was not developed for concealing its use. Maybe you can connect to the TOR network via a free Vietnamese VPN. Alternatively, you could use a bridge into the TOR network. A bridge is an obfuscated entry point to the TOR network. TOR employs such bridges to bypass TOR blockades in the context of government censorship and for concealing the use of TOR. Such measures would increase the population of suspects for the attacker from one million to tens or hundreds of millions. But unfortunately, the attacker homes in on the suspect exponentially with recorded activity times. Thus the powerful monitor may need to record 16 times of general Internet activity instead of the conservative estimate of 8 times of TOR

activity. Initially, this would not be a substantial security gain. However, it would at least force the attacker to maintain a much larger database. With a bit of luck, it would not yet be able to handle that size technically.

At this point, we can see that our Internet Service Providers would have it in their hands to protect us from nonymization by such big-data attacks launched by governments. The ISPs would only need to protect information about our activity times and safeguard these data, which pile up primarily and exclusively at our ISPs' systems. They would have to defend them against government and marketing mafia's access. However, this is a faint hope. They have already handed out such data willingly to dazzling copyright enforcement lawyers. Chat or email providers could also defend information about activity times of individual accounts against access by authorities. Unfortunately, however, neither of these entities will generally have any interest in exercising this protective function, but on the contrary, they will want to trade in such data. Internet Service Providers (ISPs) and email service providers could protect our anonymity if they wanted to.

3. You can try to cover your tracks at great expense. Organize TOR activity from home devices when you and your mobile phone are on the road. Conversely, you can leave your cell phone turned on at home, organize dummy traffic from your home system (e.g., via I2P), and at the same time, with a spoofed MAC address, conduct TOR traffic with your email and chat accounts over a free wireless network somewhere else.

Any anticorrelated or at least uncorrelated action helps. A relatively potent option would be to have data traffic on your system, 24/7, including anonymous dummy traffic. Then, correlation analyses of your temporal activity patterns could no more give meaningful results because, in such an extreme case, you are active all the time. Unlike TOR, the alternative Darknet I2P always maintains such blinding background traffic. This measure, however, would considerably increase the temporal attack surface. Besides, there would be some collateral damage to the wallet and the earth's climate due to increased energy consumption. All these defensive measures are tedious and cumbersome, yet an intelligent and flexible attacker could easily circumvent them. However, if you have to communicate electronically within a political domain based on total surveillance, the defensive measures listed here in point 3 are your only chance.

In summary, we can conclude that the anonymity of our communication on the Internet is a much more fragile resource than confidentiality, i.e., the encryption of private content, which we deal with in the following chapter.



7-zip, 127

Academic Signature, 123

AES, 125

alias identity, 87

anonymity, 39

anonymous accounts, 82

asymmetric cryptography, 20

asymmetric encryption, 112

big data attack, 79, 100

chat account, 90

complementary attack, 155

configuration files, 133

Confirmation Attack, 56, 77

console, 163

crapware, 18, 36

cryptographic hash function, 106

cryptographic self-protection, 19, 22

digital signature, 20

Domain Name System, 42

ECC domain, 125

ECDSA standard, 124

EGOTISTICALGIRAFFE, 78

ElGamal-cryptosystem, 112

elliptic curve cryptosystem (ECC), 28

Enigmail, 122

enterprise IT-security, 9

expressive anonymity, 39

fictive identity, 81

fingerprint, 28, 35, 53

Forward Secrecy, 116

FOXACID, 78

free VPN, 58

garlic routing, 56

Gnu Privacy Assistant, 120

GnuPG, 22, 119

gpg4usb, 120

gpg4win, 120

graphical user interface (GUI), 27, 30

hash value, 21, 24

hex-editor, 34

hybrid encryption, 113

I2P, 56, 69, 98

Internet access, 42

Internet Service Provider, 25, 42

25

IP geolocation, 58

Jabber, 84

Jailbreak, 18

JH, 124

Key Derivation Function, 105

Linux distribution, 21

- Live-CD/DVD, 22
- Live-USB-Stick, 22
- MAC address, 52, 98
- MAC address spoofing, 74
- man-in-the-middle attack (MITM), 45, 78, 130
- mass data, 41
- Memory Footprint, 111, 142
- modulating data rate, 79
- NADA-Cap, 125, 135, 136
- negligible adversary advantage, 115, 125
- NOBUS principle, 153
- Nonce, 89
- nonymizing information, 100
- nonymous email account, 138
- onion routing, 49
- Onion Service, 85
- OTR, 33, 116, 130, 131
- out of band, 115
- overlay networks, 56
- p7zip, 127
- passphrase, 105
- password, 105
- password hardening, 106
- Payload Size Camouflage, 125
- Pidgin, 84, 90, 130
- political domain, 41
- pre-key, 105
- primary anonymity, 82
- privacy, 105
- pseudo random number generator (PRNG), 29
- pseudonymous account, 82
- public key server, 28
- public WLAN, 52
- pyramid of digital camouflage, 16
- pyramid of dissident protection, 16
- QUANTUM, 78
- Radix-64 encoding, 27, 121
- receptive anonymity, 39
- rendezvous type TOR routing, 85
- ROCA, 17
- routing, 43
- RSA, 27, 112
- salt, 107
- salting, 107
- Seahorse, 120
- secondary anonymity, 82
- selector lists, 159
- Skein, 124
- software updates, 19
- statistical analysis, 79
- stretching, 107
- submarine communication, 15, 137, 146, 148, 150
- symmetric encryption, 105
- Systemd, 35
- TAILS, 52, 54
- terminal, 163
- Threefish, 125
- Thunderbird, 122
- TOR, 53
- TOR rendezvous path, 97
- TOR-Browser, 53, 63
- TorBirdy, 54, 67
- TPM, 18
- trojan nodes, 79
- Vault7, 19
- virtual machine, 48
- Virtual Private Network (VPN), 44
- VPN Gate, 52, 58
- Walled Garden, 18
- Web of Trust, 28, 138
- website fingerprinting, 76
- XMPP, 84
- Zero-Day Exploit, 13, 78, 118

Michael Anders

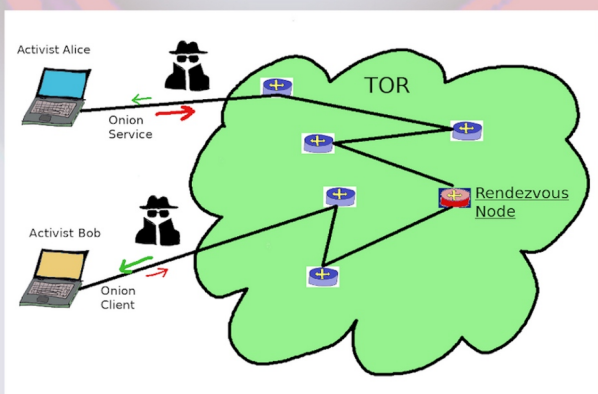
## Surveillance and the Art of Digital Camouflage

57 exercises  
to escape the dragnet  
of surveillance.

- Only a simple Windows or Linux computer required.
- Predicted working time for all exercises.
- Clear division into four skill levels.
- Enables to make good own decisions.
- Exercises get more challenging with increasing reader progress.

### Use TOR in rendezvous mode:

The communication does not require any resources outside the TOR anonymization network. Sender and recipient connect directly via TOR.



Not least since the disclosures of Edward Snowden, we are aware of the ubiquitous collection of our data for commercial and political purposes. However, we are not helplessly at the mercy of the eavesdroppers collecting and aggregating our data against our will.

This book shows how we can protect our sensitive communication even against seemingly overpowering opponents.

In western democracies, the protection of private communication can be a means of preventing further slippage into the surveillance state.

In a threatened or already damaged democracy, the protection of computer-based communication can even be a means of self-protection for critical organizations and dissidents. Lawyers and journalists with powerful, even nation-state adversaries should be able to master such secure communication for the protection of clients and sources.

In this book, we reach the peak of protection in Internet communication. The powerful surveillant cannot uncover who the sender of an intercepted message is or who the intended recipient is. Furthermore, the surveillant cannot gain any information about the type of encryption applied to the intercepted message or whether it is an encrypted message at all. It looks like pure digital noise to the surveillant. All it can find out is that someone may be communicating with someone else via its network infrastructure.