Defend Dissent

Defend Dissent

Digital Suppression and Cryptographic Defense of Social Movements

GLENCORA BORRADAILE

OREGON STATE UNIVERSITY
CORVALLIS, OR

Defend Dissent by Glencora Borradaile is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License, except where otherwise noted.

Publication and ongoing maintenance of this textbook is possible due to grant support from Oregon State University Ecampus.

Suggest a correction (bit.ly/33cz3Q1)

Privacy (open.oregonstate.education/privacy)

This book was produced with Pressbooks (https://pressbooks.com) and rendered with Prince.

Contents

Acknowledgments ix

Introduction: Why Digital Security? 1

Downloading a "Secure" App Isn't Enough 1

Political Scope of This Book 2

Overview of This Book 2

Part 1: An Introduction to Cryptography

What Is Encryption? 6

A Simple Cipher: The Caesar Cipher 6

A Slightly More Complicated Cipher: The Vigenère Cipher 7

In Context: The Unbreakable Onetime Pad 8

Modern Cryptography 10

Security through Requiring Brute-Force Attacks 10

Security Is Not Guaranteed through Obscurity 11

Security Is Provided by Transparency 11

Security Is Provided by Protecting Your Encryption Key 12

Security Is Provided by Distrusting the Infrastructure 12

In Context: The Enigma Machine 14

Exchanging Keys for Encryption 17

A Physical Example: Exchanging a Message without Exchanging a Key 17

A Mathematical Example: Exchanging a Message without Exchanging a Key 18

A Physical Example: Agreeing on a Secret over an Insecure Channel 19

Diffie-Hellman Key Exchange 21

Using Diffie-Hellman Key Exchange 22

In Context: When Good Things Go Bad 22

Cryptographic Hash 25
Using Cryptographic Hash Functions to Prove How Smart You Are 26
What Do Hash Functions Look Like? 26
In Context: Cryptographic Hashes Violate Your Fourth Amendment Rights 27
The Man in the Middle 29
A Physical Man-in-the-Middle Attack 29
A Man-in-the-Middle Attack against Diffie-Hellman Key Exchange 31
Spotting a Man-in-the-Middle Attack with Cryptographic Hashes: Fingerprinting 33
In Context: The Great Firewall of China 36
Passwords 38
When "Password Protected" Does Not Mean Encrypted 38
Password Cracking 39
Best Practices for Passwords 40
Generating Encryption Keys from Passwords 41
In Context: When Precautions Are Not Enough 42
Public-Key Cryptography 44
Revisiting Diffie-Hellman Key Exchange: Public-Key or Symmetric-Key Cryptography? 45
Combining Public-Key and Symmetric-Key Cryptography 46
In Context: Antinuclear Activism and Pretty Good Privacy 47
Authenticity through Cryptographic Signing 49
Cryptographically Signing Cryptographic Hashes 50
Applications of Cryptographic Signing 51
In Context: Warrant Canaries 52
Metadata 54
What Is Metadata? 54
Metadata and the Internet 55
In Context: Protecting a Whistleblower 55
Anonymous Routing 57
Trusting a Middle Man: Virtual Private Networks 58
Not Trusting the Middle Man: The Onion Router 59
Use and Prevention of Anonymous Browsing Technologies 61
In Context: Disruptj20 62

Part 2: Digital Suppression of Social Movements (in the US)

Mechanisms of Social Movement Suppression 65

Modes of Suppression 66

Information Technology Interference 69

In Context: COINTELPRO and the COINTELPRO Era 69

Digital Threats to Social Movements 75

Surveillance Adversaries 76

Surveillance Strategies 77

Surveillance Tactics 78

In Context: Standing Rock 83

Part 3: Defending Social Movements (in the US)

Defending against Surveillance and Suppression 87

Reducing the Threat 88

Where Is Your Data? 89

In Context: Edward Snowden 90

Security Culture 92

Security Culture Meets Digital Security 92

In Context: Saint Paul Principles 94

Protecting Your Devices 96

Physical Attacks 96

Remote Attacks 99

In Context: Compromising Protesters' Phones 100

Protecting Your Communications 102

Encrypted or Not 102

In-Transit Encryption 103

End-to-End Encryption 105

Authentication 105

In Context: Multiparty Video Chatting 106

Protecting Your Remote Data 108

In Context: Trusted or Encrypted Cloud Storage 109

Protecting Your Identity 111
Anonymity versus Pseudonymity 111
Ways to Use Tor 112
Hiding Your Physical Location 112
Tor Warnings 113
In Context: Getting the Real Tor Browser 114

Conclusion: Selecting Digital Security Tools 115
Required Criteria 115
Additional Desirable Technical Criteria 116
Nontechnical Criteria 117

Creative Commons License 119
Recommended Citations 120
Versioning 122

Acknowledgments

I thank Michele Gretes for codeveloping CS175 and the Civil Liberties Defense Center's digital security trainings with me and helping with drafts of this book. I thank the Civil Liberties Defense Center for creating space for educating activists on digital security.

Introduction: Why Digital Security?

In the summer of 2013, Edward Snowden shook the world with a trove of disclosed documents from the National Security Agency (NSA), Central Intelligence Agency (CIA), and a host of other global three-letter agencies. For more than a year, there were weekly (if not daily) revelations of just how extensive these agencies' digital information-gathering capabilities were, particularly those of the United States. It felt as though the NSA was able to get any information that went through internet or phone networks that wasn't encrypted plus some information that was weakly encrypted and some more information that wasn't encrypted on corporate servers. That feeling is close to the truth.

At the time, I had been engaged in environmental activism and was aware of how social movements had historically suffered from state repression that was made possible through spying. The sheer extent of the information that the NSA and CIA were able to gather meant that suppression efforts by the State could be that much easier and more effective. The more information the State knows about your activities, the easier it is for it to interfere with your goals.

So I was worried. Could we combat climate change when the odds were stacked against all the groups working to do so? What about systemic racism? Did we have any hope?

Not long after the Snowden revelations, I partnered with the Civil Liberties Defense Center (CLDC), a nonprofit that provides legal support to social movements that "seek to dismantle the political and economic structures at the root of social inequality and environmental destruction." The CLDC gives know-your-(legal)-rights trainings to social movement participants, emphasizing how to protect and invoke one's First and Fourth Amendment rights: the right to free speech and the right to no illegal searches and seizures. These rights are eroded with mass surveillance. This is quite clear with Fourth Amendment rights, but for First Amendment rights, legal scholars often point to this *chilling effect*: citizens restrict their speech if they know they are being surveilled. To complement the CLDC's legal trainings, I started regularly holding digital security trainings for activists centered on the premise that **encryption is the only way to protect your First and Fourth Amendment rights in the modern world of mass surveillance**. This book has grown out of these educational efforts.

Downloading a "Secure" App Isn't Enough

It isn't enough to download a "secure" app. First, what does "secure" even mean? Security is a complex, subjective, and multifaceted concept. While perfect freedom from risk is usually out of reach, especially when digital technologies are involved, strong relative protections are possible. In order to evaluate or at least explain (and convince a group of people to take advantage of) the relative protections of an app requires some understanding about cryptography and what information is at risk (and with what likelihood) when using a given app or digital service.

Our trainings scratch the surface of the information that I would like to impart. Social movement

participants tend to be busy people and often want a set of simple and doable digital security recommendations from people they trust. My goal with this book (and the companion course at Oregon State University, CS175: Communications Security and Social Movements) is to increase the number of people who know enough to make those recommendations (or at least know how and where to learn more).

Political Scope of This Book

As might be gathered from the references to the First and Fourth Amendments, this book is rooted in the political arena of the United States. While much of the book will be relevant outside of the US, we recommend that anyone applying this knowledge in other countries seek additional advice.

Overview of This Book

This book is not intended to be comprehensive for three reasons:

1. I want this book to be accessible to any curious person. Going into further details in cryptography would require some college-level mathematics. I also believe that one doesn't need to understand specific cryptographic protocols to make reasoned digital security recommendations—one can lean on cybersecurity experts for that.
2. The state of mass surveillance and the apps that are available to counter surveillance are constantly changing. As I put the finishing touches on this book, I am resisting the urge to include the latest news on State surveillance capabilities.
3. I want the book to be short enough to read in a weekend.

The book has three parts as follows.

Part 1: An Introduction to Cryptography

This is a basic introduction to cryptography: enough to understand the basics of what information is protected and what is not and why. Some interesting concepts (such as forward secrecy and blockchains) are left out because I felt that these advanced topics might overwhelm my intended audience. However, the curious reader, after reading part 1, should be able to appreciate, say, the Wikipedia articles on more advanced topics like forward secrecy and blockchains.

When describing cryptographic protocols, most people refer to a cast of characters: Alice sends a message to Bob, and Eve might be eavesdropping on their communications. The companion course for this book focuses on civil rights–era social movements—particularly Black liberation

movements—and the State suppression of those movements. To that end, rather than Alice, Bob, and Eve, we use the running example of Assata communicating with Bobby, with Edgar eavesdropping:

1. Assata Shakur was a member of the Black Liberation Army and the Black Panther Party (in the early 1970s), was targeted by the FBI (as described in the chapter "Mechanisms of Social Movement Suppression"), and is still a political refugee in Cuba.
2. Bobby Seale is a cofounder of the civil rights–era Black Panther Party and was also subject to surveillance and harassment by the FBI.
3. J. Edgar Hoover founded the FBI and has been deemed responsible for the surveillance and repressive efforts of the FBI. We occasionally refer to Edgar as "the Man" where appropriate (i.e., in the chapter "The Man in the Middle," where a man-in-the-middle attack is standard cryptographic terminology).

While the remainder of the book is likely to require significant updates in the coming years, part 1 is likely to stand the test of time.

Part 2: Digital Suppression of Social Movements (in the US)

This part is rather depressing, as it overviews the following:

1. How social movements have historically been suppressed in the US (and where surveillance plays a role) in the chapter "Mechanisms of Social Movement Suppression"
2. What surveillance and other digital threats are in use in the US in the chapter "Digital Threats to Social Movements" In this part, we use "the State" to refer to any constellation of governmental and nongovernmental organizations that represents established power structures with the resources and motivation to deploy a wide range of suppressive strategies and sophisticated technical measures against social movements.

We keep this part deliberately short so that we can move onto the last part, which is more empowering. In part 2, we pick *illustrative examples* to give an overview of how mechanisms of social movement suppression are used and what types of surveillance and other digital threats are in play. The chapter "Digital Threats to Social Movements," in particular, will never be up to date, as new threats and capabilities are constantly being developed and deployed. We hope that anyone who reads this part quickly follows up with the last part.

Part 3: Defending Social Movements (in the US)

Part 3 is intended to be empowering. Starting with threat analysis (which is country and context dependent), we quickly move into *classes* of tools to protect your information. I say classes of tools rather than specific tools because specific tools can come and go as the projects supporting

those tools or apps fail or appear, and it will not be feasible to update this book multiple times a year. This section is country dependent, as the availability of or associated risk of using certain tools can depend on your political context. For example, it can be more challenging to use Tor (an anonymity-providing internet browser, which we will discuss in the chapters "Anonymous Routing" and "Protecting Your Identity") in certain countries that engage in widespread censorship (such as China).

What to Learn Next

- What Is Encryption?
- Mechanisms of Social Movement Suppression

External Resources

- Civil Liberties Defense Center. "About."

PART 1: AN INTRODUCTION TO CRYPTOGRAPHY

What Is Encryption?

What You'll Learn

1. The basic elements of encryption: the *plaintext*, the *ciphertext*, the *cipher* (or encryption protocol), and the *cryptographic key*
2. How some classic encryption methods work
3. Ways that encryption can be broken
4. An unbreakable cipher

Let's start with the basics—think "pen and paper encryption"—before moving on to more complex encryption methods made possible by computers.

Encryption is the process of scrambling a message so that it can only be unscrambled (and read) by the intended parties. The method by which you scramble the original message, or *plaintext*, is called the *cipher* or *encryption protocol*. In almost all cases, the cipher is not intended to be kept secret. The scrambled, unreadable, encrypted message is called the ciphertext and can be safely shared. Most ciphers require an additional piece of information called a *cryptographic key* to encrypt and decrypt (scramble and unscramble) messages.

A Simple Cipher: The Caesar Cipher

Consider the first and perhaps simplest cipher: the *Caesar cipher*. Here, each letter in the message is shifted by an agreed-upon number of letters in the alphabet. For example, suppose you wanted to encrypt the *plaintext*

```
IF VOTING CHANGED ANYTHING IT WOULD BE ILLEGAL
```

by shifting each letter in the message forward by three places in the alphabet, so that `A` becomes `D`, `B` becomes `E`, and so on, with `Z` wrapping around to the start of the alphabet to become `C`. The plaintext gets encrypted to the following *ciphertext*:

```
LI YRWLQJ FKDQJHG DQBWKLQJ LW ZRXOG EH LOOHJDO
```

To decrypt this message, the recipient would do the reverse, shifting each letter in the message backward three places in the alphabet, so `Z` becomes `W` and `A` wraps around through the end of the alphabet to become `X`. For the recipient to be able to decrypt the message (quickly), they would

have to know the *key* to the cipher. For the Caesar cipher, this is the number of places that each letter is shifted in the alphabet; in this example, it is the number `3`. A Caesar cipher key can also be represented by a letter of the alphabet corresponding to the result of the translation from `A`. For example, a shift of `3` would be the key `D`, a shift of `23` would be the key `Z`, and the shift of zero (the identity shift) would be the key `A`.

Let's review the terms. In this example, to apply the *cipher* (or *encryption protocol*), one must simply follow these instructions: "To encrypt, shift each letter in the plaintext message forward in the alphabet by *n* letters. To decrypt, shift each letter in the message ciphertext backward in the alphabet by ***n*** letters." The *key* is the amount of the shift, ***n***.

Of course, the Caesar cipher is not a strong cipher, and you certainly shouldn't trust it to keep your plans secret. All an adversary would need to do to *break* (or *crack*) your secret code (ciphertext) is to try every possible backward shift through the alphabet. There are not many possibilities, so this wouldn't take long: since the key `A` makes the ciphertext equal the plaintext, there are only twenty-five possible keys. Such an *attack* is called a *brute-force attack*, in which an adversary attempts to decipher an encrypted message by trying every possible key. This attack is feasible in the case of the Caesar cipher because there are very few possible keys.

A Slightly More Complicated Cipher: The Vigenère Cipher

The Vigenère cipher is a set of Caesar ciphers, each with its own key. Typically the key is given as a word, and the position of the word's letter in the alphabet indicates how the letter `A` is shifted, as in a Caesar cipher. This is easiest to see with an example. Suppose you wish to encrypt the plaintext

```
RESPECT EXISTENCE OR EXPECT RESISTANCE
```

with the key

```
ACT
```

Then

- Encrypt every third letter starting with the first letter of the plaintext (`R`, `P`, `T` ...) with a Caesar cipher that maps `A` to `A` (a shift of zero, or a Caesar cipher with the key `A` or `0`).
- Encrypt every third letter starting with the second letter of the plaintext (`E`, `E`, `E` ...) with a Caesar cipher that maps `A` to `C` (a Caesar cipher using the key `C` or `2`).
- Encrypt every third letter starting with the third letter of the plaintext (`S`, `C`, `X`) with a Caesar cipher that maps `A` to `T` (a Caesar cipher using the key `19`).

Applying these three Caesar ciphers results in the ciphertext:

```
RGLPGVT GQIUMEPVE QK EZIEEM RGLIUMAPVE
```

To break this cipher, suppose your adversary knows the length of your key: your adversary would try to decrypt the ciphertext with all possible three-letter words (or, in general, any three-letter sequence of letters) of that length. In this example, that would require at most `25 × 26 × 26 = 16,900` attempts, which is more than could be easily attempted by hand but is trivially done by a computer. If your adversary doesn't know the length of your key, then they would have to try many more possible keys (as many as `25 + 25 × 26 + 25 × 26 × 26 + ...`) to apply this brute-force method to break the encryption. Notice that the longer your key is, the more difficult brute-force methods are—and the harder an adversary must work to break the encryption.

In Context: The Unbreakable Onetime Pad

A Vigenère cipher—whose key is a sequence of *randomly* selected letters and is at least as long as the message plaintext—makes possible a cipher known as the *onetime pad*. Historically, the key itself would be written on a pad of paper and distributed among communicating parties. To encrypt, a Vigenère cipher is applied to the plaintext, where each letter in the onetime pad is used only once before proceeding to the next letter and so on. Decryption relies on possession of this onetime pad, and the starting position in the key. It is *impossible* to break this cipher without the key—that is, it is impossible to guess the key and crack the ciphertext, even with unlimited time and resources. This is because a ciphertext of a given length could correspond to any plaintext of the same length. For example, without knowledge of the random key, the onetime pad-encrypted ciphertext `SOU DUCYFUK RXL HQKPJ` could (with equal probability) correspond to either the plaintext `ALL ANIMALS ARE EQUAL` or `FEW ANIMALS ARE HAPPY`. Without the key, there is no way to know what the intended (plaintext) message is! Omitting spaces between words or encrypting the spaces between words (using a twenty-seven-letter alphabet `ABCDEFGHIJKLMNOPQRSTUVWXYZ_`, where `_` is a space) would make it far more difficult to guess even the set of possible plaintext messages.

Of course, the onetime pad has the practical problem of how to exchange the key (the onetime pad itself), which is as long as the message, or as long as the total length of all possible future messages. Despite that, it has been used historically, with groups sharing a onetime pad in person and then sending messages over insecure channels. In the late 1980s, the African National Congress (ANC), at the time fighting apartheid in South Africa, used onetime pads to encrypt messages between foreign supporters and in-country operatives. The onetime pads (the keys) were physically transported by a trusted air steward who worked the Amsterdam-to-Johannesburg route. Incidentally, the ANC also computerized the encryption and decryption, making it possible to translate encrypted messages into tonal sequences transmitted over a phone connection and recorded to—or received from—an answering machine, allowing for *asynchronous* communication.

What to Learn Next

- Modern Cryptography

External Resources

- Jenkin, Tim. "Talking with Vula: The Story of the Secret Underground Communications Network of Operation Vula." *Mayibuye: Journal of the African National Congress*, October 1995.

Modern Cryptography

We recommend that you read the chapter "What Is Encryption?" before reading this chapter.

What You'll Learn

1. What key length means for security
2. What open-source software is and why it is important to security

Modern cryptography is not something you do by hand. Computers do it for you, and the details of the algorithms they employ are beyond the scope of this book. However, there are certain principles that will help you better understand and evaluate modern digital security tools.

Security through Requiring Brute-Force Attacks

Modern cryptographic protocols are designed to force an adversary (not in possession of the cryptographic key) to spend (close to) as much time as it would take to try every possible key to break the code. Recall that trying every possible key is known as a *brute-force attack*. The parameters of a given protocol are chosen so that this amount of time is impractical. Usually, the most important parameter is the length of the key. Just as with the classic Vigenère cipher, longer keys mean that more possible keys must be explored in order to guess the correct key. As time goes by and computer processing becomes faster and more powerful, often longer keys are required to guarantee that a brute-force attack would be infeasible. For this reason, many cryptographic protocols will mention the key size in terms of the number of bits it takes to represent the key.

Computers represent information, including cryptographic keys, in binary—using just 0s and 1s. Just like the numbers 0 through 9 represent the *digits* of a decimal number, the numbers 0 and 1 represent the *bits* of a binary number. How many three-digit decimal numbers are there? $10 \times 10 \times 10 = 10^3 = 1000$—that is, the numbers 0 through 999. Likewise, there are $2 \times 2 \times 2 \times 2 = 2^4 = 16$ four-bit binary numbers.

As an example, the AES cryptographic protocol may be referred to as AES-128 or AES-256 when using the protocol with 128-bit or 256-bit encryption keys, respectively. In AES-128, there are 2^{128}

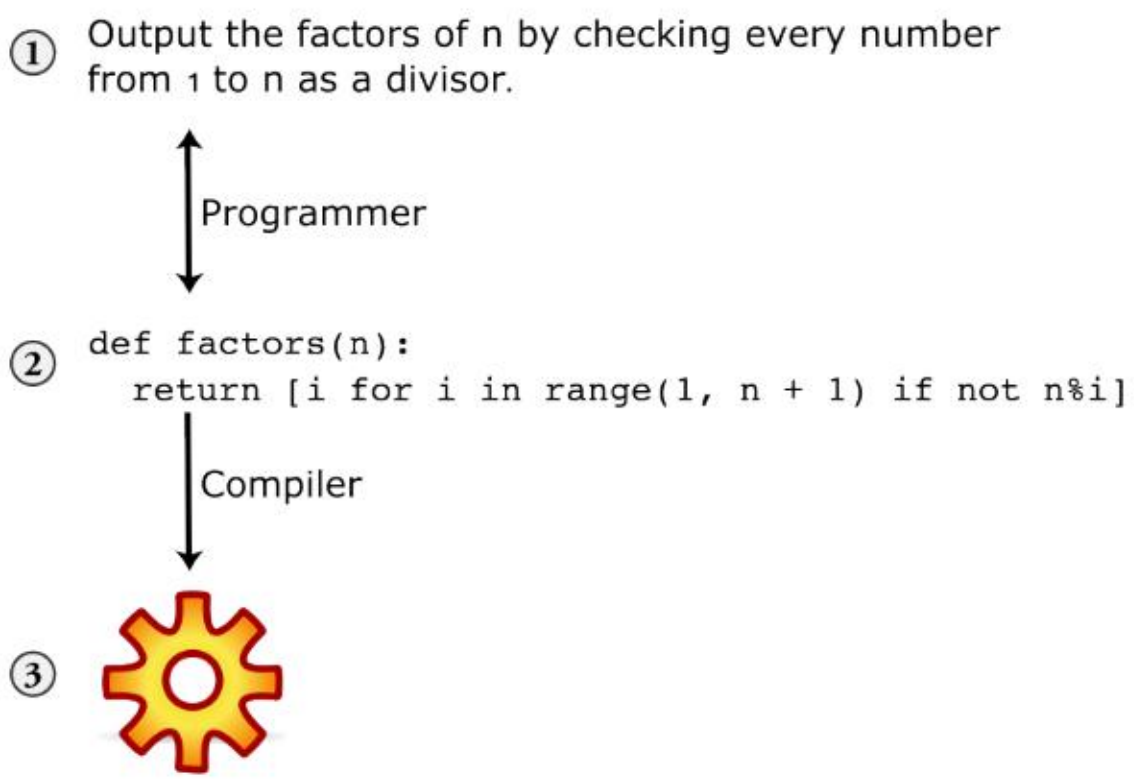
= 340282366920938463463374607431768211456 possible keys. In AES-256, there are 2^{256} = 115792089237316195423570985008687907853269984665640564039457584007913129639936 possible keys. Trying every possible key—or even a small fraction of all possible keys—for AES-256 is computationally infeasible, even given the computational power of nation-states such as the United States.

Security Is Not Guaranteed through Obscurity

Since as early as the nineteenth century, mathematicians have held as a standard that cryptographic schemes should be secure even if the method being used is not secret. This is based on the following principle: If security requires keeping the method secret, then one risks all messages that have ever been encrypted or ever will be encrypted with that method being revealed if the method is ever uncovered. On the other hand, if your method only requires keeping the key secret, then one only risks those messages that have been encrypted with that particular key being revealed if the key is compromised.

Security Is Provided by Transparency

In fact, the more transparency around a cryptographic method, the more you can trust the security of the method. To understand this, consider how an encryption program (or any computer program, in fact) is created. It starts with an algorithm as to how to perform the encryption. A programmer turns this algorithm into a *source* computer code. A computer compiles this source code into the program or app that runs on your computer or phone.



A good computer programmer should be able to translate from an algorithm (1) to source code (2) and back. A security professional would be able to evaluate the security of a cryptographic protocol based on the algorithm but should also evaluate the source code to be certain of its faithful implementation (that there are no mistakes or *bugs*, whether intentional or not). However, as a user, you would only have access to the compiled program (3). Unfortunately, given only the compiled code, it is impossible for anyone to re-create the source code.

So unless the source code is available, no one can be certain that the security claims of an app are true. On the other hand, having just the compiled program is enough for a hacker to try to break the security of the app. Many software projects make their source code available to the public: such software is called *open-source software* and includes many well-known projects, security and otherwise, such as Signal, Firefox, and Linux. The alternate is *closed-source software* and is popular among projects that aim to monetize their product through sales of proprietary software, such as Safari, Internet Explorer, Windows, and Mac OS. While it is possible to evaluate the security of closed-source software (e.g., through private audits), it is much more difficult to maintain this on an ongoing basis. Open-source projects are open to scrutiny by anyone, giving every opportunity for security (or other) problems to be discovered.

Security Is Provided by Protecting Your Encryption Key

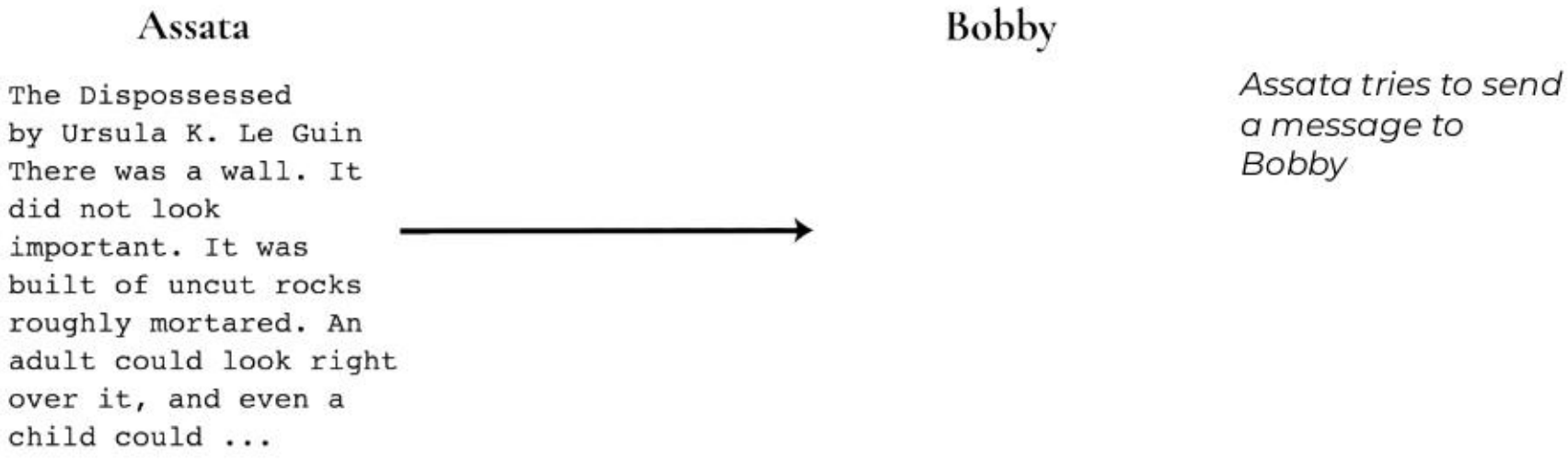
Since the encryption method is typically public in modern cryptographic protocols, the way that one achieves security is through protecting their encryption key. What this looks like in practice depends on where the key resides. In the case of Signal, a secure instant messaging app, the encryption key is a file on your phone, and it protects your phone. In the case of a password manager that syncs your passwords to the cloud, the key that encrypts the file storing all your passwords is derived from or protected by the password that you use to log into your password manager.

Security Is Provided by Distrusting the Infrastructure

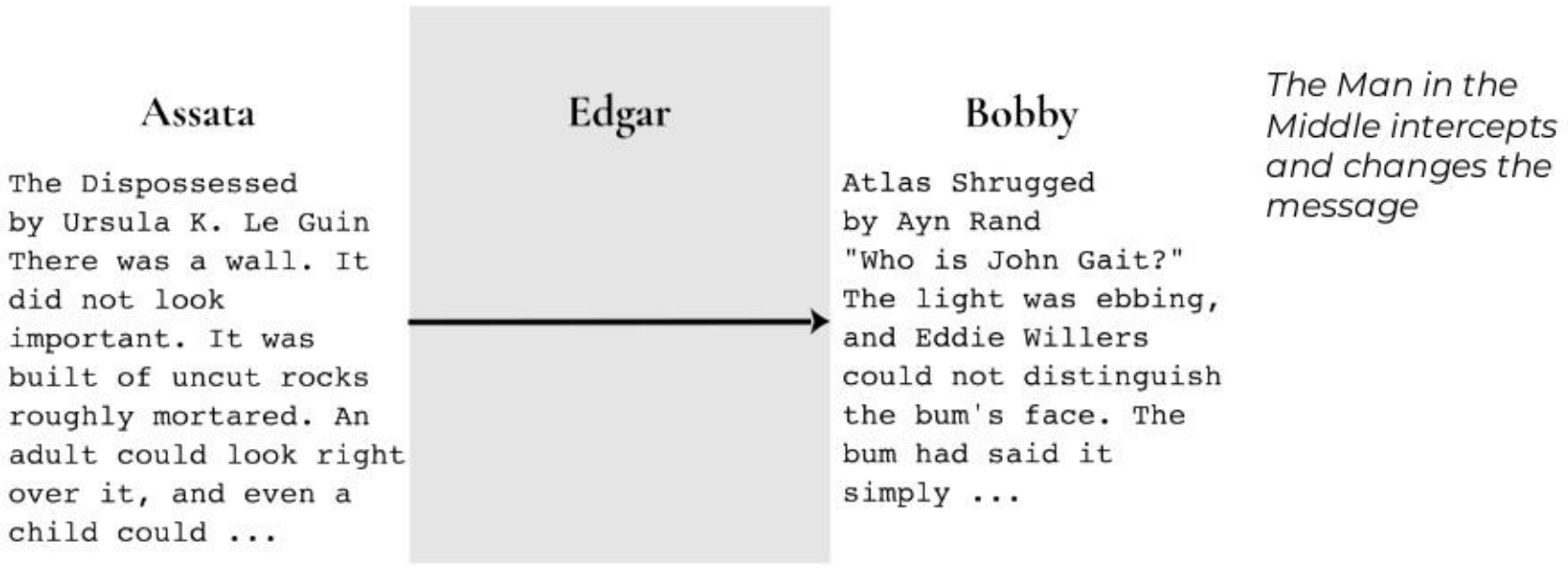
End-to-end encryption involves scrambling a message so that it can only be read by the endpoints of a conversation. But here's where the confusion comes in: What are the endpoints? Are they just you and your friends? Or is the server an endpoint too? It depends on the application. As an example, https (which secures communications between you and the servers hosting the web pages you visit) is encrypted so that only you and the server can decrypt the content of the web pages. Signal encrypts messages so that only you and your friend that you are messaging can read them. In both cases, only the people or entities that need to know the information are able to decrypt the information. This is the heart of end-to-end encryption.

Here is an illustration of why end-to-end encryption is so important in private messaging. This is covered in greater technical detail in the chapter "The Man in the Middle." In the following figure,

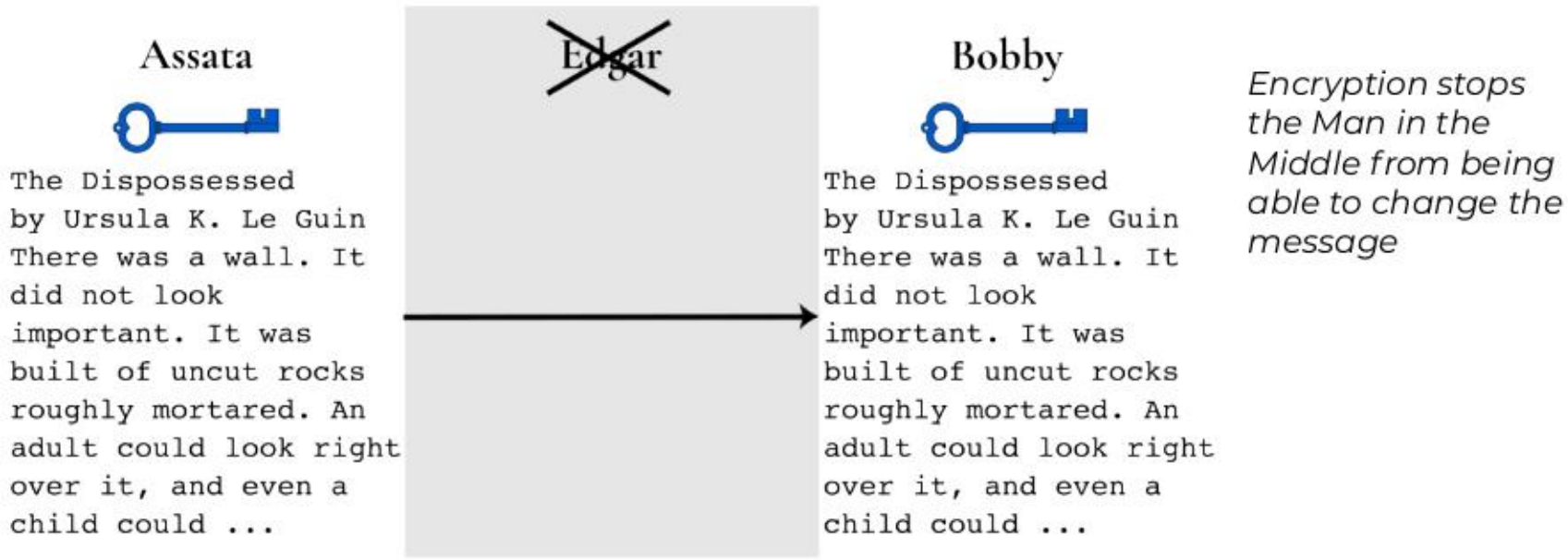
Assata (left) is trying to get a message (Ursula K. Le Guin's *The Dispossessed*) to Bobby (right) over the internet:



But the ghost of mean old J. Edgar Hoover haunts the infrastructure. This Man in the middle here is able to intercept, read, and change any unprotected message sent between our two friends. Like so:



(Edgar could also just read and send the message along unaltered.) To make matters worse, saying that an app uses "encryption" (without being specific about who holds the keys) doesn't guarantee that messages remain private and authentic. For example, if a server between the two comrades is managing the encryption keys, anyone with access to the server could read and modify all messages between them. However, if Assata and Bobby are encrypting their message (with the blue key), then Edgar won't be able to read the message and wouldn't be able to replace the message with one that can be decrypted with the blue key:



Encryption stops the Man in the Middle from being able to change the message

How do you know whether an application uses end-to-end encryption? The best indication is that there is some way to verify encryption keys—Signal makes this easy with safety numbers. We will describe this in more detail in the chapter "Authenticity through Cryptographic Signing."

Another way to reduce exposure to a malicious interloper is through peer-to-peer messaging, where it is said that there is "no server" in between managing your messages or contacts. Even this can be a bit misleading, however: there is a tremendous amount of internet infrastructure in between you and your friends; it's just invisible to most users and apps. As described above, this infrastructure is precisely what the State exploits to conduct undetectable, suspicionless mass surveillance.

In Context: The Enigma Machine

Possibly the first modern encryption techniques were used during World War II. Predating modern computers, the protocols were supported by sophisticated mechanical devices. Most notable among these is the Enigma machine used by Nazi Germany. The Enigma is an electromechanical device that allowed you to set a particular key, type in the plaintext, and get the ciphertext output. With the same key, typing in the ciphertext would output the original plaintext.



Enigma machine, courtesy of Greg Goebel

The key is an order of the rotors and initial positions of the rotors (pictured above). Standard operation required using a new key every day. The keys were listed by day in handbooks distributed to operators of Enigma machines—these are essentially onetime pads of keys. Incidentally, these were printed with water-soluble ink, allowing quick destruction of the key book when at risk of falling into enemy hands.

Much effort went into breaking Enigma-encrypted messages. Several machines were captured during World War II, but even in possession of the machine, decrypting messages was nearly infeasible (as with truly modern ciphers whose methods are public). Alan Turing, one of the founders of computer science as a discipline, worked at the secretive Bletchley Park, the central site for British code breakers during World War II. Turing designed the *bombe*, a type of computer specially designed for deciphering Enigma messages. The bombe was not enough. (In fact, decrypting Enigma messages without a key is incredibly challenging even with modern computation capabilities; at least one famous Enigma message intercepted during the war remains encrypted to this day.) However, the bombe in combination with the fact that most early morning messages contained weather reports or the phrase *Keine besonderen Ereignisse* ("nothing to report") did allow the Allies to break Enigma enciphered messages regularly.

Turing's work during the war has been estimated to shorten the war by more than two years. However, his work remained unacknowledged throughout his life, since work at Bletchley Park was classified, and in fact, he was criticized for not contributing to the war effort. More tragically, as a gay man, he was persecuted by his own government to the point of being charged with a crime in 1952. Found guilty of homosexual acts, he was given the choice of chemical castration or imprisonment. Choosing the former, he only lived another two years, reportedly ending his own life by cyanide poisoning.

What to Learn Next

- Exchanging Keys for Encryption

External Resources

- Caraco, Jean-Claude, Rémi Géraud-Stewart, and David Naccache. "Kerckhoffs' Legacy." 2020.

Media Attributions

- source-code © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- mitm-basic-1 © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- mitm-basic-2 © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- mitm-basic-3 © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- Four-rotor-enigma © Greg Goebel is licensed under a Public Domain license

Exchanging Keys for Encryption

We recommend that you read the chapter "Modern Cryptography" before reading this chapter.

What You'll Learn

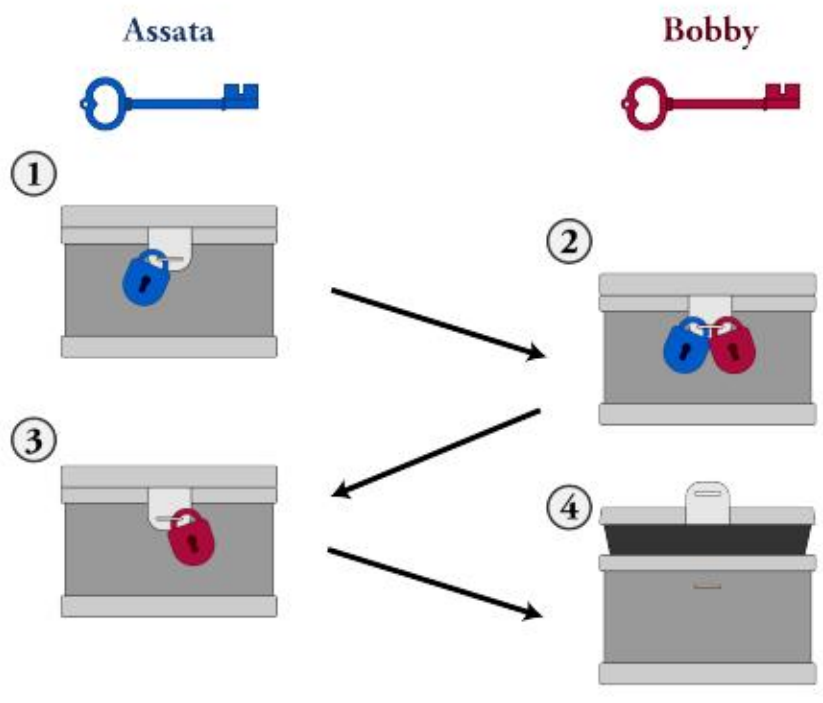
1. How messages can be encrypted without sharing an encryption key in advance
2. The primary method of exchanging keys online used today

Eavesdropping communications through the internet can be done at many points: the Wi-Fi hotspot you're directly connected to, your internet service provider, the server hosting the web pages you visit, national gateways, and the vast array of routers and switches in between.

Without encryption, all these communications would be readable by an eavesdropper, be that a stalker, a hacker, or a government agency. But in order to encrypt your communications, you need to agree on a key with the party you are communicating with. If you are visiting a website, how do you safely exchange a key with the server that hosts the website? We need a method for two parties (e.g., two people, a person and a server, or two servers) to efficiently agree on a key without meeting and while only being able to communicate over insecure channels, such as the internet.

A Physical Example: Exchanging a Message without Exchanging a Key

First consider a physical example, illustrated below. Suppose Assata wants to send Bobby a package. She puts it in a strong box with a large clasp that can take multiple locks (1). She puts a lock on the box, but Bobby doesn't have a key to the lock. Assata mails the box to Bobby, who cannot open it (and neither can anyone else while the box is in transit). Bobby puts his own lock on the box (2), a lock that Assata doesn't have the key to. When Assata receives the box, she removes her lock and sends the box back to Bobby (3). Now Bobby can open the box because it is only secured with his lock (4). The box cannot be opened in transit—an eavesdropper would have to break Assata's lock, Bobby's lock, or both.



Exchanging a secure message without sharing a key

This illustrates that it is possible to send something securely without meeting first to exchange (agree on) a key. However, we aren't about to start physically mailing lockboxes in order to exchange encryption keys. What we need is a mathematical version of this that we can use for digital communications.

A Mathematical Example: Exchanging a Message without Exchanging a Key

Let's see how we would do this without physical boxes and locks. Suppose you have an encryption protocol where you can encrypt any text (as we always expect), that you can apply multiple times for *layers* of encryption (as we also always expect), and that you can encrypt and decrypt the layers in any order you wish and end up with the same result. A mathematical operation satisfying this last property is said to be *commutative*. (All the encryption protocols we describe in the chapter "What Is Encryption?" are commutative.) Let's see this with an example, using the Vigenère cipher.

Assata encrypts the message

```
AT ONE TIME IN THE WORLD THERE WERE WOODS THAT NO ONE OWNED
```

with a Vigenère cipher and key `ALDO` to get the ciphertext

```
AE RBE ELAE TQ HHP ZCRWG HHPUS WPUS WZRRS EKOT YR CNP RKNPG
```

and sends the result to Bobby. Bobby doesn't have the key! But Bobby encrypts this ciphertext with a Vigenère cipher and key `LEOPOLD` to get the doubly encrypted text

`LI FQS POLI HF VSS KGFLU SKAYG LDFV HDFGG PNZX MG QYS COBEU`

and sends the result back to Assata. Assata "decrypts" the message from Bobby with her key (`ALDO`) to get (the still encrypted message)

`LX CCS ELXI WC HSH HSFAR EKPVS LSCH HSCSG EKLX BD CYH ZABTR`

and sends the result to Bobby. Finally, Bobby decrypts this with his key (`LEOPOLD`) and gets the message that Assata wanted to send Bobby in the first place:

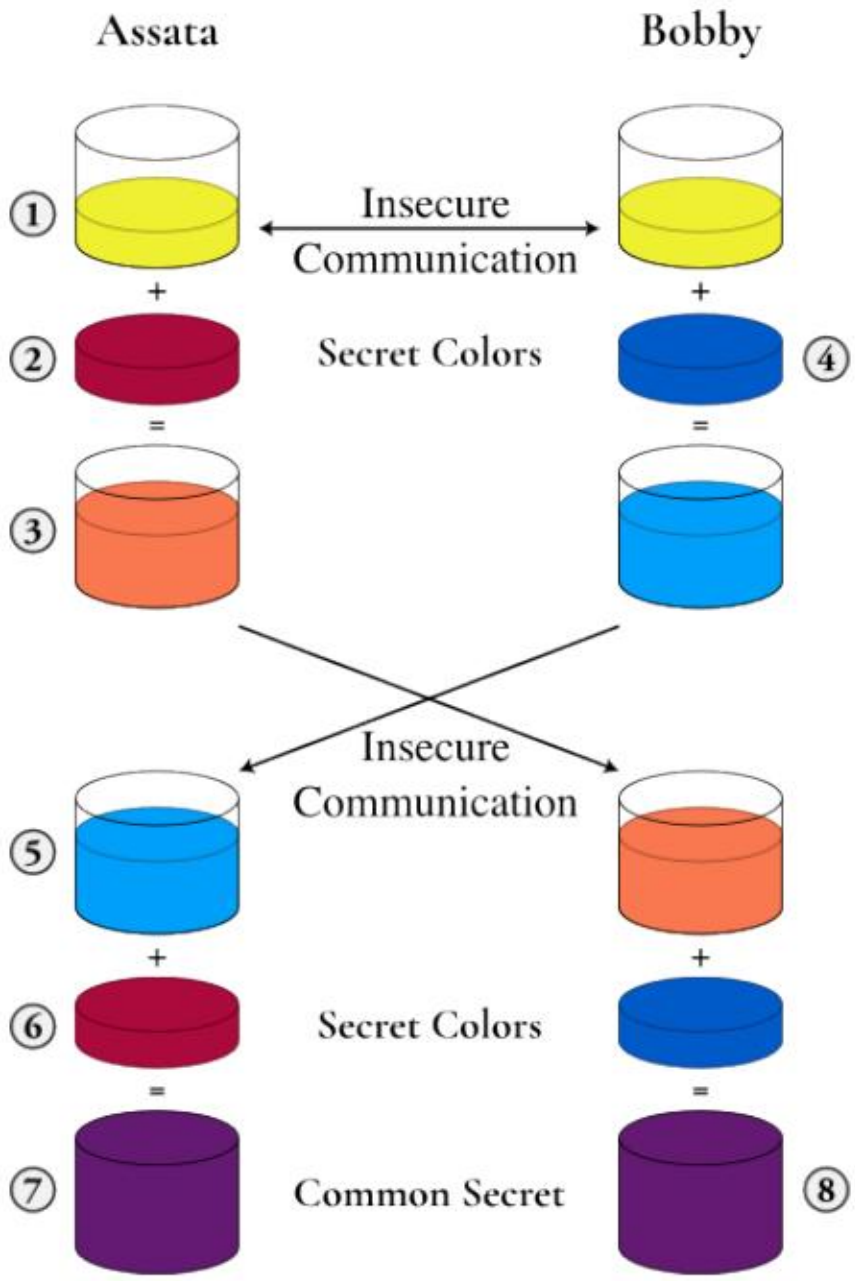
`AT ONE TIME IN THE WORLD THERE WERE WOODS THAT NO ONE OWNED`

Note that, in this example, Assata did not share her key (`ALDO`) with anyone, and Bobby did not share his key (`LEOPOLD`) with anyone either. Because the Vigenère cipher is commutative, it did not matter that the message was encrypted with Assata's key, then encrypted with Bobby's key, then decrypted with Assata's key, and finally decrypted with Bobby's key. All that matters is that the message was encrypted and decrypted once with each key. Any eavesdropper would only see one of the three intermediate ciphertexts.

A Physical Example: Agreeing on a Secret over an Insecure Channel

In modern cryptographic systems, rather than sending the entire message back and forth with different layers in this way, one has an initial exchange, much like in the above examples, to settle on a key to use for the intended communication. You could imagine that Assata, rather than sending the message `AT ONE TIME IN THE WORLD...`, sent an encryption key to use for a longer communication. We will describe the mathematical basis for key exchange as it is used by almost all modern communication, called the *Diffie-Hellman key exchange*.

First, let's see how this is done with paints instead of mathematics (illustrated below). We will assume that if you mix two colors of paint together, you can't unmix them; specifically, even if you know what one of the two colors was, you can't figure out what color was mixed with it to get the resulting mixed color.



Crafting a shared secret, in paints

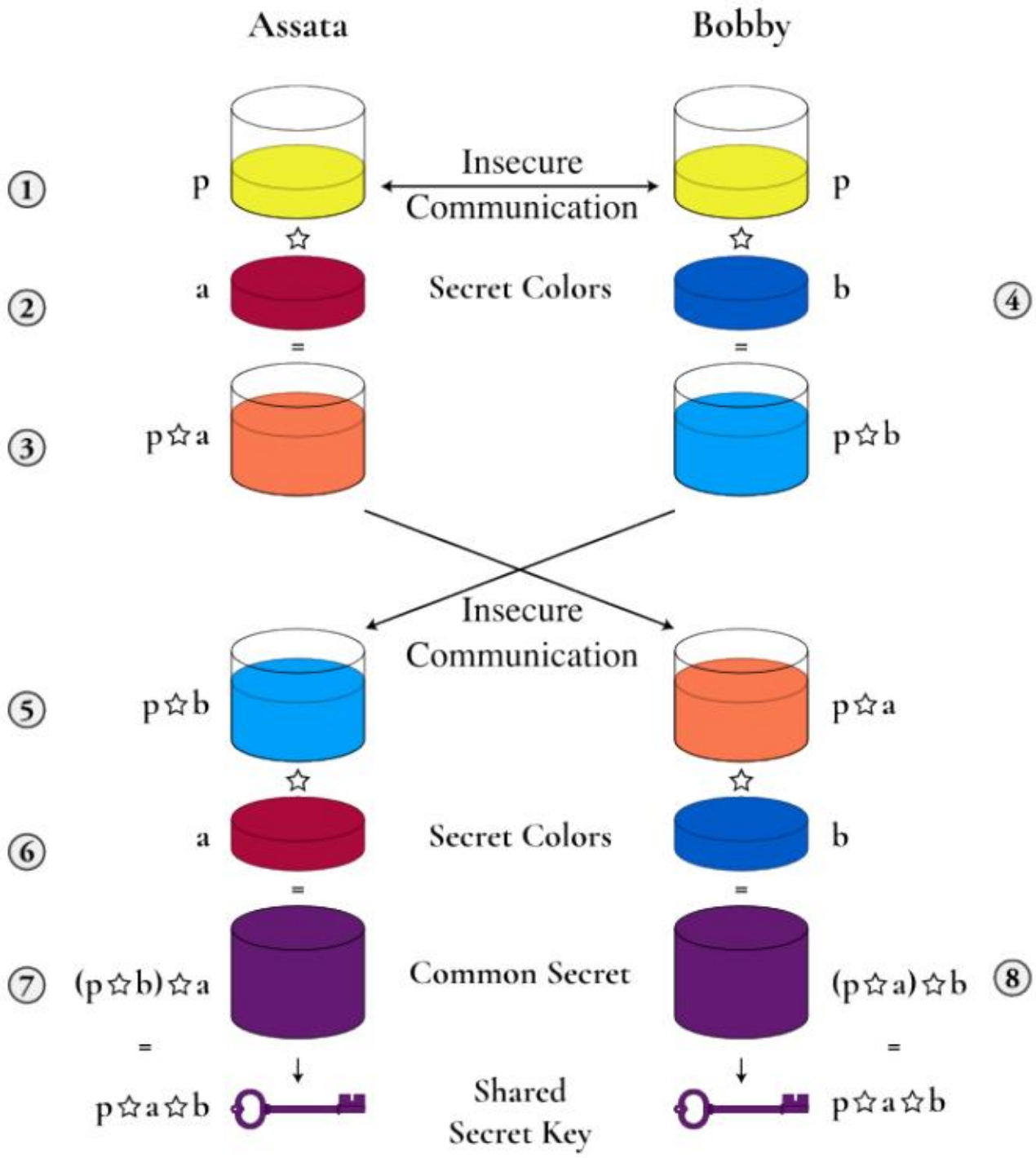
Assata and Bobby start by agreeing on one paint color (in this example, yellow) and an amount, say 10 mL (1). They can do this over an insecure communication channel and should assume that an eavesdropper will know what the color and amount are too. Then Assata picks a color (in this case, rusty orange) and keeps it secret (2). She mixes 10 mL of yellow with 10 mL of rusty orange to get a coralish color (3). She sends this to Bobby over the insecure channel, understanding that an eavesdropper will see it. Bobby does the same thing, with his own secret color (4).

Now to the paint sample received from Bobby (5): Assata mixes in 10 mL of her secret color (6), resulting in a dark purple (7). Bobby does the same thing. Assata's unpleasant brown–dark purple is obtained from a mix of 10 mL each of yellow, her secret color, and Bobby's secret color. Bobby's resulting paint mix is obtained from a mix of 10 mL of yellow, his secret color, and Assata's secret color. So Bobby also ends up with the same unpleasant brown–dark purple (8)! Can the eavesdropper create the dark purple? The eavesdropper sees yellow (1), the mix of yellow and Assata's secret color (3), and the mix of yellow and Bobby's secret color (5). But to create the unpleasant brown, the eavesdropper would have to unmix in order to obtain Assata's or Bobby's secret colors, which they can't do.

Diffie-Hellman Key Exchange

Let's revisit this process mathematically. We do so with a commutative mathematical operation that is hard or impossible to reverse. A mathematical operation or function that is hard to reverse is called a one-way function. Let's represent our mathematical operation with the symbol ☆—that is, a ☆ b = c for some numbers *a*, *b*, and *c*. Commutative means that a ☆ b = b ☆ a. That ☆ is one way means that if you know *b* and *c*, you cannot easily figure out what *a* is. In practice, one should only be able to figure out what a is by a brute-force (or close to brute-force) attack: by trying every possibility for a. You may think of ☆ as the multiplication sign (which is commutative but is not one way). (For those mathematically inclined, ☆ can be modular exponentiation for real implementations of Diffie-Hellman.)

Illustrated below, Assata and Bobby agree on a number *p*, which is public (1). Assata chooses a secret number *a* (2), computes *p* ☆ *a* (3), and sends the result to Bobby. Since ☆ is one way, an eavesdropper will know *p* and *p* ☆ *a* but will not be able to (easily) determine *a*. Bobby chooses a secret number *b* (4), computes *p* ☆ *b*, and sends the result to Assata (5). An eavesdropper knows *p* ☆ *b* but not *b*. Assata computes *(p* ☆ *b)* ☆ *a* (7), using the message from Bobby (5) and her own secret number (6). Bobby computes *(p* ☆ *a)* ☆ *b* (8), using the message from Assata (3) and his own secret number (4). Since ☆ is commutative, *(p* ☆ *b)* ☆ *a = (p* ☆ *a)* ☆ *b*, and so Assata and Bobby now have computed a common number. Since the eavesdropper only knows *p* ☆ *a*, *p* ☆ *b*, and p, and since ☆ is one way, the eavesdropper has no efficient means of computing Assata and Bobby's shared common number: it is secret to Assata and Bobby. Assata and Bobby can use this shared number as their cryptographic key.



Using Diffie-Hellman Key Exchange

Diffie-Hellman key exchange is used *all over the place* as a means of agreeing on a cryptographic key. It is used as the basis for most forms of encrypted communications that you will encounter. Most notably, it forms the basis of key exchange when you connect to a website via https. When you visit a website, the URL will either start with http:// or https://. In the former case, none of your communications with the server of the website are encrypted. In the latter, communications are encrypted, and the key used to encrypt those communications is generated using Diffie-Hellman key exchange.

In Context: When Good Things Go Bad

Remember that the first thing that Assata and Bobby do is agree on a number *p* that forms the basis of their key exchange. This number is public, but we assumed that our mathematical operation

☆ was one way, so it was OK for *p* to be public. However, someone with a lot of computational resources (such as a wealthy nation-state) can invert the operation ☆ (for functions such as modular exponentiation used for ☆ in the real world) using two phases. The first phase takes a very long time and must be done for a specific value of *p*. The second phase can be done very quickly (in real time) for the same value of *p*, assuming that the first phase has been completed. This means that everyone *should not* be using the same value *p* but should be using different values of p and changing them often.

However, in 2015, researchers showed that 18 percent of the top one million https domains use the same value of *p*. Two other communication protocols that depend on Diffie-Hellman key exchange are SSH (secure shell) and VPN (virtual private network). The same researchers showed that 26 percent of SSH servers and 66 percent of VPN servers used the same value of *p* in their Diffie-Hellman key exchange. This means that a powerful adversary would have little trouble breaking the encryption.

While the Diffie-Hellman protocol is strong and reliable, this highlights that those who implement the protocols need to do so with care to ensure that they are in fact secure.

What to Learn Next

- The Man in the Middle
- Public-Key Cryptography

External Resources

- Adrian, David, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, et al. "Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice." In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 5–17. Denver: ACM, 2015.

Media Attributions

- lockbox © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- diffie-hellman-concept © Lorddota adapted by OSU OERU is licensed under a CC BY-SA (Attribution ShareAlike) license

- diffie-hellman © Lorddota adapted by OSU OERU is licensed under a CC BY-SA (Attribution ShareAlike) license

Cryptographic Hash

We recommend that you read the chapter "Modern Cryptography" before reading this chapter.

What You'll Learn

1. What a hash function does
2. What a cryptographic hash function does and how it is distinct from an ordinary hash function
3. Some examples of the use of cryptographic hash functions

A *hash function* is any (computer) function that transforms data of arbitrary size (e.g., a name, a document, a computer program) into data of fixed size (e.g., a three-digit number, a sixteen-bit number). The output of a hash function is called the digest, fingerprint, hash value, or hash (of the input *message*).

A *cryptographic hash* function has the following properties that make it useful for cryptographic applications:

1. The same message always results in the same output hash.
2. It is infeasible to generate the input message from its output hash value except by brute force (trying all possible input messages).
3. It is infeasible to find two different input messages that result in the same output hash value.
4. A small change to the input message changes the output hash value so extensively that the new hash value appears uncorrelated with the old hash value.

The first two of these properties are similar to most encryption protocols. If you encrypt the same message on two different occasions, you would expect the same result, assuming you are using the same encryption key. Given only the ciphertext, it should be infeasible to generate the plaintext (without the decryption key). However, encryption allows you to go backward, from ciphertext to plaintext, using the decryption key. Hash functions are inherently one way: there is no key to go backward. That the result is sometimes called a digest or a fingerprint is a useful analogy: while the output of a cryptographic hash function does not encode all the information of the input message (in the way that a ciphertext does), it encodes enough information that you can use it to identify the input (relying on properties 1 and 3) and that this is very difficult to fake (property 2).

We will see applications of cryptographic hash functions in the chapters "The Man in the Middle,"

"Passwords," and "Public-Key Cryptography," but let's look at a simple use here, known as a *commitment scheme*.

Using Cryptographic Hash Functions to Prove How Smart You Are

Assata and Bobby are both trying to solve a difficult math problem. Assata gets the answer (S) first and wants to prove to Bobby that she has the answer before he has solved it without leaking the solution to him. So Assata takes a cryptographic hash of the solution S, hash(S), and gives Bobby hash(S). Since the hash is cryptographic, Bobby can't learn S from hash(S) (property 2). When Bobby eventually solves the problem, finding S for himself, he can compute hash(S) and check that the result is the same as what Assata gave him. By properties 1 and 3, Bobby knows that Assata's input to the hash function must be the same as his input to the hash function, thus proving that Assata solved the problem first. (Property 4 was not used here, but without this property, if Assata got a solution that was close to correct but not quite, the two outputs might be very similar, and a cursory comparison may not uncover that they are different.)

What Do Hash Functions Look Like?

There are many different cryptographic hash functions in use today, but describing them in detail is beyond the scope of this book. However, to give you a sense of what they might look like, we give an example that satisfies some, but not all, of the properties that cryptographic hash functions have.

The example hash function is called `chunked XOR`. Exclusive, or `XOR`, is a function that, when given a pair of inputs, outputs true (or 1) if the inputs are different and false otherwise. So, for example, `apple XOR banana = 1`, `apple XOR apple = 0`, `0 XOR 1 = 1`, `1 XOR 1 = 0`. We can take a chain of `XOR`s on binary numbers (0s and 1s) and get a meaningful answer: `1 XOR 1 XOR 0 = 0`, `1 XOR 1 XOR 0 XOR 1 = 1`. For a sequence of binary numbers, `XOR` returns `1` if there are an odd number of 1s in the chain and 0 otherwise.

`Chunked XOR` operates on a binary input. (If your input is not binary, you could represent it in binary first, the same way a computer would.) We group the input into chunks equal to the size of the output of the hash function—for example, groups of eight bits. We line the chunks up vertically, and then `XOR` the contents of each column, as illustrated below:

```
        input: 00111011 11101101 00101000 00101011 01011000 11001110

      chunked: 00111011
               11101101
               00101000
               00101011
               01011000
               11001110

XOR'd columns: 01010011 (output)
```

This is a hash function in that no matter what the length of the input, the output will always have the same length (eight in this example). You should be able to see that `chunked XOR` satisfies the first property of cryptographic hash functions. However, it fails on the remaining properties. It is easy to create *an* input message (but not necessarily your desired input message) with a given output hash—for example, you could concatenate 11111111 11111111 onto the result of the hash. For the same reason, you could create multiple messages having the same output hash. Finally, changing a single bit of the input message will change only a single bit of the output hash.

In Context: Cryptographic Hashes Violate Your Fourth Amendment Rights

In 2008, a US district judge ruled that if the US government wants to cryptographically hash your private data, then they need a warrant first. The case in question had a special agent of the Pennsylvania Office of the Attorney General copy the hard drive of a suspect's computer. The special agent computed a cryptographic hash of the copy (so that it could be later compared to the original to prove that they did not tamper with it, relying on properties 1 and 3). The agent then used a forensic tool that computed a cryptographic hash of each individual file (including deleted but not yet overwritten files) on the copied hard drive and compared these hashes to hashes of files in a database of contraband files. The agent found three matches between hashes of files on the hard drive and hashes of contraband files. By properties 1 and 3, this means that the hard drive contains at least three illegal files. The judge on the case determined this practice (of hashing the files and comparing them to known hashes) to constitute a *search* of the hard drive, violating the accused's Fourth Amendment rights to protection from illegal searches and seizures. As a result, the evidence could not be used in trial.

We should disclose the particulars of the case, which involves the possession of child pornography. While we would never defend the right of possession (or the creation or distribution) of child pornography, it is important to imagine a power (in this case, that of determining the existence of particular files on a computer) to be used in a way that you would *not* want it to be used. Music

that a friend shared with you? Images of oil spills? Images of #blacklivesmatter protests? *Earth First! Journal* articles?

What to Learn Next

- Passwords
- The Man in the Middle
- Authenticity through Cryptographic Signing

External Resources

- United States of America v. Robert Ellsworth CRIST, III, Defendant. Criminal Action No. 1:07-cr-211. 627 F.Supp.2d 575 (2008).

The Man in the Middle

We recommend that you read the chapters "Exchanging Keys for Encryption" and "Cryptographic Hash" before reading this chapter.

What You'll Learn

1. What an impersonation attack is
2. What a man-in-the-middle attack is
3. The difference between a passive and active man-in-the-middle attack
4. How to uncover man-in-the-middle attacks occurring during key exchange using fingerprinting

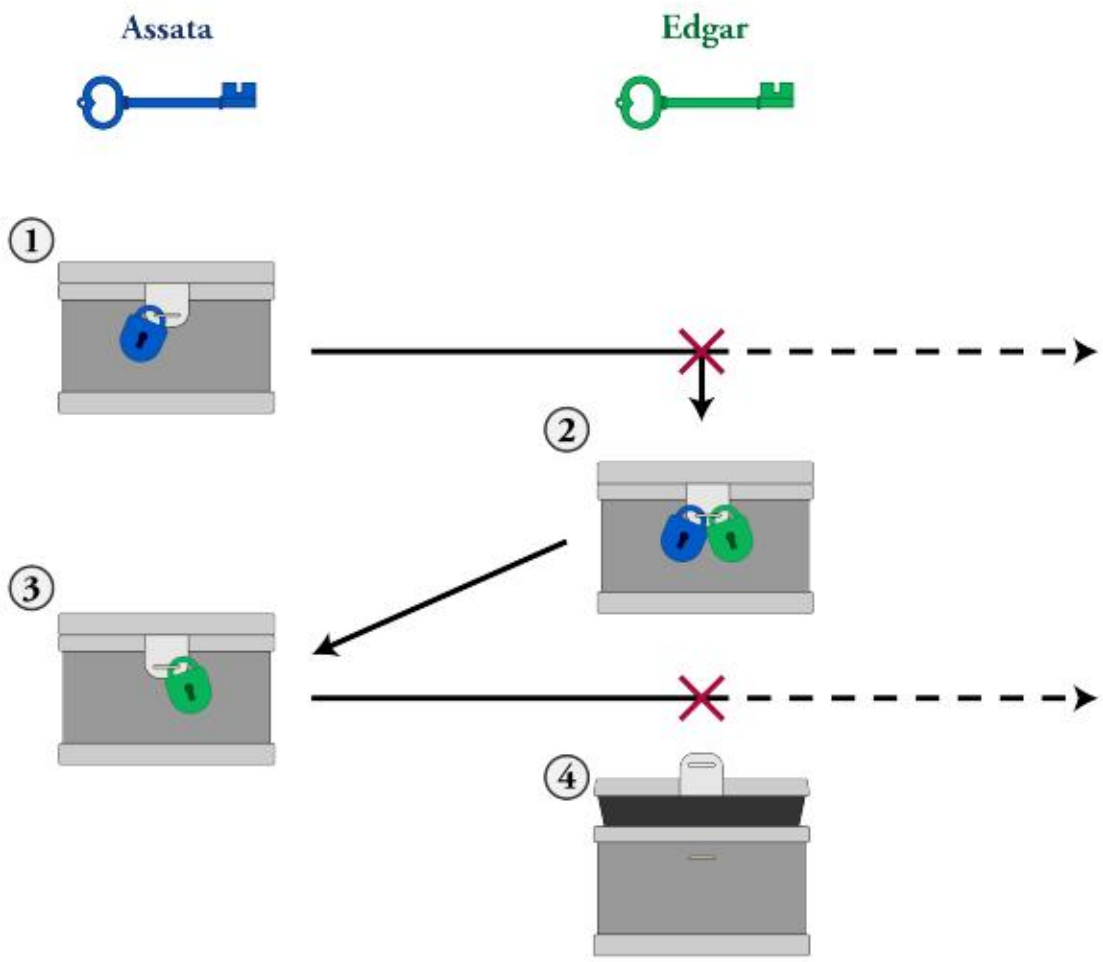
In the chapter "Exchanging Keys for Encryption," you learned how two people can agree on a cryptographic key, even if they have not met. While this is a robust method, it suffers from the limitation that on the internet, it is difficult to be sure that you are communicating with the person or entity you are trying to communicate with, be that a friend you are instant messaging or emailing or the server that you are trying to load a web page from. We will first show how an eavesdropper can intercept your communications with our lockbox example from the chapter "Exchanging Keys for Encryption" and then show how this plays out in a Diffie-Hellman key exchange. These interceptions of communications are called attacks.

A Physical Man-in-the-Middle Attack

Recall that Assata was able to send Bobby a secure package by sending a lockbox back and forth three times: once with her lock on it, once with Bobby's lock on it, and finally with her lock removed and only Bobby's lock on it. However, how does she know it is actually Bobby who receives the package? And how does she know that it is Bobby's lock on the box when it is sent back to her?

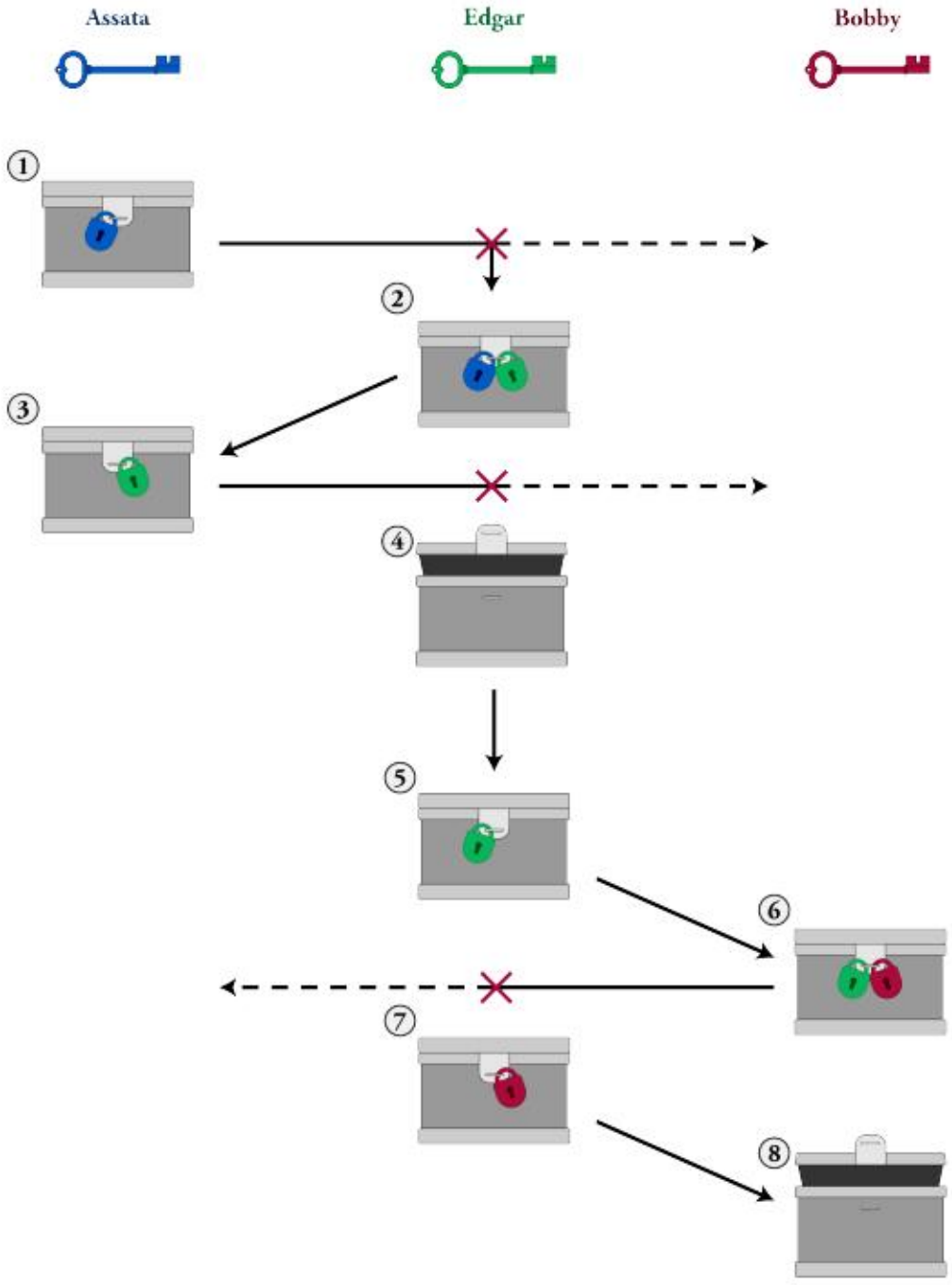
Illustrated below, suppose Edgar intercepts the lockbox from Assata to Bobby with Assata's lock on it (1). Edgar could send the lockbox back to Assata with his own lock on it (2). Unless Assata is able to tell the difference between a lock from Edgar and a lock from Bobby, Assata would assume that the lock is Bobby's lock, remove her lock, and send the package on to Bobby (3). If Edgar intercepts the package again, he can now open the box and examine the contents of the package, since it only has

his lock on it (4). For Edgar to do this, he must intercept all the packages being sent from Assata to Bobby. This attack on Assata's communication with Bobby is called an *impersonation attack*: Edgar is impersonating Bobby. (This is not generally considered a man-in-the-middle attack.)



An impersonation attack

In the situation as described, Bobby never received a package at all. Edgar could go further though (illustrated below). Edgar could, after opening the lockbox from Assata (4), choose to send it along to Bobby, using a mirror image of the same three-exchange method, so that Bobby thinks he is receiving a locked box from Assata (5–8).



A man-in-the-middle attack

Edgar passes along Assata's original message (just inspecting the message for himself), then we call this a *passive man-in-the-middle attack*. If Edgar substitutes the package with a completely different package, we call it an *active man-in-the-middle attack*. In either case, Edgar would need to intercept all packages between Bobby and Assata, as the packages will be addressed to Bobby or Assata, not Edgar.

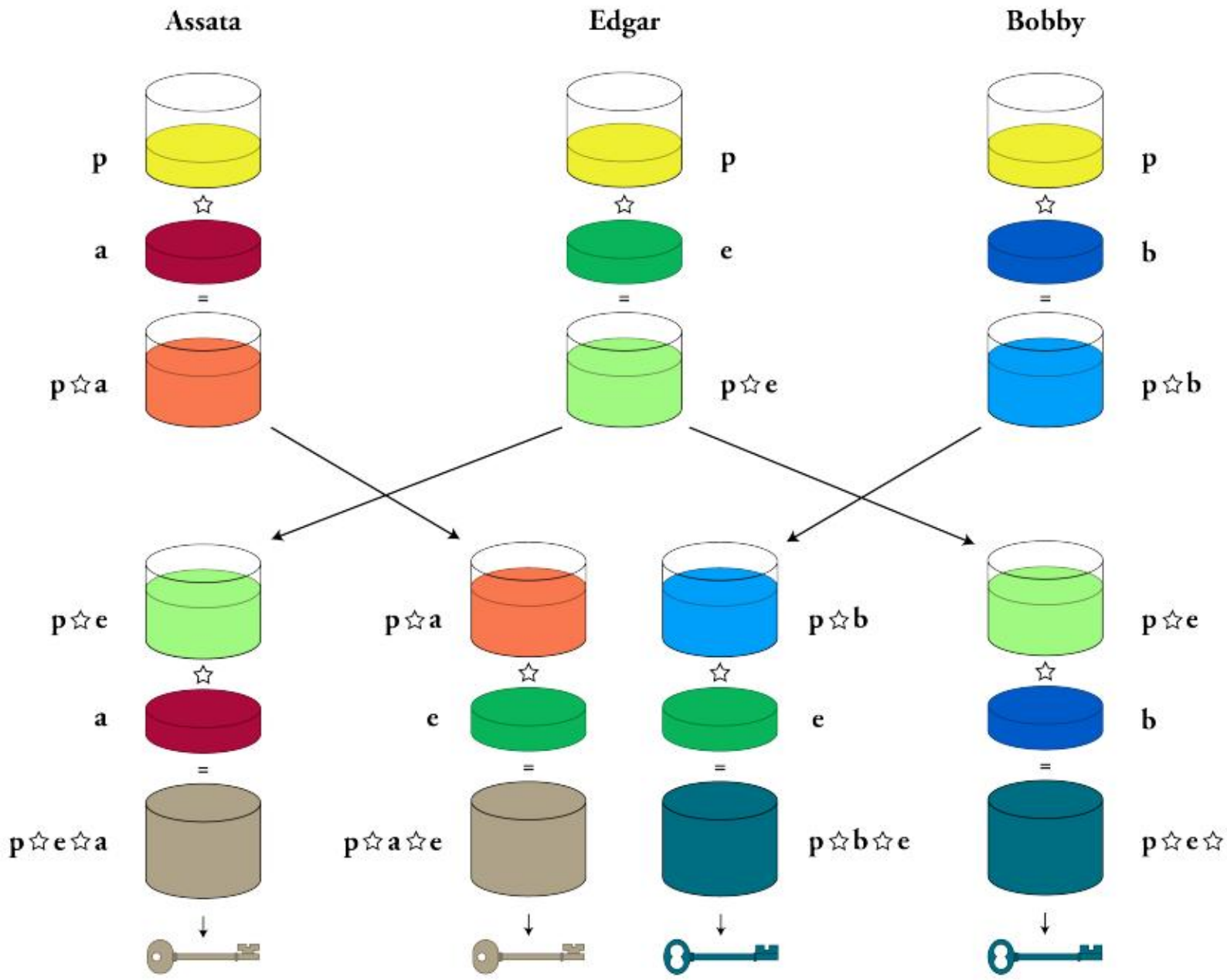
These types of attacks are called man-in-the-middle attacks because Edgar is the man in the middle of Assata and Bobby's communication. (In the case of J. Edgar Hoover, quite literally "the man.")

A Man-in-the-Middle Attack against Diffie-Hellman Key Exchange

Let's see how this plays out in the Diffie-Hellman key exchange, using the notation we introduced in the chapter "Exchanging Keys for Encryption." Recall that in order for Assata and Bobby to generate a key, they first agree on a number p. Assata picks a number a, computes $p \star a$, and sends the

result to Bobby. Bobby picks a number *b*, computes $p \star b$, and sends the result to Assata. Assata and Bobby (and no one else) can now compute $p \star a \star b$, which they use as their cryptographic key for their encrypted communication.

Suppose, though, that Edgar is able to intercept Assata's and Bobby's communications. Then Edgar can do one Diffie-Hellman key exchange with Assata and another Diffie-Hellman key exchange with Bobby, illustrated below. Assata will think that she doing a Diffie-Hellman key exchange with Bobby, when really she is exchanging keys with Edgar, resulting in the beigeish key $p \star a \star e$. Bobby will think that he doing a Diffie-Hellman key exchange with Assata, when really he is exchanging keys with Edgar, resulting in the blueish key $p \star b \star e$. In the end, Assata and Edgar have a shared key (left), and Edgar and Bobby have a shared key (right). But Assata and Bobby think that they have a shared key with each other.



A man-in-the-middle attack against the Diffie-Hellman key exchange

When Assata and Bobby start using what they think is their shared key, Edgar will have to keep up the ruse in order to not be discovered. You see, Assata will encrypt a message with the key she has. If this message makes it to Bobby, Bobby won't be able to decrypt the message because he doesn't have the same key! What Edgar needs to do is intercept the encrypted message and decrypt it with the key they share with Assata. Edgar now has two choices. Edgar could simply read the message, encrypt it with the key he shares with Bobby, and then send it to Bobby. This would be a passive man-in-the-middle attack: Edgar is reading the messages between Assata and Bobby that Assata and Bobby think no one else can read. Edgar's other option is to change the message from

Assata, encrypt it with the key he shares with Bobby, and then send it to Bobby. This would be an active man-in-the-middle attack. In either case, Edgar must continually intercept communications between Assata and Bobby, because otherwise, one of them will receive a message encrypted with a key they don't have, which would alert them to the man in the middle.

Spotting a Man-in-the-Middle Attack with Cryptographic Hashes: Fingerprinting

If Assata and Bobby, after a Diffie-Hellman key exchange, can reliably compare their keys and see that they are the same, they can be assured that any eavesdropper has not mounted a man-in-the-middle attack and can only see their encrypted communications. As a reminder, this is because the parts of the Diffie-Hellman key exchange that an eavesdropper sees does not allow them to create the hidden parts of the keys that Assata and Bobby each select (a and b in the above figure). Indeed, the most basic way to spot a man-in-the-middle attack is for Assata and Bobby to compare their keys.

You may notice some problems with this plan:

If Assata and Bobby try to compare their keys, can't Edgar manipulate the communication to make it seem like their keys are indeed the same?

Of course! Then Assata and Bobby should compare their keys over a different communication channel. For example, if they were originally communicating over the internet, then they should compare keys over the phone. The assumption is that it would be much harder for Edgar to intercept Assata's and Bobby's communications over all the different communication channels that could be used to compare keys. Ideally, Assata and Bobby would meet in person to compare keys. Either way, this is called an *out-of-band* comparison: the band refers to the communication channel, and keys should be compared outside of the band of communication that the keys are exchanged through.

But wait, if Assata and Bobby have another means of communicating, then why don't they just exchange keys the old-fashioned way, without any fancy math to worry about?

Well, cryptographic keys, for modern cryptographic methods, are very long—hundreds or thousands of characters long. It can be cumbersome to do a manual exchange of keys. If a designer of a method for secure communications wanted to automate the exchange of keys over a different communication channel, that designer would have to specify that second communication channel, making the whole secure communications system cumbersome to use. (Imagine having to make a phone call in order to visit a website.) Edgar would also then know what channel the keys are being exchanged on and could play the man in the middle on that channel.

But then isn't it cumbersome to compare keys if they are so long?

Absolutely. So instead of comparing the entire key, Assata and Bobby compare the cryptographic

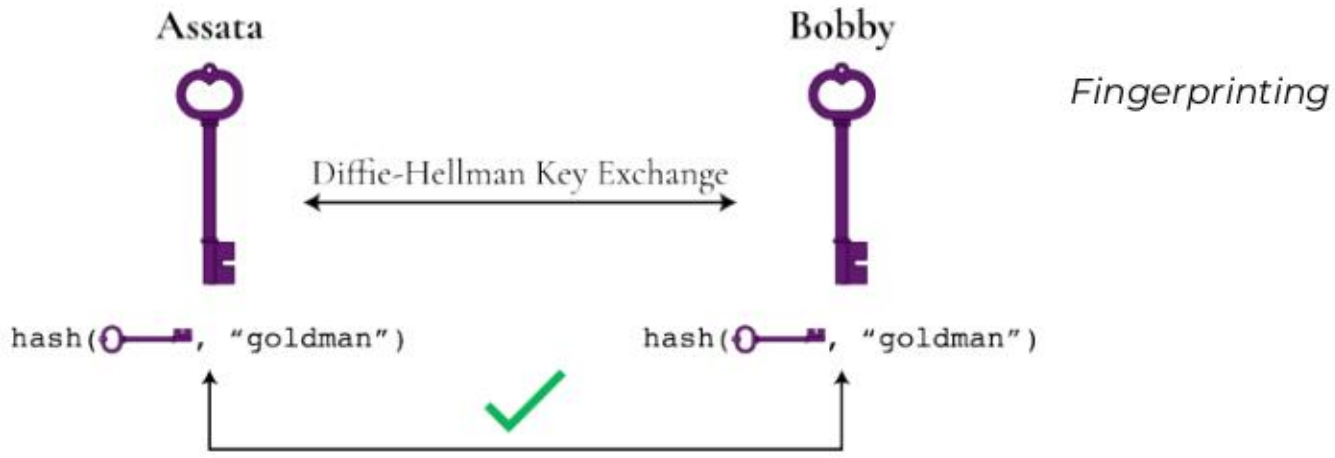
hashes of their keys, as we described in the chapter "Cryptographic Hash." Remember the following properties of the cryptographic hash: (1) It makes the input (in this case the key) much shorter (say, a few dozen characters). (2) It is next to impossible to find two inputs (in this case two keys) that have the same output hash, so Edgar certainly can't manage to do a Diffie-Hellman key exchange with Assata and Bobby so that the hashes end up the same. (3) It cannot be reversed, so if someone intercepted the hash, they could not re-create the input (in this case, the key).

You may recall that a cryptographic hash is sometimes called a fingerprint, and so we call the process of comparing the cryptographic hash of keys *fingerprinting*. Various communication apps may use different terminology for this, including *safety numbers*, *verification*, and *authentication*.

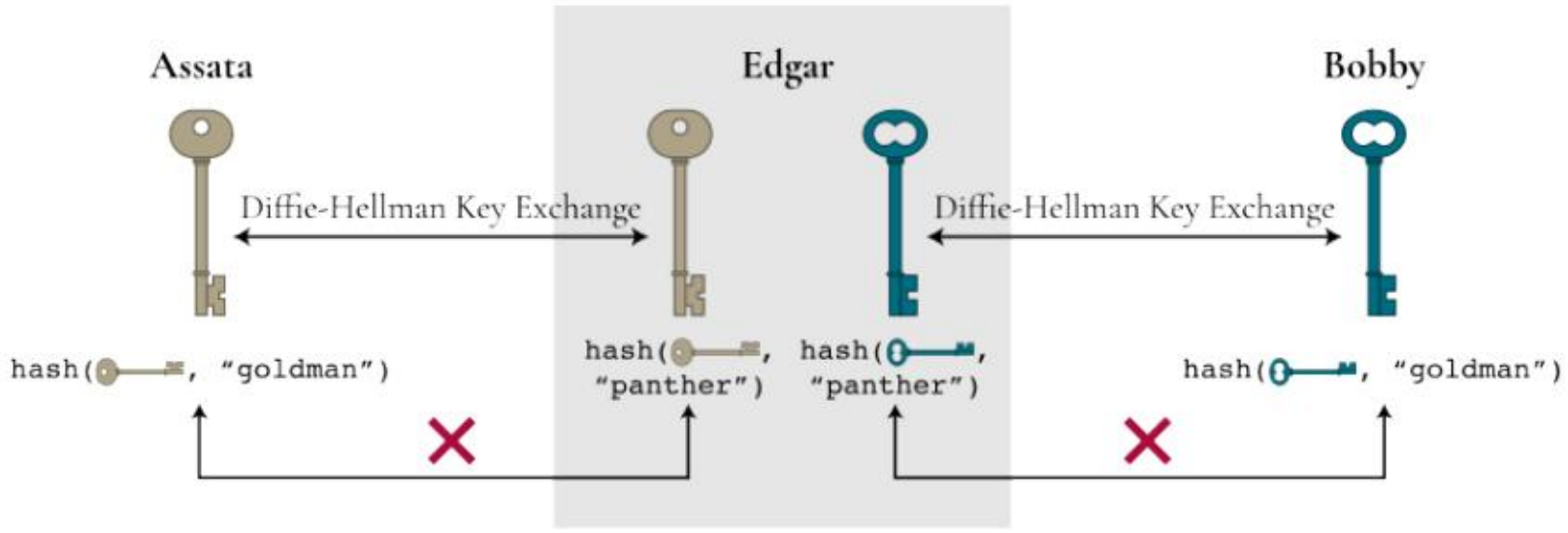
In-Band Fingerprinting

There are two methods to compare keys in band that are not commonly used but are clever and are variations on the out-of-band fingerprinting we described above.

The first relies on the use of a weak password. If Assata and Bobby both know something, like the name of Assata's first pet or the street Bobby grew up on (say "Goldman"), that their presumed adversary doesn't know, Assata and Bobby can use this as a weak password. Assata combines her key with the weak password ("Goldman") and computes the cryptographic hash of the result. Bobby does the same thing with his key. Assata and Bobby then compare the result *in band* (i.e., over the communication channel in which they are already communicating). Because of the properties of the cryptographic hash, Assata and Bobby will only have the same result if they have the same key and the same password:



If Edgar is playing the man in the middle, then Edgar shares a key with Assata and a different key with Bobby. Edgar would have to risk passing along Assata's hash or would have to guess the weak password to be able to compute a result that is the same as what Assata computes and the same as what Bobby computes (as in the figure below). The password does not need to be strong because only a small number of incorrect guesses (e.g., "panther") would be tolerated between Assata and Bobby before they assume the man is in the middle: a brute-force attack by Edgar to guess the password is not feasible.



Fingerprinting to uncover a man-in-the-middle attack

The second method is used for key comparison in voice and video calls. Here, Assata hashes her key into two human-readable words (instead of a string of numbers and characters as we have seen). Bobby does the same thing. If Assata and Bobby have the same key (i.e., there is no man in the middle), then they will have the same set of words. Assata reads the first word, and Bobby reads the second word, so each can compare the result of the hash. If Edgar is in the middle, Assata and Bobby would have different pairs of words. Edgar would have to synthesize Assata's and Bobby's voices (and possibly videos) to speak the words that Edgar shares with Assata and Bobby in order for Edgar's ruse to continue.

The Ability to Fingerprint Is Protective, Even If You Don't Do It

If a method of secure communication does not provide the ability to compare (fingerprint) keys, then there is little benefit to using end-to-end encryption. Man-in-the-middle attacks can be automated in our global surveillance system, so if a man-in-the-middle attack cannot be spotted (by fingerprinting), then it might as well be carried out by default. However, if fingerprinting is made possible, then the man risks being uncovered, particularly if the attacks are automated and widely carried out. Everyone does not need to go through the process of fingerprinting as long as some users do to prevent the widespread deployment of men in the middle.

Of course, for users at risk of targeted surveillance, fingerprinting is essential to the security of their communications.

What to Do When You Can't Fingerprint

In many modes of communication, fingerprinting isn't feasible. One example is in accessing a website via https. In using https, your browser and the website's server will generate a cryptographic key via a Diffie-Hellman key exchange. However, it isn't practical for users to contact the servers of websites via alternate communication channels to fingerprint keys before accessing the content of web pages. Of course, you don't know the voice of the operator of the web server or share any

common knowledge to use in-band comparison methods either! In this case, alternate methods of validating keys, using public-key cryptography and certificate authorities, are used. We will describe public-key cryptography in the chapter “Public-Key Cryptography.”

In Context: The Great Firewall of China

Many people know that the internet is heavily censored in China by the Great Firewall of China. Starting in mid-January 2013, parts of GitHub, a site primarily used to host computer programming code but that can also be used to share more general information, were blocked in China. By January 21, 2013, the entire domain was blocked. However, given GitHub’s central role in computer development and business and the importance of this sector to the Chinese economy, the public backlash successfully unblocked GitHub by January 23, 2013. On January 25, a petition on WhiteHouse.gov was started asking for those involved in building the Great Firewall of China to be denied entry into the United States. The petition linked to a GitHub page, created the same day, listing Chinese individuals accused of contributing to China’s censorship infrastructure. The next day, reports appeared in social media of a man-in-the-middle attack of users accessing GitHub, showing that the equivalent of the fingerprint check for accessing a website via https was failing. The Chinese government had learned that they could not block GitHub, and since GitHub supports https, the Great Firewall could not block accesses to particular pages within GitHub (e.g., based on keyword matches), since https encrypts that information from an eavesdropper. The next option is a man-in-the-middle attack. Any users who ignored warning signs of the attacks would be at risk of their government knowing what pages they were accessing or possibly editing. The Chinese government is the presumptive deployer of widespread man-in-the-middle attacks between users in China and other major internet services, such as Outlook, Apple’s iCloud, and Google.

China isn’t alone in launching man-in-the-middle attacks. Similar attacks have been caught in Syria and Iran too.

What to Learn Next

- Protecting Your Communications

External Resources

- GreatFire provides tools for accessing an uncensored internet in China and reports and verifies Chinese internet censorship and surveillance; in particular, GreatFire has reported and verified man-in-the-middle attacks, presumably by the Chinese government, against Outlook, Apple's iCloud, Google, and GitHub.
- Eckersley, Peter. "A Syrian Man-in-the-Middle Attack against Facebook." Electronic Frontier Foundation, May 5, 2011.
- Electronic Frontier Foundation. "Iranian Man-in-the-Middle Attack against Google Demonstrates Dangerous Weakness of Certificate Authorities." August 29, 2011.

Media Attributions

- mitm-impersonation © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- mitm © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- dhe-mitm © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- fingerprinting © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- fingerprinting-mitm © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license

Passwords

We recommend that you read the chapter "Cryptographic Hash" before reading this chapter.

What You'll Learn

1. When "password protected" means something is encrypted and when it does not
2. How passwords are defeated
3. What password practices you can use to minimize the risk of your passwords
4. How encryption keys can be generated from passwords

When "Password Protected" Does Not Mean Encrypted

All passwords are used to control access. *Account passwords* are used to grant access to an online account, for example. Rarely, though, is the information in that account encrypted with a key that you control, and it is likely that your information is not encrypted at all. That is, the information is usually readable by the provider (e.g., Google, Dropbox). Other passwords are used to *unlock* an encrypted file or document, and we will refer to these as *encryption passwords*. The use of an account password is like telling a bouncer your name and having that name be matched on a list of approved guests, whereas the use of an encryption password is more like using a key to unlock a safe. In the first case, it is up to the bouncer (a metaphor for your online account provider) to give you access. In the latter case, the safe represents the ciphertext, and the contents of the safe are the plaintext—gaining access to the plaintext is impossible (or at least impractical) without the key or password. (In fact, encryption keys are in some cases generated from the password, as we describe below.)

That said, even though your information is not encrypted with an account password, you should still minimize the people who could have access to your information. But to understand why we recommend the password practices we do, it helps to understand how passwords can be compromised.

Password Cracking

Passwords can be compromised or *cracked* one at a time or as a whole batch, as in all the passwords for all the accounts on a given system. Passwords are hot commodities. Since people frequently use the same password for many accounts and "popular" (a.k.a. terrible) passwords are used by many people (`123456`, `password`, `qwerty`, `admin`, `welcome`, `password`, to name a few real examples), discovering passwords used for one service can result in an account compromise for a different service and possibly for a different person.

Let's consider the ways in which passwords are compromised.

To crack one password, an adversary could do so via the same means that you enter your password—for example, via a website. This is relatively easy for the website operator to help protect against—for example, by locking an account after a few incorrect password entries or forcing delays after entering the password to slow down repeated guesses. Another way the account provider can help is by allowing for two-factor authentication. This is where, in addition to entering a password to access an account, you must also enter an authentication code that is delivered to you via text, via an app on your smartphone, or to a physical authentication key (such as a YubiKey). To compromise your account, an adversary would need your password as well as your device that receives the authentication code.

An adversary could also physically access the device (your phone or computer) on which you enter your password. More likely—and as is regularly reported in the news—the server on which your password is stored is compromised or hacked. In this case, it won't just be your username and password that is compromised; everyone who has an account on that system will be at risk. Although an adversary who has gained access to a database of passwords on the server will likely have access to your account information too, as we alluded to above, the point of the hack might be to gain access not to the hacked service but to another service entirely.

A responsible web service provider won't store your password in plaintext on their server but will store a cryptographic hash of your password. To uncover a password (or all passwords), an adversary computes the cryptographic hash of a guessed password and compares this to the database of stolen passwords. In practice, password-cracking tools (e.g., John the Ripper) use three techniques:

1. Dictionary attacks: trying dictionary words, common *salts* of dictionary words (e.g., `pa55w0rd`, `fr33d0m`), and previously cracked passwords
2. Brute force: trying all possible combinations of letters and numbers of symbols (for practical reasons, this method only works for relatively short passwords)
3. Precomputed hashes: comparing against a table of cryptographic hashes of possible passwords that are computed ahead of time

A user can foil the first two techniques by using good password practices (described below). A service provider can make password cracking less practical by using a cryptographic hash function that is slow to compute or uses a lot of memory. This wouldn't be noticeable for a single password

(such as would be done when you log in) but would slow the computation of the hashes during cracking.

A service provider can further foil the use of precomputed hashes if they add a long random sequence of characters (*salt*) to your password when you log in. This salt can be stored in plaintext with your username, so an adversary would have this information too but would not have had the salt when the precomputed table of hashes was prepared. Additionally, if two users have the same password, because their salt will be different, the cryptographic hash of their passwords with the corresponding salts will be different. This forces an attacker to uncover each password individually.

For all of this, you are trusting the online service provider to responsibly store and protect your user information, including your hashed password, if they have even hashed it. The rest is up to you.

Best Practices for Passwords

To guard against the methods deployed in password cracking, your passwords should be sufficiently long (to prevent brute-force attacks), be uncommon (to prevent dictionary attacks), and not be reused (so if one of your accounts gets compromised, all your accounts aren't compromised).

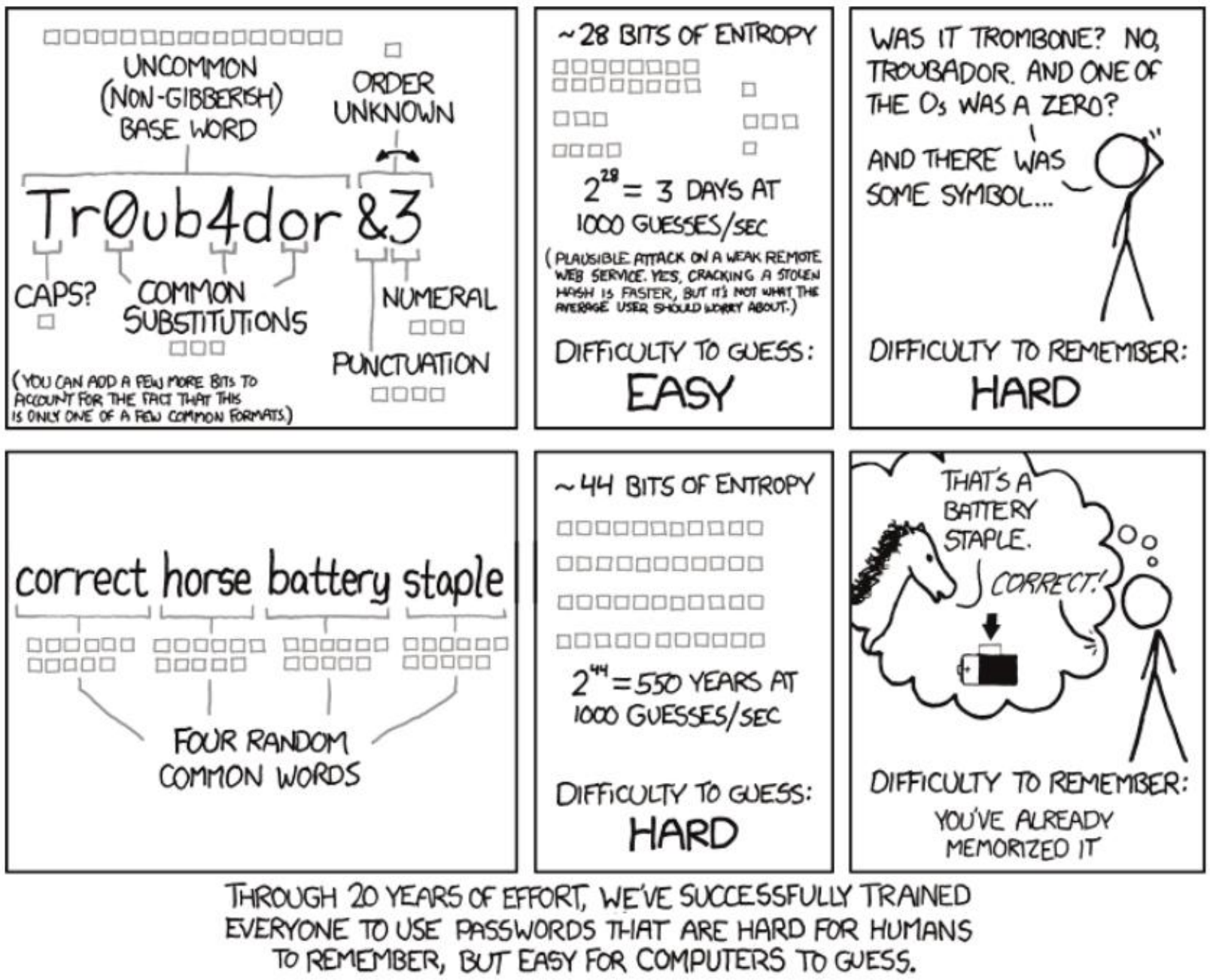
To accomplish this, use a password manager to generate and store all the passwords that you don't need to manually type in. The password manager should be able to generate strong random passwords for you such as `bdY,Fsc_7\&*Q+cFP`. This is great for a password that you never have to type in—that is, a password that the password manager will input for you.

For passwords that you will necessarily need to type in (e.g., a password you enter on your phone, the password you protect your password manager with, the password you use to encrypt or unlock your computer), use a diceware password, a.k.a. a passphrase, a.k.a. a random sequence of words such as

`remake.catfight.dwelled.lantern.unmasking.postnasal`

You can generate this password manually using dice and a word list. Many password managers will also generate such passwords, although you probably won't need many of these.

Note that the two above examples of passwords are randomly generated. This is important because even if you think your password is awesome and strong, if you came up with it with your brain, then someone else's brain probably also came up with it, and so it is susceptible to dictionary attacks.



XKCD password strength

Generating Encryption Keys from Passwords

In some cases, passwords *are* used to unlock an encrypted file or device. An encryption key in this case is in fact generated from a password or passphrase using a key derivation function, which is, essentially, a cryptographic hash function. The input to the cryptographic hash function is your password, and the output is the cryptographic key. Why would this work? Let's revisit the properties of cryptographic hash functions:

1. *Regardless of the length of the input, the output is always the same size*. So no matter how short (and weak!) your password is, you will get a cryptographic key of the right size. (But a short and weak password is susceptible to the password-cracking methods we discussed above.)
2. *The same input always results in the same output*. So your password will always generate the corresponding cryptographic key you need.
3. *It is infeasible to generate the input from the output*. So if someone manages to get your key, at least they won't be able to re-create your password.
4. *It is infeasible to find two different inputs that result in the same output*. So someone trying to crack your password would be unlikely to even find some other password that

results in the same cryptographic key as yours.

5. *A small change to the input changes the output so extensively that the new hash value appears uncorrelated with the old hash value*. Well, this property isn't as useful for cryptographic key generation...

In Context: When Precautions Are Not Enough

In 2016, longtime activist DeRay Mckesson had his Twitter account and two email addresses compromised in a targeted attack *despite* having a two-factor authentication set up. His adversary was able to get control of his phone by calling Verizon and requesting a new SIM card and knew enough about Mckesson to convince Verizon to do so. Once the adversary had access to Mckesson's phone number, they were able to receive password-reset codes to change his passwords and gain access to his accounts. It is a reminder that no security measure will be perfect, and for those who are subject to targeted attacks (in this case, Mckesson was targeted for his support of Black Lives Matter), extra vigilance is necessary. Here, the fact that access to Mckesson's phone number could be used to force a password reset reduced account protections from two-factor to one factor: only phone access separated an adversary from Mckesson's account rather than a password plus phone access.

What to Learn Next

- Protecting Your Devices

External Resources

- Dreyfuss, Emily. "@Deray's Twitter Hack Reminds Us Even Two-Factor Isn't Enough." *Wired,* June 10, 2016.
- Wikipedia. "John the Ripper." December 29, 2020.
- TeamPassword. "Top 50 Worst Passwords of 2019." December 18, 2019.

Media Attributions

- password_strength © Randall Munroe is licensed under a CC BY-NC (Attribution NonCommercial) license

Public-Key Cryptography

We recommend that you read the chapter "Exchanging Keys for Encryption" before reading this chapter.

What You'll Learn

1. The difference between symmetric- and asymmetric-key cryptography
2. How public-key cryptography works

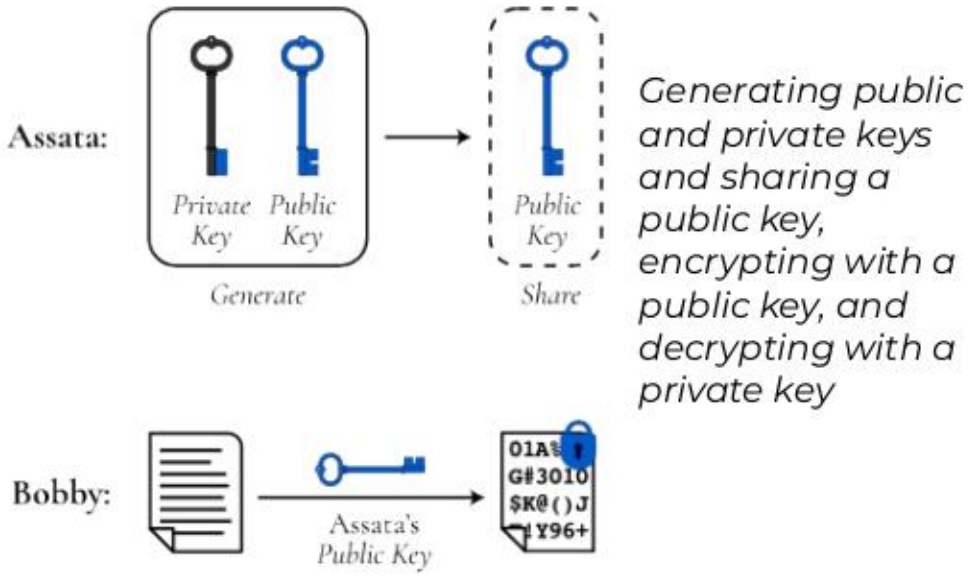
Encryption protocols can be classified into two major types. In *symmetric-key cryptography*, the key used to decrypt a message is the same as (or easy to transform from) the key used to encrypt the message. This is the case for the basic ciphers (Caesar, Vigenère, and the onetime pad) that we described in the chapter "What Is Encryption?" (There are, of course, modern symmetric-key ciphers that are used—for example, to encrypt the data on your phone or computer.) As we saw, these protocols are challenging to use for communication because you need to first find some way to privately share the key with your communication partner. Diffie-Hellman key exchange gave a method for two people to generate a shared key (that can be used in a symmetric-key encryption protocol) while only communicating over an insecure channel (such as the internet).

Asymmetric-key cryptography or public-key cryptography solves the key sharing problem in a different way. Rather than have one key that is used to both encrypt and decrypt, public-key cryptography uses two keys: one key to encrypt (called the *public key*) and one key to decrypt (called the *private key*). This pair of keys has the following properties:

1. It is infeasible to generate the private key from the public key: the keys must be generated together.
2. A message that is encrypted by the public key can only be (feasibly) decrypted with the corresponding private key.

Suppose Bobby wants to send Assata an encrypted message. Assata creates a private-key/public-key pair and sends Bobby her public key (over an insecure channel). Bobby uses the public key to create the ciphertext and sends the ciphertext to Assata. The ciphertext can *only* be decrypted using Assata's private key. Even though anyone may have Assata's public key, the *only* thing that can be

done with the public key is encrypt messages that can only be decrypted using Assata's private key. Security is therefore achieved by keeping the private key private: secret and secure.



Generating public and private keys and sharing a public key, encrypting with a public key, and decrypting with a private key

In this model, anyone can, in fact, publish their public key. For example, Assata could publish her public key online so that anyone wishing to send Assata an encrypted message could encrypt that message with her public key first. Likewise, Bobby could create his own pair of public and private keys and publish his public key online so that others could send him encrypted messages that only Bobby could decrypt with his (securely stored) private key.

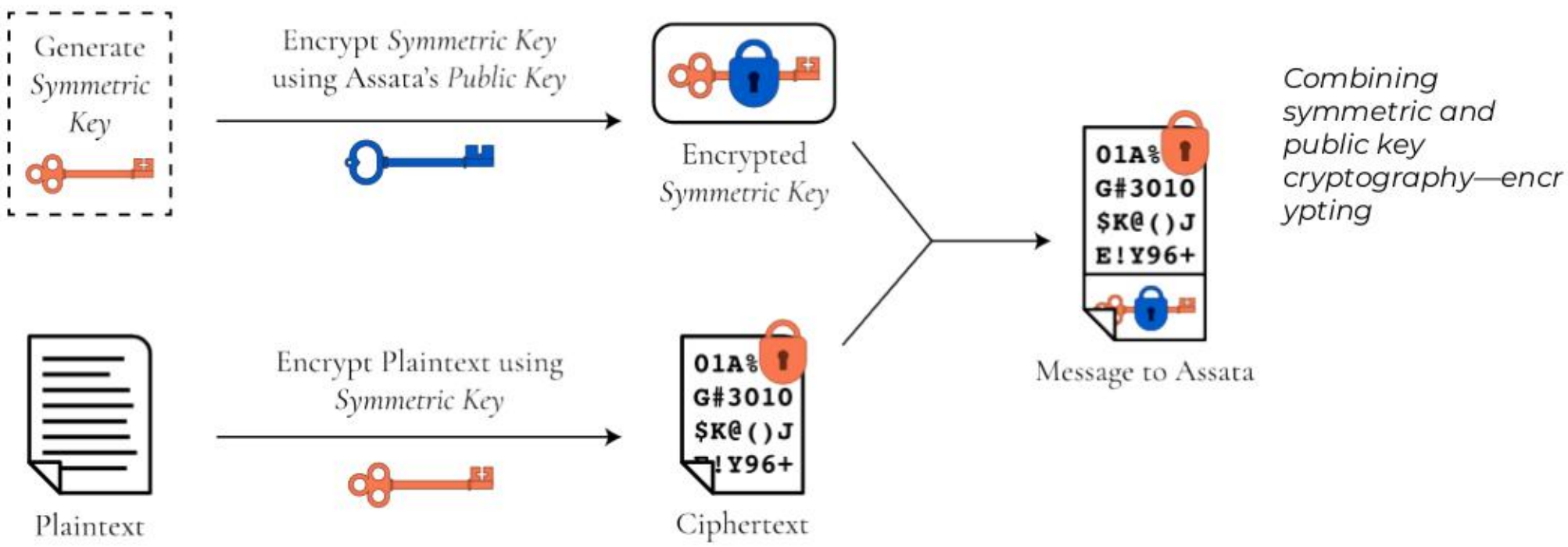
Revisiting Diffie-Hellman Key Exchange: Public-Key or Symmetric-Key Cryptography?

Let's revisit Diffie-Hellman key exchange through the lens of symmetric and public-key cryptography. Recall that Assata and Bobby agree (publicly/insecurely) on a number p. Alice picks a (secret) number a and computes $p \star a$ to send (publicly/insecurely) to Bobby. One could thus view a as Assata's private key, $p \star a$ as Assata's public key, and this scheme as part of a public-key protocol. But Bobby picks his own secret number b and combines it with Assata's public key to get $p \star a \star b$. Likewise, Assata combines Bobby's "public key" $p \star b$ with her own private key to get $p \star b \star a$. Since $p \star a \star b = p \star b \star a$, Assata and Bobby have a common key to use for encryption, and they use the same key for decryption. In this way, this is part of a symmetric-key protocol. For these reasons, the Diffie-Hellman key exchange lies somewhere between public-key and symmetric-key cryptography.

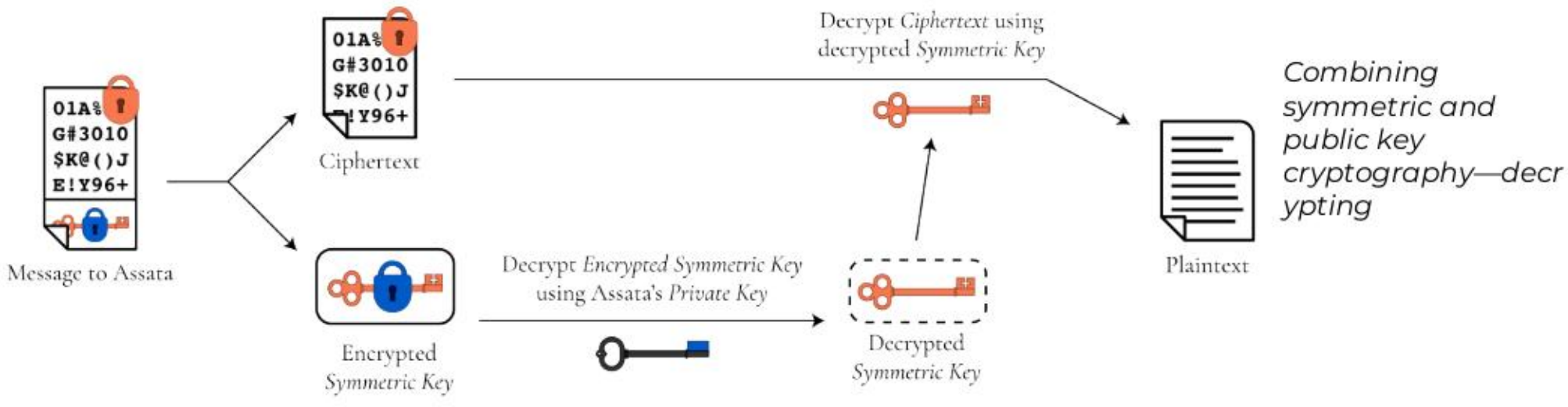
Combining Public-Key and Symmetric-Key Cryptography

Public-key encryption is usually more computationally expensive than symmetric-key encryption. To achieve the same security guarantees (e.g., against brute force and other attacks), public keys need to be much longer than symmetric keys. Also, performing the encryption itself takes longer using public keys than symmetric keys. There is also the problem that the longer you use a key for encryption, the more ciphertext examples there are to try to use to break the encryption (other than brute force)—that is, keys tend to *age poorly*.

For these reasons, public keys are generally used to encrypt a symmetric key for a given (communication) *session*. Suppose Bobby wishes to send Assata an encrypted message. Bobby generates a symmetric encryption key and encrypts the message with the symmetric key using a symmetric cipher. He then *encrypts the symmetric key* using Assata's public key. He sends the encrypted message and the encrypted key to Assata:



Assata decrypts the encrypted key using her private key and then uses the result to decrypt the encrypted message:



Since the public key is only used to encrypt keys (which are typically random-looking strings), the public key does not age, because methods of breaking the encryption that rely on human-language phrases would fail. An added benefit is that if one message is successfully decrypted, that does not help in breaking the encryption of a different message, since each message is encrypted with a different key.

In Context: Antinuclear Activism and Pretty Good Privacy

A particularly robust implementation of public-key cryptography is PGP, an acronym for the understatement Pretty Good Privacy. (An interoperable, free, and open-source version of PGP is GPG, or GNU Privacy Guard.) PGP encryption is most commonly used for encrypting email communications with several plug-ins and email clients that support using PGP encryption. There are a number of (synchronized) online directories of PGP keys, each associated with an email address, that allow Bobby to look up Assata's PGP key in order to send her an encrypted email.

Phil Zimmermann, a longtime antinuclear activist, created PGP in 1991 so similarly inclined people might securely use bulletin-board services (BBSes, the Reddit of the 1980s) and securely store messages. He developed PGP as an open-source project, and no license was required for its noncommercial use. Posting it initially to a newsgroup that specialized in grassroots political organizations, mainly in the peace movement, PGP made its way to a newsgroup used to distribute source code and quickly found its way outside the United States. Users and supporters included dissidents in totalitarian countries, civil libertarians, and cypherpunks. However, at the time, cryptosystems using keys larger than forty bits were then considered munitions within the definition of the US export regulations. PGP was initially designed to support 128-bit keys. In February 1993, Zimmermann became the formal target of a criminal investigation by the US government for "munitions export without a license." Zimmermann challenged this by publishing the entire source code of PGP in a book, which was distributed and sold widely. Anybody wishing to build their own copy of PGP could cut off the covers, separate the pages, and scan them using an OCR program, creating a set of source code text files. While the export of munitions (guns, bombs, planes, and software) was (and remains) restricted, the export of books is protected by the First Amendment. After several years, the investigation of Zimmermann was closed without filing criminal charges against him or anyone else.

US export regulations regarding cryptography remain in force but were liberalized substantially throughout the late 1990s. PGP encryption no longer meets the definition of a nonexportable weapon.

What to Learn Next

- Authenticity through Cryptographic Signing

External Resources

- Electronic Frontier Foundation. “A Deep Dive on End-to-End Encryption: How Do Public Key Encryption Systems Work?” *Surveillance Self-Defense*, September 29, 2014.
- Zimmermann, Phil. “Why I Wrote PGP.” 1999.

Media Attributions

- pubkeycrypto-key-gen-share © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- pubkeycrypto-split1-encrypt © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- pubkeycrypto-split2-decrypt © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license

Authenticity through Cryptographic Signing

We recommend that you read the chapters “Cryptographic Hash” and “Public-Key Cryptography” before reading this chapter.

What You'll Learn

1. How to achieve the digital equivalent of a signature
2. How cryptographic signatures can be used to provide authenticity
3. What electronic authenticity means
4. How cryptographic signatures can be used to propagate trust

Public-key cryptographic systems can often be used to provide authenticity. In PGP, this is allowed by the complementary nature of the public and private keys. In the beginning, two cryptographic keys are created, and either can be used as the public key; the choice as to which is the public key is really just an arbitrary assignment. That is, either key can be used for encryption as long as the other one is used for decryption (and the one used for decryption is kept private to provide security).

Once you have assigned one cryptographic key as the public key and the other cryptographic key as the private key, you could still choose to encrypt a message with your private key. However, then anyone with your public key could decrypt the message. If you make your public key, well, public, then anyone could decrypt your message, and so this would defeat the purpose of using encryption to achieve message privacy.

However, this should illustrate to you that the only person who could have encrypted a message that can be decrypted with *your* public key is *you*, the person with *your private key*. Encrypting a message with your private key provides the digital equivalent of a signature and is called *cryptographic signing*. In fact, cryptographic signing provides two properties of *authenticity*:

1. *Attribution*. You wrote the message (and not someone else).
2. *Integrity*. The message is received as it was written—that is, it has not been altered.

The second property comes from the fact that a tamperer would have to alter the ciphertext

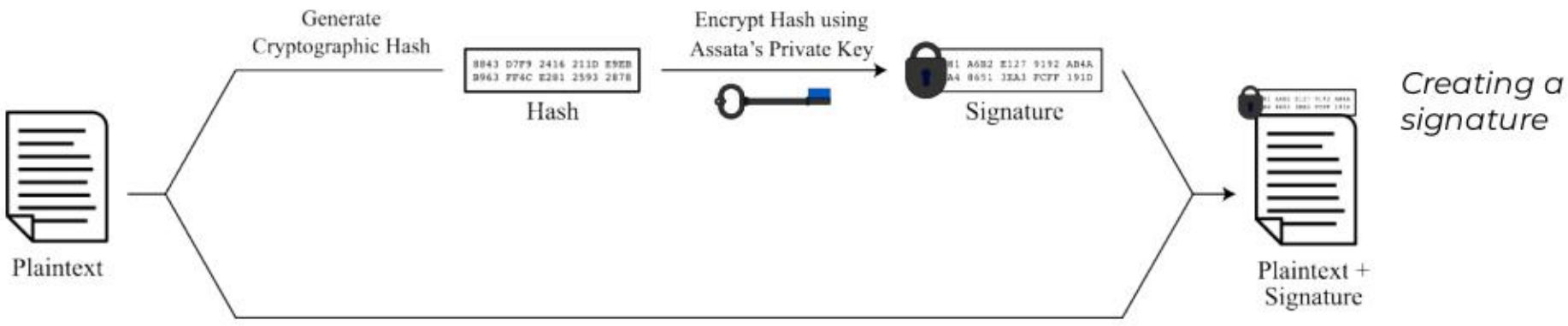
such that decrypting the ciphertext with your public key generates the tamperer's desired altered plaintext. But this is completely infeasible.

These properties are only meaningful if you are the only one who controls your private key, since anyone who gains control of your private key could cryptographically sign their own altered text.

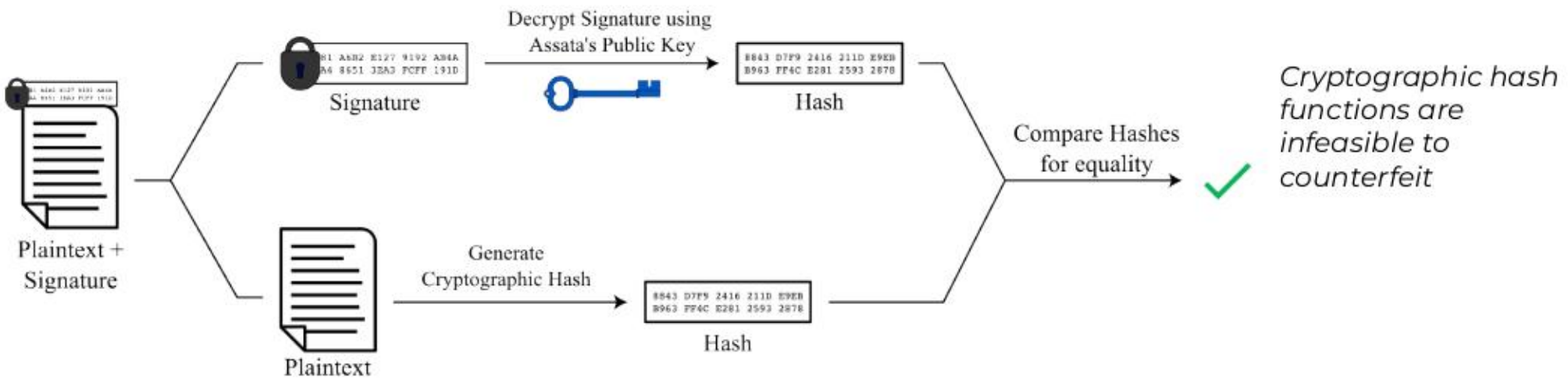
Cryptographically Signing Cryptographic Hashes

In practice, rather than encrypting the entire message, one would encrypt a cryptographic hash (a.k.a. digest or fingerprint) of the message for the purpose of cryptographic signing. This is done for efficiency reasons. Let's consider the protocol for Assata signing a message and Bobby verifying the signature, as illustrated in the following text.

Assata takes a cryptographic hash of her message and encrypts the result with her private key, creating a signature, which she can attach to the message:



Bobby takes the signature and decrypts it using Assata's public key, giving the hash that is the same as what Assata generated. He then takes his own cryptographic hash of the message and compares the result to the decrypted hash he received from Assata:



Recall that cryptographic hash functions are infeasible to counterfeit. So if the two hashes that Bobby generates (one directly from Assata's message and one from Assata's signature) are the same, then we know two things:

1. Only Assata could have generated the signature. Only Assata could encrypt something that can be decrypted with her public key, since she is the only one with her private key.
2. The message has not been altered since Assata wrote it. If someone altered the message, then the hash of the message would differ from the hash contained in the signature. Any counterfeiter would therefore have to forge a new signature but can't generate one without Assata's private key.

That is, we obtain authenticity cryptographically.

Note that Edgar, in a man-in-the-middle attack, could simply remove the signature. So for cryptographic signing to be effective, you need to agree to use cryptographic signing all the time. Modern end-to-end encrypted messaging apps generally have signing built in by default, though this is often invisible to the average user.

Applications of Cryptographic Signing

As in our example above, cryptographic signing can provide authenticity to messages in a similar way that traditional handwritten signatures and wax seals did. However, you can cryptographically sign more than just messages (such as emails).

Verifying Software

Perhaps the most explicit and common use of cryptographic signatures is for verifying software, even if you aren't aware of it. Software such as apps will only do what their developers want and say they will do if they haven't been tampered with on the way from the developer to your computer or phone. Responsible developers will sign their products in the same way as we described signing for messages. A program or app is really just a computer file (or set of files), which is just a sequence of characters or a type of message. If a developer has signed their software using public-key cryptography, a careful user can check the signature by getting the developer's public key and performing the validation illustrated above. (The developer should provide their public key via a channel different from the one you downloaded the software from. This would allow an out-of-band comparison as described in the chapter "The Man in the Middle"—the public key, which you use for validation, is out of band from the message or software download.)

Managing Fingerprint Validation and the Web of Trust

To trust Assata's public key, Bobby should really verify her public key by checking the fingerprint of the key, as we have described. Otherwise, Edgar, the interloper, could furnish Bobby with a public key that he holds the corresponding private key for. But if Bobby has verified that Assata's public key is

genuinely hers, Bobby can cryptographically sign Assata's public key (with his own private key). This allows Bobby to keep track of the public keys that he has verified and allows him to share Assata's key with other people as follows.

Suppose Cleaver wants to send Assata an encrypted message and wants to be sure that Edgar is not going to play the man in the middle. But Cleaver does not have a secondary channel through which to verify Assata's public key. However, Cleaver has received and verified Bobby's public key. So Bobby can send Assata's public key to Cleaver with his signature. If Cleaver trusts Bobby and has verified his public key, then Cleaver can verify his signature on Assata's public key and trust that Assata's public key is genuine.

This is the basis of the web of trust. Rather than directly verifying fingerprints of keys, you can do so indirectly, as long as there is a path of trust between you and your desired correspondent.

In Context: Warrant Canaries

Warrant canaries or *canary statements* inform users that the provider has *not*, by the published date, been subject to legal or other processes that might put users at risk, such as data breaches, releasing encryption keys, or providing back doors into the system. If the statement is not updated according to a published schedule, users can infer that there has been a problem that may put their past or future data at risk. Riseup.net, for example, maintains a quarterly canary statement that is cryptographically signed so that you can ensure its authenticity—that is, that the people at riseup.net wrote the statement. They include a link to a news article dated on the day of release in their statement to give evidence that the statement was not published before the day of release.

The use of warrant canaries began as a way to circumvent gag orders in the US that accompany some legal processes in which the US government can force someone to withhold speech. On the other hand, the US government can rarely force someone to say something (particularly that isn't true) and so could not compel a provider to keep up a canary statement that falsely claims nothing has happened.

The term originates from the use of canaries in coal mines to detect poisonous gases: if the canary dies, get to clean air quickly!

What to Learn Next

- Metadata

External Resources

- Riseup. “Canary Statement.” Accessed February 9, 2021.
- Tor Project. “How Can I Verify Tor Browser’s Signature?” Accessed February 9, 2021.

Media Attributions

- pgpsigning1 © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- pgpsigning2 © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license

Metadata

What You'll Learn

1. What metadata is
2. What metadata can reveal
3. Why metadata is difficult to protect

What Is Metadata?

Metadata is all the information *about* the data but not the data itself and is best illustrated with a few examples.

1. For a phone call, the metadata will include the phone numbers involved, the start time of the call, and the length of the call. For cell phone calls, the metadata will likely include the location of your phone (the GPS coordinates), the cell tower that you are connected to, and even the type of phone you are using. Metadata of phone calls would not include the audio transmission itself—this would be the "data." The historical use of recording phone-call metadata is for the purposes of billing.
2. Most modern digital photographs include information about the time and place the photo was taken, the type of camera used, and its settings. In this case, the photo itself is the data. Many websites, such as Facebook, Twitter, and Instagram, remove this metadata for your privacy when you upload a photo or video. Others do not, such as Google, Flickr, and YouTube.
3. Almost all modern color printers, at the request of the US government to printer manufacturers over fears of their use in money counterfeiting, print a forensic code on each page that may be visible or not. In this case, the printed sheet (less the forensic code) would be the data, and the information encoded by the forensic code would be the metadata. The forensic code, which may or may not be visible to the human eye, has been known to include the day and time the sheet was printed and the serial number of the printer used.

The first disclosure by Edward Snowden revealed that the NSA was collecting all the metadata of calls made by Verizon customers, forcing a conversation about metadata into the public consciousness. A debate on what privacy was being invaded by this practice ensued. Earlier that year, the Associated Press fought back against the collection of metadata obtained by subpoena from the Justice Department, saying, "These records potentially reveal communications with

confidential sources across all of the newsgathering activities undertaken by the AP during a two-month period, provide a road map to AP's newsgathering operations, and disclose information about AP's activities and operations that the government has no conceivable right to know." A court opinion noted that the collection of GPS data through such metadata collection "can deduce whether he is a weekly churchgoer, a heavy drinker, a regular at the gym, an unfaithful husband, an outpatient receiving medical treatment, an associate of particular individuals or political groups."

In an internal document, the NSA has referred to metadata as being one of the agency's "most useful tools."

Metadata and the Internet

When you visit a website, information is being sent between your computer and the server of the website through the internet. At a basic level, a message is sent from your computer to the server requesting the contents of the website, and then the contents of the website are sent from the server to your computer. The information being sent over the internet is often referred to as *traffic*, and any message being sent will actually be broken up into many shorter messages or *packets*. Each packet has three main parts:

1. The *header* includes the internet address of the sender and the receiver (e.g., your computer and the website's server) and a description of the type of data that is being sent (e.g., HTML).
2. The *data* is the content of the message (e.g., the content of the web page or part of the web page).
3. The *trailer* indicates the end of the packet and provides proof that the packet has not been corrupted in transit (using a hash function).

The metadata is composed of the header and the trailer. The header is difficult to protect or conceal because it indicates where a packet should be sent. Just like sending a letter, an address is needed for delivery. Your internet address, or IP address, is related to your physical location; in fact, often your physical location can be determined from your IP address.

This description applies to any information that is sent over the internet—email, video streaming, VOIP calls, and instant messages included.

In Context: Protecting a Whistleblower

In May 2017, Reality Winner disclosed NSA documents reporting on Russian interference in the 2016 US presidential election. Her arrest, days before the story was published, prompted much speculation around how she was so quickly identified as the whistleblower, with many people pointing the blame at the website Intercept for their handling of the story. Reality Winner had

anonymously mailed a color printout of the documents to the Intercept. In standard journalistic fashion, the Intercept sent a photograph of the documents to the NSA for verification. The same photograph was redacted and made public in their reporting. Shortly after the publication of the story, several people pointed out that printer forensic code was visible in the photo and determined the day and time the document was printed and the serial number of the printer. While it is possible that the FBI could have identified Reality Winner from this information (to best protect its source, the Intercept should have redacted the forensic code from the photo), it is probably more likely she was outed by logs of file accesses on her work computer.

What to Learn Next

- Anonymous Routing

External Resources

- CNN. "AP Blasts Feds for Phone Records Search." May 14, 2013.
- Electronic Frontier Foundation. "Justice Department Subpoena of AP Journalists Shows Need to Protect Calling Records." May 13, 2013.
- Electronic Frontier Foundation. "Secret Code in Color Printers Lets Government Track You." October 16, 2005.
- Snowden Archive—the SIDtoday Files. "The Rewards of Metadata." Intercept, January 23, 2004.
- *New Yorker*. "The Metadata Program in Eleven Documents." December 31, 2013.
- Intercept. "Top-Secret NSA Report Details Russian Hacking Effort Days before 2016 Election." June 5, 2017.
- *Atlantic*. "The Mysterious Printer Code That Could Have Led the FBI to Reality Winner." June 6, 2017.

Anonymous Routing

We recommend that you read the chapters “Exchanging Keys for Encryption” and “Metadata” before reading this chapter.

What You’ll Learn

1. Who has access to what on the internet
2. Technologies that allow for anonymous communications online
3. What anonymity is and the pitfalls of anonymity

In order to communicate online, packets of information need to be addressed to your computer, whether that information is from an instant-message conversation, an email, or browsing the web. In this section, we mostly focus on web browsing, although the same ideas apply in most settings. Your computer’s address, or IP address, is how internet communications reach your computer in the same way as a mailing address allows an envelope or package to reach your mailbox. For that reason, your computer’s current IP address (which changes depending on where you are connecting to the internet) is related to your physical location. How refined that physical location is depends on how much information the internet service provider (ISP) reveals and to whom they are willing to reveal it. The ISP knows which cable, phone line, or cell tower you are receiving internet traffic through but may only provide zip code information to the proliferation of IP geolocation websites. Or they may provide the location of a specific house.

Your IP address is just one piece of metadata that is necessary in order to get information to your computer. When browsing the web, though, a lot of other metadata, while not strictly necessary, is transmitted to “maximize your browsing experience.” This information includes details such as what browser plug-ins you use, your time zone, and your screen size and can be used as a unique identifier across IP addresses that you use to connect to the internet.

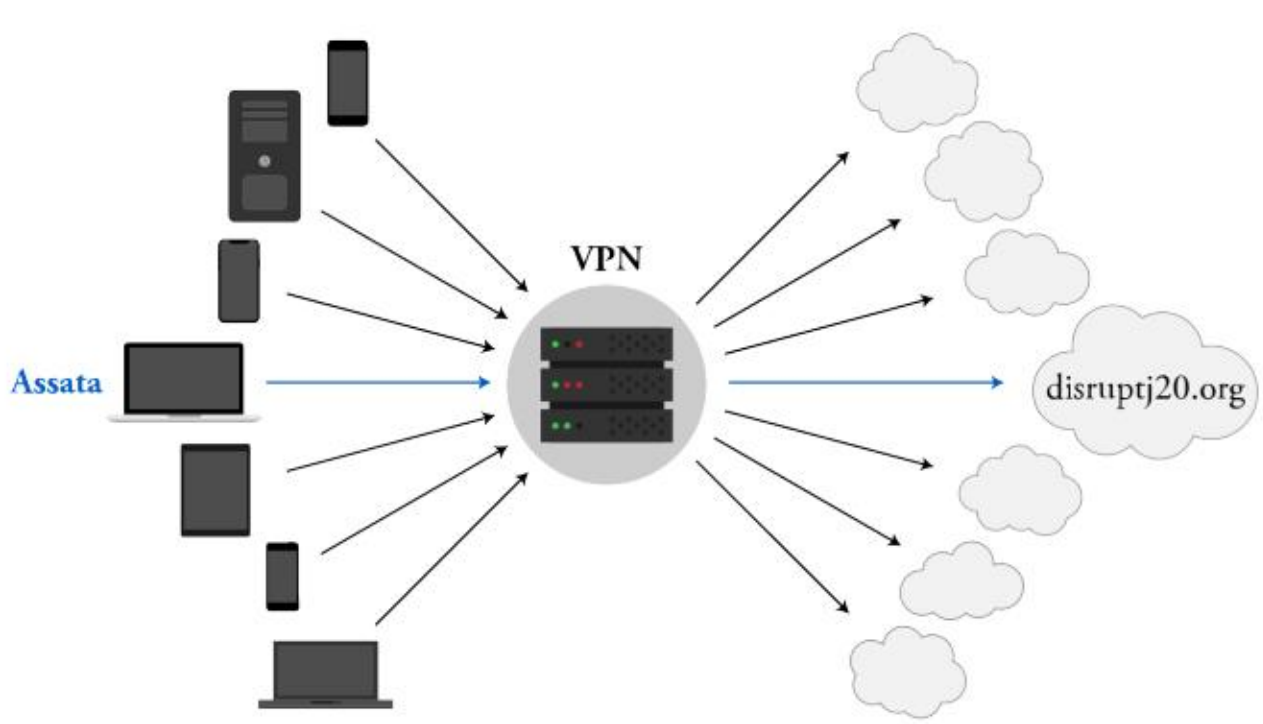
Who has access to all this metadata that can be used to identify you? Without encryption, such as using https, any eavesdropper would have access to this metadata as well as the content of your communications. Encryptions will protect some metadata from your ISP and eavesdroppers (such as which browser you are using) but not your IP address and not the web domains you are visiting. And the servers of the websites you are visiting will have access to your metadata as well as any content.

But since metadata is used to get information to you, is there any way to protect this metadata, and who could you protect it from? We describe two ways to anonymize your web browsing.

Trusting a Middle Man: Virtual Private Networks

Virtual private network (VPN) technology began as a means to extend a local network (such as a university's or company's network) to remote locations (such as off-campus housing and home offices) so that no matter where you were, you could access the same resources as you would if you were on the local networks (such as library and software subscriptions). While connected to a VPN, a web page host will see the IP address of the local network the VPN is extending *as your address*, the IP address of your home. For this reason, VPN use has become popular for anonymizing your location.

A VPN operates as a (hopefully benign) middle man (illustrated below). Rather than sending all her web requests directly, Assata sends all her web requests to her VPN, the VPN fetches her request from the internet for her, and then the VPN sends the results back to Assata. The specifics of how this is done vary between different VPN services, but generally, the communications between you and the VPN are encrypted. The protective quality of a VPN relies on many other people also connecting to that VPN. An eavesdropper looking at communications to and from the VPN will be able to identify the individuals connecting to the VPN and the web requests the VPN is fetching but ideally will be unable to match those web requests with the corresponding users because there are too many simultaneous requests in and out of the VPN.

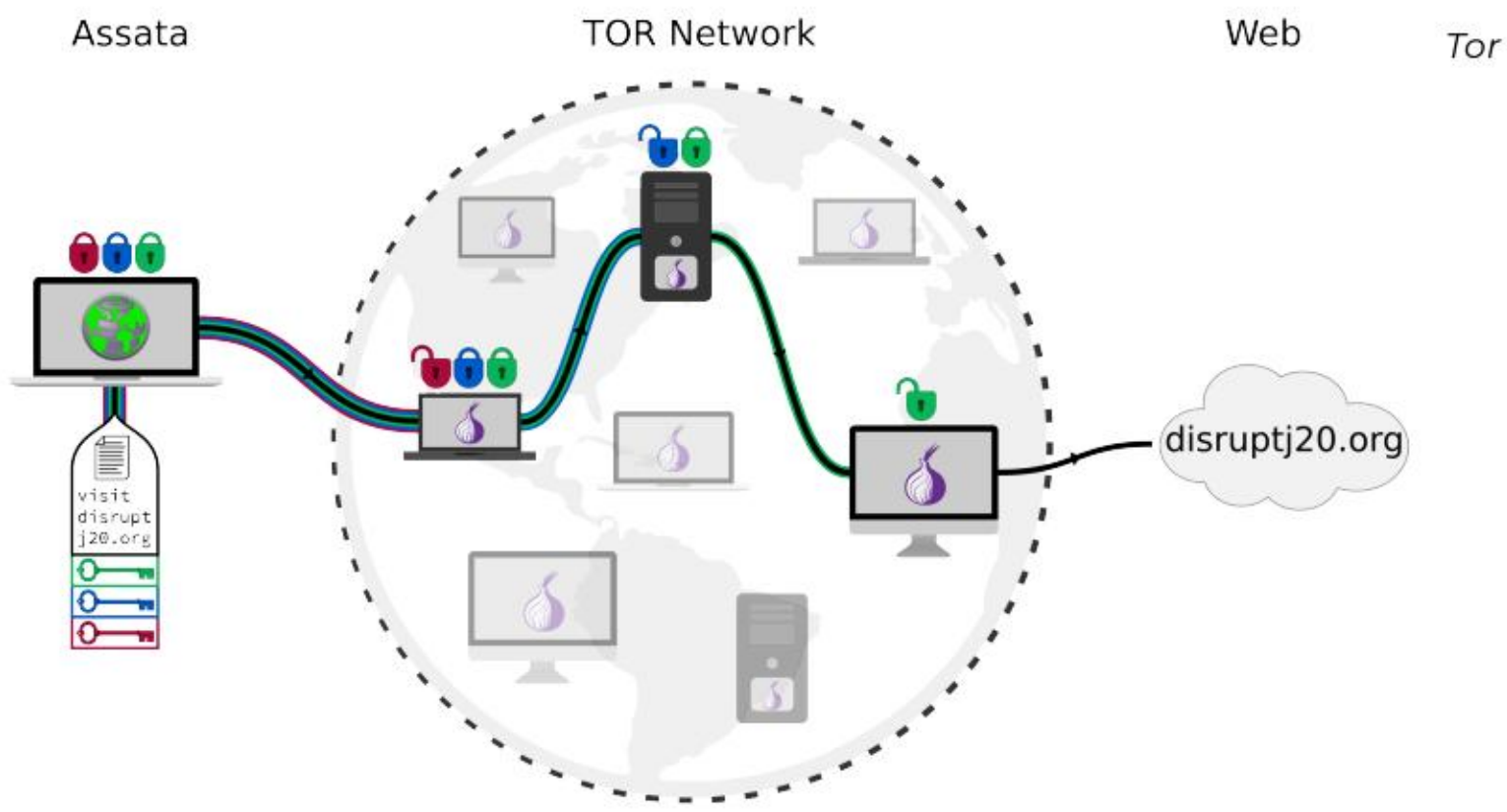


Of course, the VPN provider knows all your internet behavior, and with their cooperation, an adversary would too: you are trusting your VPN provider with that information. However, your ISP

(without using a VPN) has access to the same information: you are putting the same trust in your VPN provider as you must in your ISP. The difference is that your ISP does not conceal your IP address from destination servers on the internet, while a VPN does. Some increased privacy risk, however, comes with using the same VPN across many connection locations (e.g., home, work, coffee shop), giving a single entity (that VPN) a more complete view of your internet use than available to the ISP at each location.

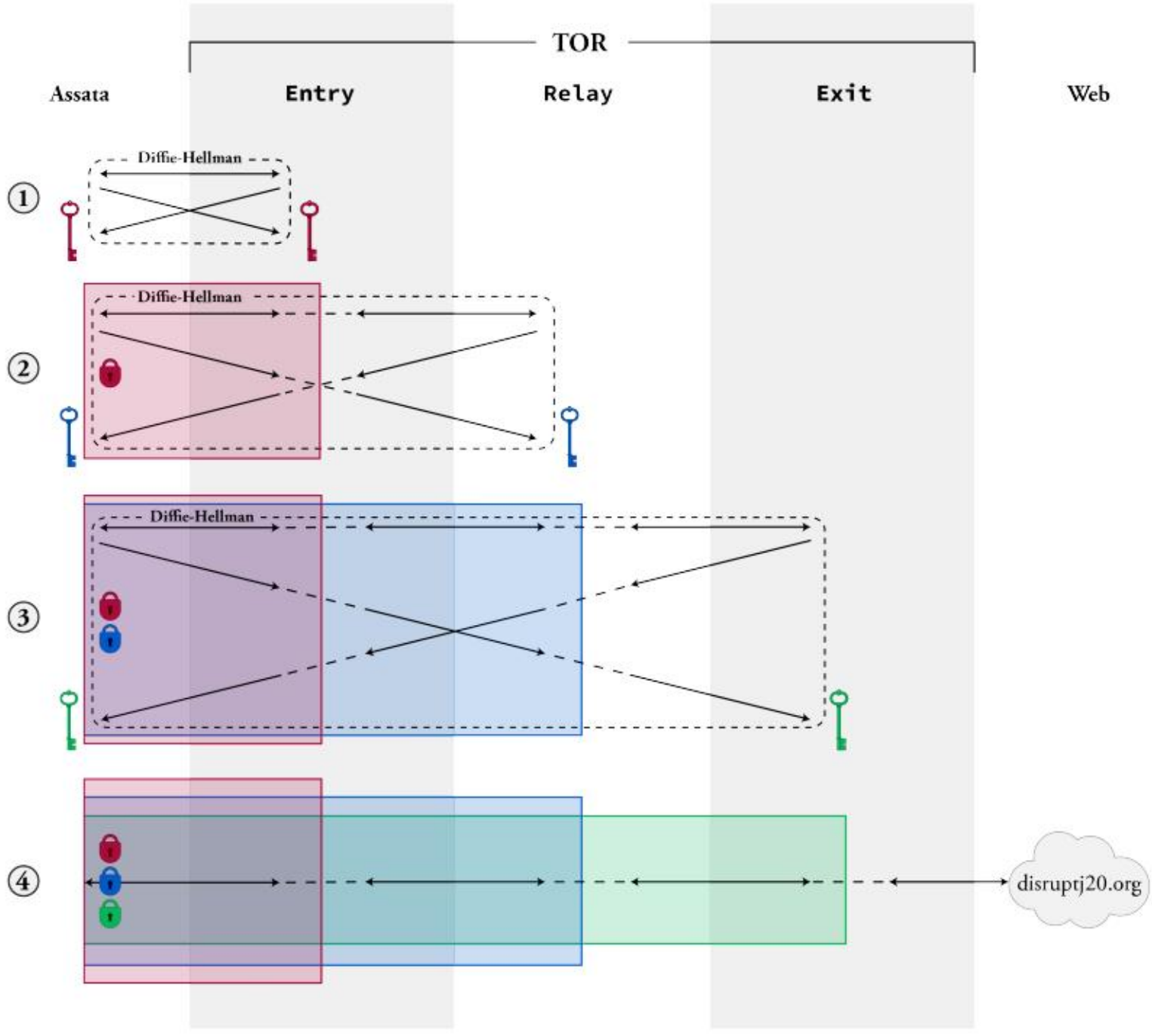
Not Trusting the Middle Man: The Onion Router

The Onion Router, or Tor, is a means of accessing the internet anonymously while sidestepping trust issues and gets its name from using *layers* of encryption (like the layers of an onion). Rather than using one middle man with whom you trust all your information, you use (at least) three intermediaries, chosen at random from a selection of thousands of volunteer servers (illustrated below). Traffic through this path of intermediaries is encrypted so that the first (entry) node only knows that you are accessing the internet via Tor, the second (relay) node only knows that someone is accessing something on the internet via Tor (but not who specifically or what specifically), and the last (exit) node only knows that a certain web page (for example) is being requested by a Tor user (but not which Tor user).



The way this is done is by Diffie-Hellman key exchanges first with the entry node, then with the relay node, and finally with the exit node as follows (and illustrated below). (1) Assata establishes a cryptographic key that she shares with the entry node (which we will call the entry key, in red). This establishes an encrypted communication channel between Assata and the entry node. (2) Assata uses this encrypted channel to communicate with the relay node via the entry node. The traffic between the entry and relay nodes is not encrypted, but Assata uses the channel via the entry node

to establish a cryptographic key that Assata shares with the relay node (the relay key, in blue). All that the relay node knows is that it is setting up a shared key with some Tor user but not the identity of that Tor user. (3) This process is repeated to establish an encryption key that Assata shares with the exit node (the exit key, in green). (4) This created a sequence of keys (red, blue, green) that allow for encryption between Assata and the entry, relay, and exit nodes, respectively.

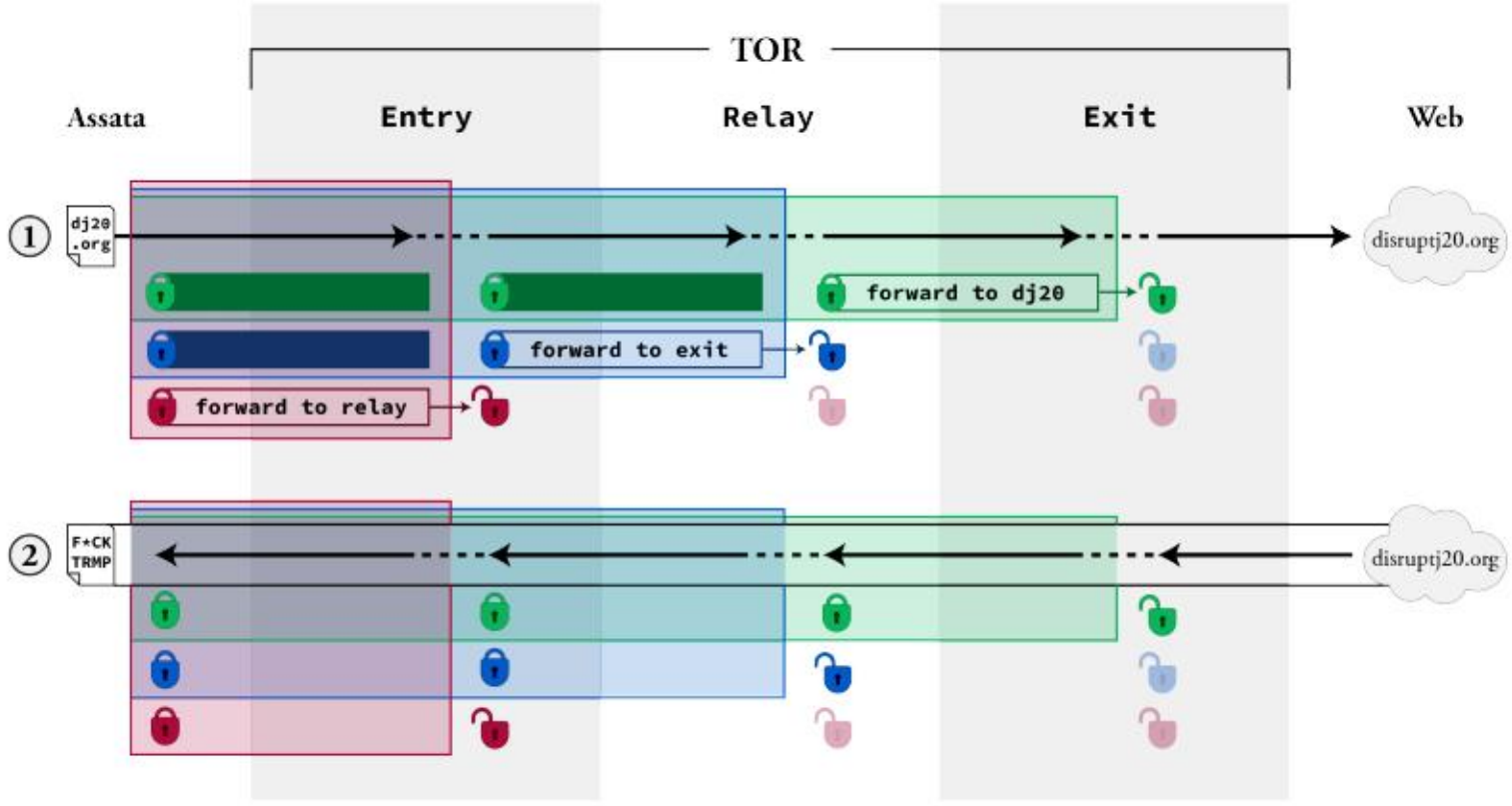


How Tor establishes keys using rounds of the Diffie-Hellman key exchange

For Assata to send a request to disruptj20.org, she encrypts the request, addressed to disruptj20.org, with the green key and addresses this to the exit node; she wraps this in a message addressed to the relay node and encrypts this with the blue key; she wraps this in a message addressed to the entry node and encrypts this with the red key. The message is sent to the red node. The first *layer* of encryption is removed by the entry node (with the red key that the entry node shares with Assata), revealing a message addressed to the relay node. The second *layer* of encryption is removed by the relay node (with the blue key that the relay node shares with Assata), revealing a message addressed to the exit node. The third *layer* of encryption is removed by the exit node (with the green key that the exit node shares with Assata), revealing a message addressed to disruptj20.org, which the exit node forwards along. This is illustrated below (1).

For disruptj20.org to send information back to Assata, the web server sends the information back to the exit node. The exit node encrypts with the green key and sends it to the relay node. The relay

node encrypts with the blue key and sends it to the entry node. The entry node encrypts with the red key and sends it to Assata. Assata can remove all three *layers* of encryption because she has all the necessary keys. This is illustrated below (2).



How data traverses the Tor network

In order to re-create your path through the Tor network and therefore your web request, your adversary would need to control all three nodes that you select as your entry, relay, and exit nodes. Even an adversary who controls 80 percent of the Tor network would only have a 50 percent chance of controlling all three nodes that you select. Since there are thousands of Tor nodes (that anyone can volunteer to operate), this is unlikely.

An alternate attack that an adversary could take would be a *confirmation attack*. In this scenario, the adversary is trying to prove that you have visited a particular web service. If they can access your web traffic (e.g., through your ISP) and the web traffic of the target web service (through legal or extralegal means), then your adversary may be able to match up your use of Tor to access the web service from Tor based on their timing. This type of correlation was used in the case against Jeremy Hammond, convicted for hacking activities conducted by the activist collective Anonymous.

Other attacks have been made on Tor too, but the Tor project is very responsive to improving their technology and security. We discuss obstacles to anonymous browsing below and pitfalls a user may run into as well as best practices when trying to access the web anonymously in the chapter "Protecting Your Identity."

Use and Prevention of Anonymous Browsing Technologies

Many people in countries where censorship of the internet is common, such as China and Iran, use VPNs and Tor to access the uncensored web. On the other hand, evidence of VPN traffic can

be gleaned from the metadata of internet communications, and governments can use this to block all such communications, as has been done in China and Syria in their censorship efforts. Other countries, such as Iran, are known for blocking access to specific VPN providers that are not sanctioned by the government.

Tor as a whole can be blocked from use (e.g., by a government), since Tor nodes are publicly listed. This is done by simply blocking all traffic addressed to the Tor nodes. This is overcome by the use of *bridges*, a set of Tor nodes that are not publicly listed, which you use in lieu of a publicly listed entry node. To get access to a small set of bridge nodes, you need to email the Tor project from a restricted email account (e.g., Google, Riseup!, or Yahoo!) to request one. Tor can also be blocked by packet inspection—that is, by looking at the metadata of the communications (as with VPN traffic). The Tor project makes this process challenging by using methods of obfuscating Tor internet traffic so that it doesn't look like Tor traffic.

VPN and Tor are also used to gain access to particular sites that might not be available in your jurisdiction because of a choice of the web host. This is common for many media platforms such as Hulu and Netflix. To this end, companies will often block access to content from known VPN service providers or from Tor exit nodes.

In Context: Disruptj20

On January 20, 2017, mass protests erupted around the inauguration of the forty-fifth president of the United States. Much of the organizing for those events was coordinated on the website disruptj20.org. In August 2017, it came to light that the US Department of Justice had issued a warrant to the disruptj20.org web host DreamHost requesting, among other items, "all HTTP request and error logs," which would include the IP addresses of all individuals, purported to be 1.3 million people, who visited the website, along with which subpages they visited, how often they did so, and any text a visitor may have typed into the web page.

Of course, anonymous browsing technologies would have protected the IP addresses of visitors to that site.

What to Learn Next

- Protecting Your Identity

External Resources

- The Great Firewall of China keeps track of which and how many or how many times sites are censored in China.
- Search Warrant to DreamHost, August 2017.

Media Attributions

- anonymous-browsing-vpn © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- anonymous-browsing-tor © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- anonymous-browsing-TOR-1-keyexchange © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- anonymous-browsing-TOR-2-data-transfer © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license

PART 2: DIGITAL SUPPRESSION OF SOCIAL MOVEMENTS (IN THE US)

Mechanisms of Social Movement Suppression

Before reading this chapter, it's a good idea to (re)read the introductory chapter "Why Digital Security?" and recall that our primary focus in this book is US social movements.

What You'll Learn

1. How the US suppresses social movements
2. What COINTELPRO (Counter Intelligence Program) was and the mechanisms employed to suppress social movements of that era

The US has a long history of interference, including on its own soil in the form of suppressing the efforts of social movements and, in particular, liberatory and leftist social movements. Labor organizing, independence, civil rights, and environmental movements have all been subject to opposition by the US government, often at the behest of or in cooperation with large corporations.

In trying to grapple with the risks associated with a social movement *not* attending to digital security, it is helpful to look at how the State has interfered with social movements in the past. This history can be overwhelming, and it can be tempting to dismiss this as something that has happened in the past but that is not happening now. Going through this history can also lead to defeatism, especially in light of the additional digitally enhanced tools that the State can employ against an adversary (perceived or actual).

However, since we should not condemn ourselves to repeat mistakes of the past, we need to attend to enough history to learn appropriate lessons so that our movements may be successful moving forward. In order to do so without turning this into a history textbook, we draw on the scholarship of Jules Boykoff, who categorized the ways in which the US has messed with social movements in the twentieth century. Boykoff enumerated twelve *modes of suppression*, which we compress to seven in this presentation.

Understanding these historical modes will allow us to predict how digital surveillance could support or enhance those modes, as we will discuss in the chapter "Digital Threats to Social Movements." But more importantly, we will be able to see how encryption and attending to digital hygiene can

protect social movements against (some of) these oppositional forces, as we will cover in part 3 of this book.

Modes of Suppression

These are seven ways in which the US has suppressed and continues to suppress social movements, each with a few examples of its use. Unfortunately, these examples are far from exhaustive.

1. Direct Violence

Beatings, bombings, shootings, and other forms of violence are carried out by the State or other institutions or nodes of power against dissident citizens.

This may be the result of the policing of large groups (such as when the Ohio Army National Guard fired at students during an antiwar protest at Kent State University, killing four people and injuring nine others) or targeted assassinations (such as in the FBI-organized night-raid shooting of Black Panther Party leader Fred Hampton). To risk an understatement, these actions discourage participation in social movements for fear of life and limb.

2. The Legal System

The legal system allows for harassment arrests, public prosecutions and hearings, and extraordinary laws that are used to interfere with individuals in biased fashions. The State arrests activists for minor charges that are often false and sometimes based on obscure statutes that have remained on the books, buried and dormant but nevertheless providing vessels for selective legal persecution. Public prosecutions and hearings can land dissidents in jail or consume their resources in legal proceedings that sidetrack their activism and demobilize their movements. Current supporters and potential allies are discouraged from putting forth dissident views. Prosecution and hearings publicized in the mass media reverberate outward into the public sphere. Another form of legal suppression, the State promulgates and enforces exceptional laws and rules to tie up activists in the criminal justice labyrinth. This is the legal system being used to squelch dissent.

Controversial stop-and-frisk programs allow police officers to briefly detain and at times search people without probable cause. Free-speech zones greatly limit the time, place, and manner of protests. Those arrested at First Amendment–protected protests on the inauguration of Donald Trump faced public prosecutions that were unlikely to ever reach a conviction. And certain crimes, such as arson or the destruction of property, are elevated to terrorism when they are accompanied by a political motive and allow for the State to greatly increase the punishment doled out. Other laws

are specifically tailored to prevent activism, such as "ag-gag laws," which criminalize the filming of agricultural operations (which is done to expose the abuse of animals).

3. Employment Deprivation

One's political beliefs can result in threats to or actual loss of employment due to one's political beliefs or activities. Some dissidents are not hired in the first place because of their political beliefs. This is typically carried out by employers, though the State can have a powerful direct or indirect influence.

Recently, we have seen university professors forced out of their jobs or have job offers revoked, as with Steven Salaita, whose offer of employment as a professor of American Indian studies was withdrawn following university donors' objections to a series of Tweets critical of Israel and Zionism. For several years (but since struck down by a federal court), government contractors in Texas were required to sign a pledge to not participate in the pro-Palestinian Boycott, Divestment, and Sanction movement or else they would have their contracts canceled; this resulted in the firing of an American citizen of Palestinian descent who worked as a school speech pathologist and refused to sign the statement.

4. Conspicuous Surveillance

Conspicuous surveillance aims primarily not to collect information (which is best done surreptitiously) but to intimidate. This is intended to result in a *chilling effect*, in which individuals guard their speech and action out of fear of reprisal. It may drive away those engaged in activism or make it difficult to encourage new activists. Although the chilling effect has been deemed unconstitutional, it is difficult to prove harm in a court (as required), so it is a safe means of suppression (from the perspective of the surveiller).

The FBI has a long history of "knock and talks" or simply visiting the houses of dissidents and activists (and those of their families and employers) to "have a chat" in order to let people know that they are being watched.

5. Covert Surveillance

Surveillance might be concentrated or focused, as with the use of spies, targeted wiretaps, and subpoenas or warrants for data; the use of infiltrators (covert agents who become members of the target group); or the use of informants (existing group members who are paid or threatened in order to extract information). Surveillance might also be diffused, such as the accumulation, storage, and analysis of individual and group information that is obtained through internet monitoring, mail openings, and other mass-surveillance techniques.

The scale of the FBI informant program is sizeable, having over fifteen thousand informants in 2008. In the wake of 9/11, the FBI and large law enforcement agencies such as the NYPD turned their intelligence programs against Muslim American communities. This included the close surveillance of lawyers, professors, and the executive director of the largest Muslim civil rights organization in the US (the Council on American-Islamic Relations) by the FBI. The NYPD singled out mosques and Muslim student associations, organizations, and businesses through the use of informants, infiltrators, and surveillance. The NYPD's supposed rationale was to identify potential "terrorists" by looking for "radicalization indicators," including First Amendment–protected activities such as "wearing traditional Islamic clothing [and] growing a beard" and "becoming involved in social activism."

6. Deception

Snitch jacketing is when a person (often an infiltrator) intentionally generates suspicion that an authentic activist is a State informant or otherwise maliciously present in a social movement group. Infiltrators or informants who are in place to encourage violence or illegal activities or tactics (rather than simply report on activities) are known as agent provocateurs and do so in order to legally entrap or discredit the group. False propaganda is the use of fabricated documents that are designed to create schisms or undermine solidarity between activist organizations. These controversial, offending, and sometimes vicious documents are meant to foment dissension within and between groups.

FBI infiltrators have acted as agent provocateurs by leading them down a path to illegal activity that they would not have otherwise followed. Mohamed Mohamud was an Oregon State University student who was contacted by an undercover FBI agent who over a period of five months suggested and provided the means to bomb the lighting of the Portland Christmas tree on November 26, 2010. The bomb was a fake, but Mohamud was sentenced to thirty years of imprisonment.

Eric McDavid spent nearly nine years in prison for conspiring to damage corporate and government property after a paid FBI informant acted as an agent provocateur, encouraging McDavid's group to engage in property destruction and providing them with bomb-making information, money to buy the raw materials needed, transportation, and a cabin to work in. McDavid's conviction was overturned due to the FBI failing to disclose potentially exculpatory evidence to the defense.

7. Mass Media Influence

There are two major types of mass media manipulation: (1) story implantation, whereby the State makes use of friendly press contacts who publish government-generated articles verbatim or with minor adjustments, and (2) strong-arming, whereby the State intimidates journalists or editors to withhold unwanted information from reaching publication. In addition to that, mass media deprecation portrays dissidents as ridiculous, bizarre, dangerous, or otherwise out of step with

mainstream society. This is often not so much due to conspiracy as to dutiful adherence to journalistic norms and values. Mass media underestimation occurs when activists and the State come up with discrepant estimates of crowd sizes for protests, marches, and other activities, with the mass media tending to accept the State's lower numbers. The mass media may also falsely balance dissidents with counterdemonstrators. Many dissident efforts never make it onto the mass media's agenda or are buried in the minor sections of the newspaper. Not only the State but also powerful media organizations or individual owners are able to carry out this type of suppression.

Following the invasion of Iraq after 9/11, antiwar sentiment was consistently downplayed through underreporting. As just one example, the September 2006 antiwar protests that saw more than two hundred thousand people take to the streets across the US was reported by the *Oregonian* in this way: A one-hundred-thousand-strong antiwar protest in Washington, DC, was reported on page 10, along with an article on a Portland protest. The article estimated one hundred people at the protest, even though aerial evidence pointed to over three thousand. A counterdemonstration to the Washington, DC, antiwar protest was covered on page 2 with a larger photo and longer text, even though only four hundred people attended.

Information Technology Interference

This resource would be lacking if we didn't talk about censorship and other interference with information technology. It is an additional mode of suppression with particular relevance to the Information Age that dovetails with deception and mass media influence, wherein access to the internet or related infrastructure is blocked or otherwise denied to social movements—for example, cutting off internet or mobile network access during a protest, censoring certain sites or types of internet traffic, or shutting down a social movement group's website.

Boykoff does not include this in his catalog of suppression, since its use is not widespread within the US by the US largely due to the country's constitutional protections. However, its use is widespread around the globe. Governments have been known to cut off internet access at the country level (such as the week-long total shutdown of the internet in Iran as a means to suppress protests) or limit access to certain sources (such as the Great Firewall of China blocking Google, Facebook, Twitter, and Wikipedia). US companies also participate in this, complying with foreign censorship: Zoom (a web conferencing service) shut down the accounts of three activists at the behest of China, who had planned online events to commemorate the Tiananmen Square massacre.

In Context: COINTELPRO and the COINTELPRO Era

From the 1950s through the 1970s, the FBI conducted a set of secret, domestic counterintelligence activities, which became known as COINTELPRO, under the leadership of FBI director J. Edgar Hoover. Originating with US government anticommunist programs during the Red Scare,

COINTELPRO aimed to "disrupt, by any means necessary," the organizing and activist efforts of the Black Power, Puerto Rican independence, civil rights, and other movements. With respect to civil rights and Black Power movements (including the activities of Martin Luther King Jr.), COINTELPRO was ordered to "expose, disrupt, misdirect, discredit, or otherwise neutralize the activities of black-nationalist, hate-type organizations and groupings, their leadership, spokesmen, membership and supporters to counter their propensity for violence and civil disorder."

COINTELPRO was exposed through the theft of boxes full of sensitive FBI paperwork obtained through a burglary in 1971 by the Citizens' Commission to Investigate the FBI, the members of which only went public in the wake of Ed Snowden's disclosures, with remaining COINTELPRO documents coming to light through Freedom of Information Act (FOIA) requests. The commission's leak resulted in the formation of the US Senate's Church Committee in 1975, which castigated the FBI for the "domestic intelligence activities [that] have invaded individual privacy and violated the rights of lawful assembly and political expression" and ultimately shut down COINTELPRO. The Church Committee prefaced their admonishment this way:

> We have seen segments of our Government, in their attitudes and action, adopt tactics unworthy of a democracy, and occasionally reminiscent of totalitarian regimes. We have seen a consistent pattern in which programs initiated with limited goals, such as preventing criminal violence or identifying foreign spies, were expanded to what witnesses characterized as "vacuum cleaners," sweeping in information about lawful activities of American citizens. The tendency of intelligence activities to expand beyond their initial scope is a theme which runs through every aspect of our investigative findings. Intelligence collection programs naturally generate ever-increasing demands for new data. And once intelligence has been collected, there are strong pressures to use it against the target.

All the following modes of suppression were used by the FBI or support partners as part of COINTELPRO or against COINTELPRO targets:

1. *Direct violence*

The murder of Fred Hampton mentioned above was a joint operation of the FBI and Chicago Police Department. Fred Hampton was the chairman of the Black Panther Party, a revolutionary socialist political organization of the late 1960s through 1970s that aimed to protect Black Americans and provide social programs (such as free breakfast and health clinics). The Black Panther Party (BPP) was labeled as a "Black nationalist hate group" by the FBI for inclusion as a COINTELPRO target. Hampton's assassination was supported by other modes of suppression, including the following:

- *Covert surveillance*. A paid FBI infiltrator provided intelligence that made the raid leading to Hampton's murder possible.
- *Deception*. The same infiltrator created an atmosphere of distrust and suspicion within the

BPP in part by snitch jacketing other members of the BPP.

- *Mass media influence*. Following Hampton's assassination, BPP members were depicted as "folk devils," with media representations becoming increasingly distorted.

2. *The legal system*

Communist and Black Panther Party member and COINTELPRO target Angela Davis was charged with "aggravated kidnapping and first-degree murder" in the death of a judge in California who was kidnapped and killed during a melee with police, even though Davis was not on the scene. California held that the guns used by the kidnappers were owned by Davis and considered "all persons concerned in the commission of a crime, whether they directly commit the act constituting the offense…, principals in any crime so committed." Davis could not be found at the time and was listed by J. Edgar Hoover on the FBI's "Ten Most Wanted Fugitives" list. Months later, Davis was apprehended and spent sixteen months in prison awaiting a trial in which she was found not guilty.

3. *Employment deprivation*

Prior to Davis's battle with the legal system, she was fired from her job as a philosophy professor at UCLA for her Communist Party membership in her first year of employment in 1969, calling her unsuitable to teach in the California system. The firing was at the request of then California governor Ronald Reagan, who pointed to a 1949 law outlawing the hiring of Communists in the University of California. This highlights the lingering Red Scare era or McCarthyism that ran through the 1940s and 1950s. The FBI supported the demonization of communism through a predecessor program to COINTELPRO: COMINFIL (Communist Infiltration) probed and tracked the activities of labor, social justice, and racial equality movements.

4. *Conspicuous surveillance*

Among the first round of FBI documents to come to light about COINTELPRO was a document memo called "New Left Notes." The New Left refers to a broad political movement of the 1960s and 1970s, groups of which campaigned on social issues such as civil, political, women's, gay, and abortion rights. In discussing how to deal with "New Left problems," the FBI Philadelphia field office memo says, "There was a pretty general consensus that more interviews with these subjects and hangers-on are in order for plenty of reasons, chief of which are it will enhance the paranoia endemic in these circles and will further serve to get the point across [that] there is an *FBI Agent behind every mailbox*. In addition, some will be overcome by the overwhelming personalities of the contacting agent and volunteer to tell all—perhaps on a continuing basis."

5. *Covert surveillance*

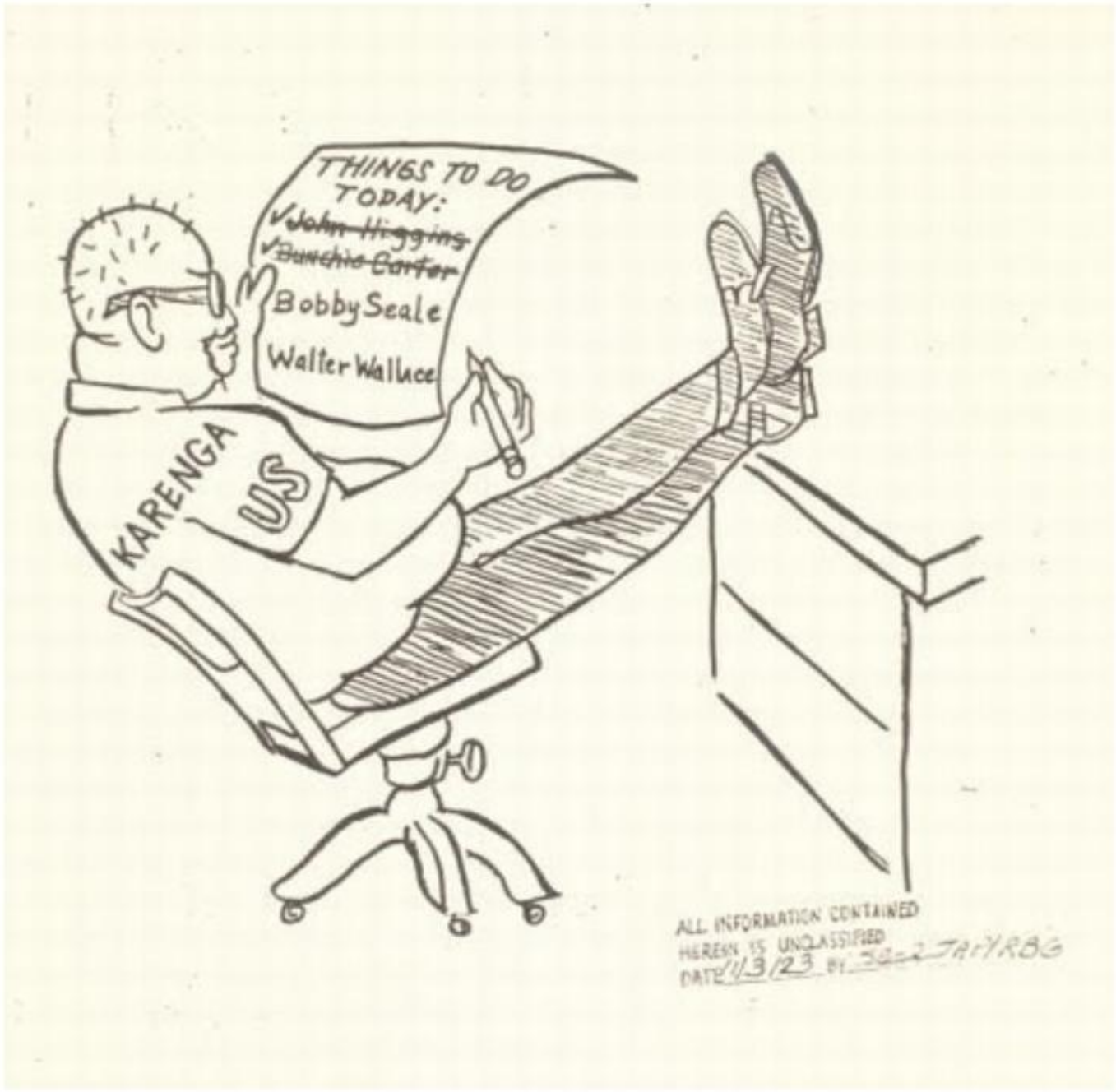
The Church Committee reported enumerated covert surveillance that "was not only vastly excessive in breadth… but also often conducted by illegal or improper means." Notably, both the CIA and FBI had "mail-opening programs" that indiscriminately opened and photocopied letters mailed in the US on a vast scale: nearly a quarter of a million by the CIA between 1953 and 1973 and another 130,000 by the FBI from 1940 to 1966. Further, both the CIA and the FBI lied about the continuation of these programs to President Nixon.

6. *Deception*

The FBI often sent fake letters or flyers in order to drive wedges between otherwise aligned groups. Below is an example of a cartoon drawn by FBI operatives as a forgery of movement participants and is intended to incite violence between the Black nationalist groups Organization Us (coestablished by Maulana Karenga) and the Black Panther Party (with prominent members Huey Newton, David Hilliard, Bobby Seale, John Huggins, and Bunchy Carter). The cartoon depicts BPP being knocked off by Karenga. The FBI later claimed success in the deaths of two BPP members by US gunmen.

7. *Mass media influence*

Manipulation of the mass media was an explicit tenet of the FBI's COINTELPRO against the New Left. According to the Church Committee, "Much of the Bureau's propaganda efforts involved giving information to 'friendly' media sources who could be relied upon not to reveal the Bureau's interests. The Crime Records Division of the Bureau was responsible for public relations, including all headquarters contacts with the media. In the course of its work (most of which had nothing to do with COINTELPRO), the Division assembled a list of 'friendly' news media sources—those who wrote pro-Bureau stories. Field offices also had 'confidential sources' (unpaid Bureau informants) in the media, and were able to ensure their cooperation."



FBI cartoon to incite violence

What to Learn Next

- Digital Threats to Social Movements
- Defending against Surveillance and Suppression

External Resources

- Boykoff, Jules. *Beyond Bullets: The Suppression of Dissent in the United States*. AK, 2007. Oakland, CA.
- Wikipedia. "2019 Internet Blackout in Iran." December 18, 2020.

- Shieber, Jonathan. “Zoom Admits to Shutting Down Activist Accounts at the Request of the Chinese Government.” TechCrunch, June 2020.
- Duane de la Vega, Kelly, and Katie Galloway. “Eric and ‘Anna.’” Field of Vision, November 19, 2015.
- US Senate Select Committee to Study Governmental Operations with Respect to Intelligence Activities. *Intelligence Activities and the Rights of Americans*. Report No. 94-755. Washington, DC: US Government Printing Office, 1976.
- Churchill, Ward, and Jim Vander Wall. *The COINTELPRO Papers: Documents from the FBI’s Secret Wars against Domestic Dissent*. South End, 1990. Cambridge, MA.

Media Attributions

- things-to-do © Federal Bureau of Investigation is licensed under a Public Domain license

Digital Threats to Social Movements

We recommend that you read the chapters "What Is Encryption?" and "Metadata." After reading this section, be sure to read the chapter "Defending against Surveillance and Suppression."

What You'll Learn

1. What threat modeling is
2. Who is engaged in surveillance and the strategies they use
3. Examples of tactics and programs used in surveillance

Social movements challenging powerful individuals, organizations, and social structures face a broad range of surveillance risks. Specific threats vary widely in technical sophistication, likelihood, and potential for harm. **Threat modeling** is a process whereby an organization or individual considers their range of adversaries, estimates the likelihood of their various data and devices falling victim to attack, and finally considers the damage done if attacks were to succeed. (And then they work to protect the data that is most at risk and that would be the most damaging to lose or have accessed.)

We think about surveillance in the following order to inform how to protect oneself:

- Who is your **adversary**? Is it a neighborhood Nazi who is taking revenge on you for your Black Lives Matter lawn sign? Is it an oil corporation that is fighting your antipipeline activism? Is it the US government trying to prevent you from whistleblowing? By understanding who your adversary is, you can surmise their resources and capabilities.
- Is your adversary going after you in particular, are they trying to discover who you are, or are they collecting a lot of information in the hopes of getting your information? What **surveillance strategy** are they likely to employ? This will help you understand what *type* of data and *where* your data may be at risk.
- What particular **surveillance tactics** will your adversary employ to get that desired data? This will help you understand *how* to protect that data.

We examine surveillance risks starting with the adversary because it is strategic to do so. No one can achieve perfect digital security, but one can be smart about where to spend one's effort in protecting oneself against surveillance. In an actual threat-modeling discussion within an organization or social movement, *who* potential adversaries are is often more readily apparent than *how* such adversaries

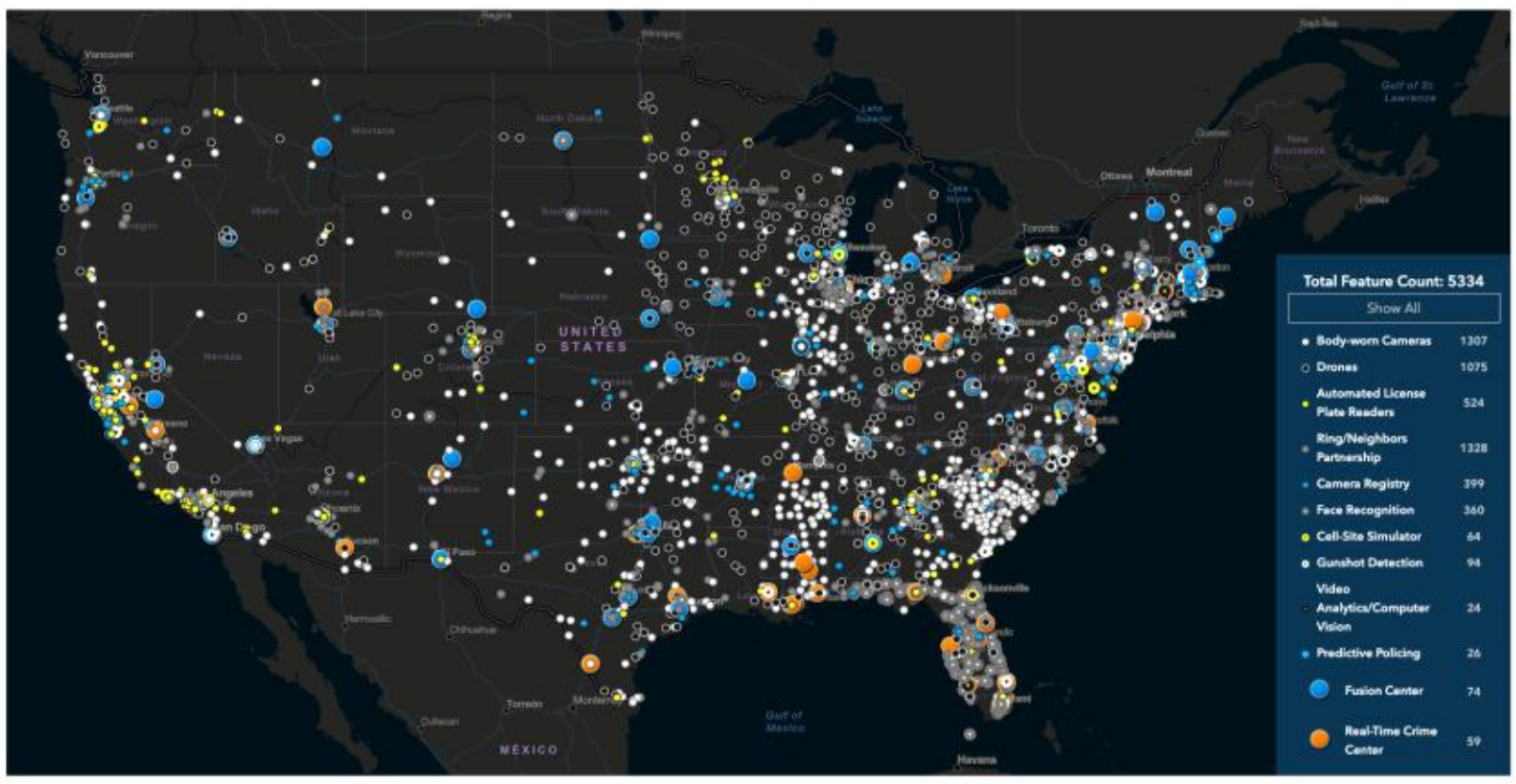
would carry out attacks. Who the adversary is then informs the range of techniques available to that adversary (depending on their available resources and legal authorities) and, in turn, what protective behaviors and technologies the organization can employ.

Surveillance Adversaries

We generally think of adversaries in terms of what resources they have available to them. For the purposes of this book, we will limit ourselves to three adversary categories:

Nation-states have access to the most resources in a way that may make it seem like their surveillance capabilities are limitless. Even so, they are unlikely to be able to break strong encryptions. Here, we think of the National Security Agency (NSA) as the entity with access to the most sophisticated surveillance capabilities. The disclosures by Edward Snowden in 2013 give the most comprehensive window into nation-state-level capabilities and are searchable in the Snowden Surveillance Archive.

Large corporations and **local law enforcement** are often heavily resourced and share information with each other but don't necessarily have access to the capabilities of nation-states. However, the use of technology to aid in surveillance is widespread among law enforcement agencies in the US, as illustrated below in this screenshot from Electronic Frontier Foundation's Atlas of Surveillance.



Atlas of police surveillance technologies

Individuals have the least resources but might know you personally and so be able to more effectively use social engineering to obtain your data.

Note that techniques available to lower-resourced adversaries are also available to higher-resourced adversaries. As an example, corporations and law enforcement employ informants and infiltrators,

who may be individuals who know you personally. Also, while more sophisticated surveillance capabilities are not usually available to lower-resourced adversaries, it is not always the case: police departments in large cities may have access to nation-state-level resources (e.g., through data sharing that is facilitated by fusion centers), or a particularly skilled neighborhood Nazi may possess advanced hacking skills that enable some corporate-level attacks.

So while these categories are not sharply defined, they can act as a starting point for understanding the risks and focusing on your adversaries' *most likely* strategies and your *most likely* weaknesses.

Surveillance Strategies

There are two broad strategies of surveillance: mass-surveillance and targeted surveillance.

Mass surveillance collects information about whole populations. This can be done with the purpose of trying to better understand that population. For example, the collection and analysis of health-related data can help identify and monitor emerging outbreaks of illnesses. Mass surveillance may also be used as a strategy to identify individuals of interest within the surveilled population. For example, video feeds from security cameras can be used to identify those who engaged in property damage. Or mass surveillance may garner information about a particular individual. For example, information collected from the mass deployment of license-plate cameras can be used to track the movements of that particular individual.

Targeted surveillance only collects information about an individual or a small group of individuals. For example, wiretapping intercepts the communications of a particular individual. Targeted surveillance allows for the existence of prior suspicion and can (conceivably) be controlled—for example, law enforcement obtaining a warrant based on probable cause before intercepting someone's mail.

Historically, there was a clearer divide between targeted and mass surveillance. However, in the digital age, many tactics of targeted surveillance can be deployed on a mass scale, as we will discuss further. In addition to this classic division of surveillance strategies, we draw attention to a bigger strategy that is unique to the digital age.

Collect-it-all may simply be viewed as mass surveillance on steroids but goes far beyond what may have historically been viewed as mass surveillance. Where mass surveillance may encompass things like security cameras, the monitoring of bank transactions, and the scanning of emails, collect-it-all aims to vacuum up any information that is digitized. Collect-it-all goes further: for any information that isn't online or available (e.g., video from truly closed-circuit security cameras), it digitizes it and then collects it. Collect-it-all is infamously attributed to General Keith Alexander, former director of the NSA, whose mass-surveillance strategies were born in post-9/11 Iraq and were described as follows: "Rather than look for a single needle in the haystack, his approach was, 'Let's collect the whole haystack. Collect it all, tag it, store it… And whatever it is you want, you go searching for it." This

is likely the inspiration for many of the NSA programs that were uncovered by Edward Snowden that we highlight below.

Different adversaries tend to deploy different surveillance strategies, or rather, lower-resourced adversaries tend to be limited in their strategies, as depicted:

Adversary (Increasing Resources and Sophistication)	Surveillance Strategies: Targeted	Mass	Collect-it-all
Nation State	X	X	X
Corporation	X	X	
Individual	X		

Adversaries and their surveillance strategies

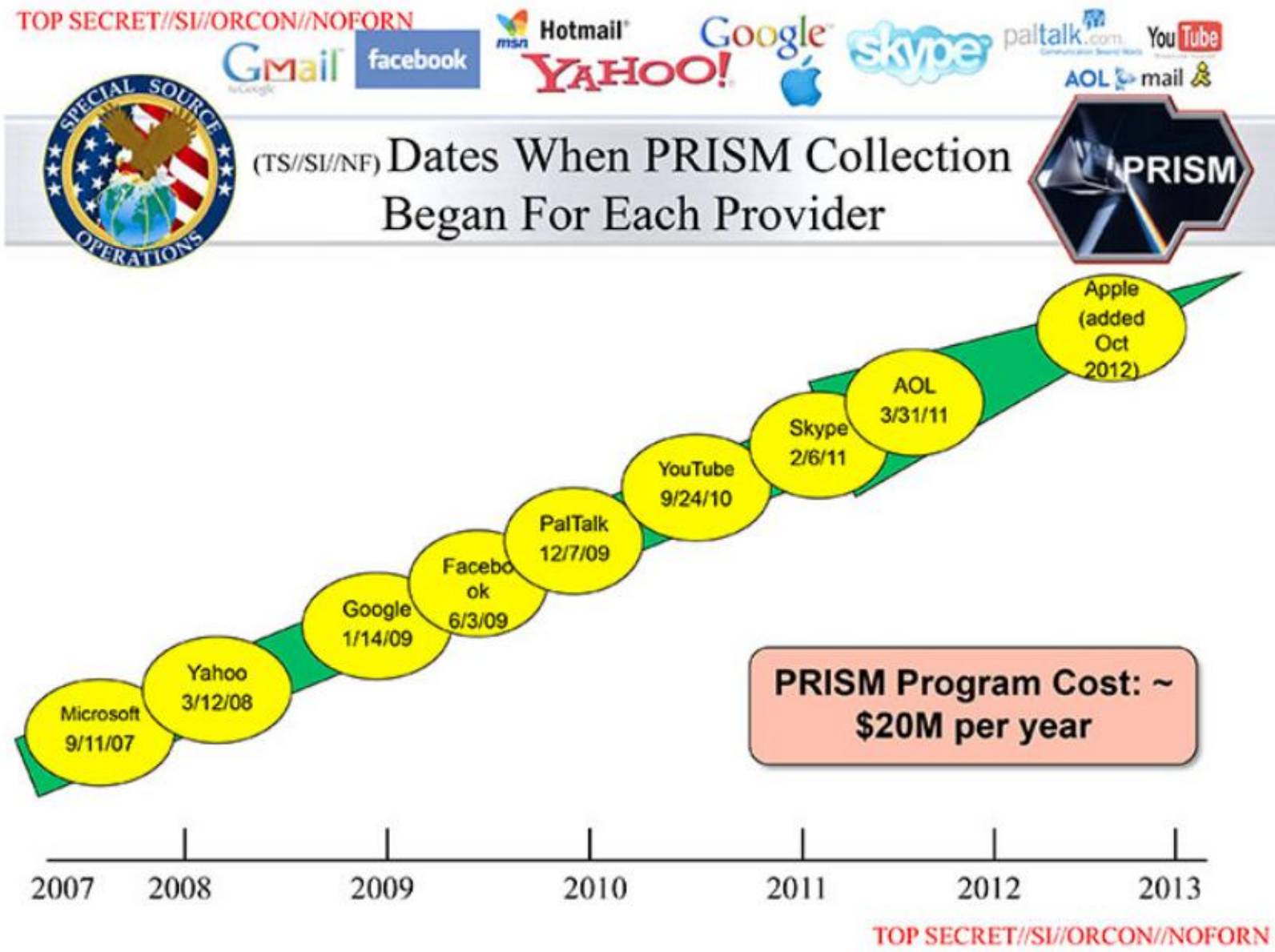
Surveillance Tactics

To go over all the surveillance tactics that are available to adversaries at all levels would fill an encyclopedia. Here we illustrate a few examples of surveillance programs and tactics that support the surveillance strategies above. We illustrate these programs (below) according to the minimum level of sophistication required to use the tactic and the number of people whose information would be collected via these means.

Mass Interception and Collection of Data

We start with what most people probably think of when they think of mass surveillance: the interception and possible recording of vast amounts of communications. Many mass interception programs were uncovered as part of Edward Snowden's disclosures in 2013. **STORMBREW**, **FAIRVIEW**, and **BLARNEY** are three such programs through which the NSA collects data in transit by partnering with telecommunications companies and getting access to data passing through submarine data cables. This allows for the collection of any unencrypted content and all associated metadata while it is in transit from origin to destination. However, these programs cannot see content that is typically encrypted in transit, such as email or files in cloud storage. The **PRISM** program is a partnership of the NSA with various internet companies (such as Google, Microsoft, and Facebook, as illustrated below) to allow NSA access to data held on company servers. That is, if the information was encrypted in transit and so not collectible via STORMBREW, FAIRVIEW, and

BLARNEY, then the NSA can get it via PRISM—unless the information is encrypted on the company servers with a key that the user controls.



PRISM Collection, National Security Agency

Aggregating and Analyzing Data

Once you have a whole lot of surveillance data, what do you do with it? Surely the man will not be able to find my tiny little needle in that massive haystack. This is where data mining, from basic search to (creepy) predictive machine-learning models, comes in to make vast amounts of mass-surveillance data (including from disparate sources) useful to powerful adversaries.

The most basic functionality is the ability to search—that is, given a large amount of data, retrieving the data of interest, such as that related to a particular person. **XKEYSCORE** acts as a Google-type search for the NSA's mass-surveillance data stores. While the functionality is basic, the sheer amount of information it has access to (including that from the NSA programs highlighted above) places XKEYSCORE (and any related program) as accessible to only the most powerful adversary.

On the other hand, **Dataminr** searches publicly available data (such as social media posts) to uncover and provide details of emerging crises (such as COVID-19 updates and George Floyd protests) to their customers, which include newsrooms, police departments, and governments, through both automated (software) and manual (human analysis) means. Dataminr and other social media–monitoring platforms, of which there are dozens if not hundreds, have come under fire for

their surveillance of First Amendment–protected speech, most notably of the Movement for Black Lives. In several instances, Twitter and Facebook cut off social media–monitoring companies' easy access to their data after public outcry over misuse.

Going further, **Palantir** is one of many policing platforms that supposedly predict where policing is needed, be that a street intersection, a neighborhood, or an individual. In reality, these platforms do little but reinforce racist norms. Predictive policing platforms use current police data as the starting point and tend to send police to locations that police have been in the past. However, communities of color and impoverished neighborhoods are notably overpoliced, so predictive models will simply send police to these areas again, whether or not that is where crimes are being committed.

Going further still, **EMBERS** (Early Model Based Event Recognition Using Surrogates) has been used since 2012 to predict "civil unrest events such as protests, strikes, and 'occupy' events" in "multiple regions of the world" by "detect[ing] ongoing organizational activity and generat[ing] warnings accordingly." The warnings are entirely automatic and can predict "the when of the protest as well as where of the protest (down to a city level granularity)" with 9.76 days of lead time on average. It relies entirely on publicly available data, such as social media posts, news stories, food prices, and currency exchange rates.

Targeted Collection of Data

Another surveillance tactic that comes to mind is that of the wiretap. However, the modern equivalent is a lot easier to enable than the physical wire installed on a communication cable, from which wiretap gets its name. One modern version is the *cell site simulator* (**CSS**), which is a miniature cell tower (small enough to be mounted on a van). To cell phones in the vicinity, this tower provides the best signal strength, and so they will connect to it. At the most basic level, a CSS will uncover the identities of the phones in the area. (Imagine its use at the location of a protest.) Different CSSes have different capabilities: Some CSSes simply pass on the communications to and from the broader cell network while gaining access to metadata. In some cases, CSSes are able to downgrade service—for example, from 3G to GSM—removing in-transit encryption of cell communications with the service provider and giving access to message content. In other cases, CSSes can block cell communications by having phones connect to it but do not pass information onto the greater cell phone network. CSSes are fairly commonly held by law enforcement agencies (as illustrated in the map at the start of this chapter).

Surveillance equipment, including CSSes and high-resolution video, can also be mounted to surveillance *drones* or *unmanned aerial vehicles* (**UAV**s), which can greatly increase the scope of surveillance from a few city blocks to a whole city. This is one example where tactics of targeted surveillance are expanded toward a mass level. Persistent Surveillance Systems has pitched the use of UAVs to many US police departments. Persistent Surveillance's UAV uses ultrahigh resolution cameras that cover over thirty-two-mile square miles in order to be able to track the movements of individual cars and people, saving a history so that movements can be tracked backward in time.

Of course, often it isn't necessary to surreptitiously collect information. Sometimes you can just ask politely for it. In the US, subpoenas and warrants are used to request information from corporate providers. While warrants require probable cause (in the legal sense), subpoenas do not. As it publishes in its transparency report, Google receives around forty thousand data requests every year, about a third of which are by subpoena. Google returns data for roughly 80 percent of requests, and each request impacts, on average, roughly two user accounts (i.e., each individual request is highly targeted). Of note is that the contents of emails are available by subpoena. While subpoenas and warrants are basic in their nature, they usually can only be accessed by governmental adversaries.

Attacking Devices

The above tactics attempt to collect data while it is in transit or when it is held in the cloud. A final place to collect your data is right from your own device (phone or computer). This might happen if your device is confiscated by the police during a detention or search. We will discuss this more in the chapter "Protecting Your Devices," but we highlight some tactics for extracting device-held data here.

Cellebrite is an Israeli company that specializes in selling tools for extracting data from phones and other devices, such as their *Universal Forensic Extraction Device* (**UFED**), which is small enough to carry in a briefcase and can extract data quickly from almost any phone. However, this requires physical control of your device. NSO Group (another Israeli company) sells the ability to remotely install spyware called **Pegasus** on some iPhones and Androids that will extract text messages, call metadata, and passwords, among other data. The NSA has a family of malware (malicious software) denoted **QUANTUM** that can either gather data or block data from reaching the target device. But the NSA is able to install this malicious software on a mass scale with the use of their **TURBINE** system, which is able to disguise NSA servers as, for example, Facebook servers and use this as a means to injecting malware onto the target's device.

While Pegasus and QUANTUM can be deployed widely, it can be politically dangerous to do so, as these programs are generally met with public outcry. The more widely an invasive surveillance technology is deployed, the more likely it is to be discovered, as was the case with Pegasus.

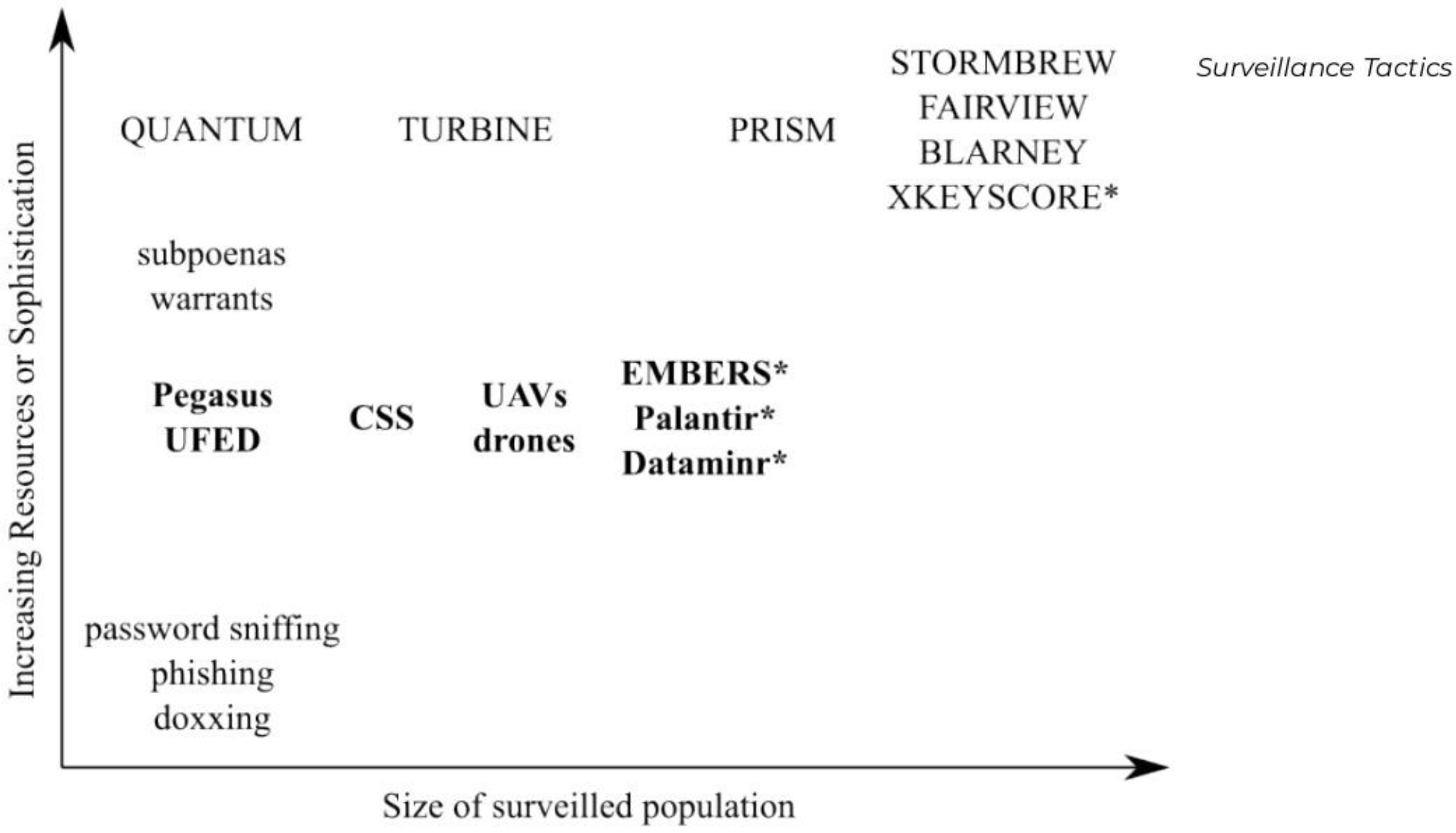
Personalized Harassment

While outside the realm of typical surveillance, personalized harassment should be in mind when considering digital security risks. Doxxing, phishing, and password sniffing are techniques available to the lower-resourced adversary but shouldn't be ignored for that reason. You may wish to revisit the story of Black Lives Matter activist DeRay Mckesson from the chapter "Passwords," who had his Twitter account compromised despite employing two-factor authentication. All his adversary needed was access to some personal information, which may have been discoverable from public sources or through personal knowledge.

Doxxing is the process of publishing (e.g., on a discussion site) a target's personal information that might lead to the harm or embarrassment of the targeted individual. While this is very easy to do, it is also very difficult to protect yourself: once information about you is available online, it is challenging or impossible to remove it.

Phishing describes methods of obtaining personal information, such as a password, through spoofed emails and websites. While phishing can be deployed on a mass scale, the most successful type of phishing (spear phishing) targets individuals by using already known information to improve success rates.

Password sniffing can be as low tech as looking over your shoulder to see you type in a password or can involve installing a keystroke logger to record you typing in your password, but this requires the ability to install a keystroke logger on your device, for which there are methods of varying degrees of sophistication. Traditional password sniffing captures a password as it passes through the network, which can be possible if the traffic is not encrypted, and again requires varying degrees of sophistication but certainly could be deployed by a skilled individual.



* indicates a program that analyze surveillance data, rather than create surveillance data.
Bold items are sold by companies and essentially available to anyone with the budget.

In Context: Standing Rock

In 2016, opponents of the construction of the Dakota Access Pipeline (DAPL) set up a protest encampment at the confluence of the Missouri and Cannonball Rivers, under which the proposed oil pipeline was set to be built. The pipeline threatened the quality of the drinking water in the area, which included many Native American communities, including the Standing Rock Indian Reservation. Eventually the protest encampment would grow to thousands of people and was in place for ten months.

Energy Transfer Partners, the company building DAPL, employs a private security force, which, a few months into the protest encampment, unleashed attack dogs on the protesters. In addition, Energy Transfer Partners very quickly hired TigerSwan to aid in their suppression of the protest movement. TigerSwan is a private mercenary company that got its start in Afghanistan as a US government contractor during the war on terror. As such, TigerSwan employs military-style counterterrorism tactics and referred to the Native American protesters and others who supported them as insurgents, comparing them (explicitly) to the jihadist fighters against which TigerSwan got its start. TigerSwan's surveillance included social media monitoring, aerial video recording, radio eavesdropping, and the use of infiltrators and informants.

Eventually local, regional, and federal law enforcement would be called in, with TigerSwan providing situation reports to state and local law enforcement and in regular communication with the FBI, the US Department of Homeland Security, the US Justice Department, the US Marshals Service, and the Bureau of Indian Affairs. While many (but not all) of the tactics employed by TigerSwan would be illegal for government law enforcement to adopt, the State is able to skirt this by receiving updates from private companies. This is common practice in many areas in law enforcement, with police departments buying privately held data that would violate the Fourth Amendment if the data was collected directly by the State.

The public-private partnership between state law enforcement agencies, Energy Transfer Partners, and TigerSwan was instrumental in bringing an end to the protest encampment, with the State eventually violently removing protesters through the use of tear gas, concussion grenades, and water cannons (in below-freezing weather), resulting in approximately three hundred injured protestors (including one woman who nearly lost an arm).

While the encampment ended and the pipeline eventually was built, continued opposition eventually led to a court ruling that the pipeline must be shut down and emptied of oil in order to complete a new environmental impact review.

It is important to remember that even though mass surveillance collects information about almost everyone, the harm it causes is differential. Certain groups are surveilled more heavily, or surveillance information about certain groups is used disproportionately. Examples of groups in the US that are disproportionately harmed by State and corporate surveillance are Muslim Americans, Black and African Americans, Native Americans, and social movement participants, as we discussed in the chapter "Mechanisms of Social Movement Suppression."

What to Learn Next

We encourage you, after this rather dismal account of all the ways your data can be swept up, to immediately start reading the chapter "Defending against Surveillance and Suppression."

External Resources

- Canadian Journalists for Free Expression. "Snowden Surveillance Archive." Accessed February 9, 2021.
- Electronic Frontier Foundation. "Atlas of Surveillance." Accessed February 9, 2021.
- Department of Homeland Security. "Fusion Centers." Accessed February 9, 2021.
- Google. "Transparency Report." Accessed February 9, 2021.
- Nakashima, Ellen, and Joby Warrick. "For NSA Chief, Terrorist Threat Drives Passion to 'Collect It All.'" *Washington Post*, July 14, 2013.
- Greenwald, Glenn. *No Place to Hide: Edward Snowden, the NSA and the Surveillance State*. London: Hamish Hamilton, 2015.
- N, Yomna. "Gotta Catch 'Em All: Understanding How IMSI-Catchers Exploit Cell Networks." Electronic Frontier Foundation, June 28, 2019.
- Stanley, Jay. "ACLU Lawsuit over Baltimore Spy Planes Sets Up Historic Surveillance Battle." American Civil Liberties Union, April 9, 2020.
- Biddle, Sam. "Police Surveilled George Floyd Protests with Help from Twitter-Affiliated Startup Dataminr." Intercept, July 9, 2020.
- Ahmed, Maha. "Aided by Palantir, the LAPD Uses Predictive Policing to Monitor Specific People and Neighborhoods." Intercept, May 11, 2018.
- Muthiah, Sathappan, Anil Vullikanti, Achla Marathe, Kristen Summers, Graham Katz, Andy Doyle, Jaime Arredondo, et al. "EMBERS at 4 Years: Experiences Operating an Open Source Indicators Forecasting System." In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining—KDD '16*, 205–14. San Francisco: ACM, 2016.
- Boot, Max. "Opinion: An Israeli Tech Firm Is Selling Spy Software to Dictators, Betraying the Country's Ideals." *Washington Post*, December 5, 2018.
- Intercept. "Oil and Water." 2016–17.
- Fortin, Jacey, and Lisa Friedman. "Dakota Access Pipeline to Shut Down Pending Review, Federal Judge Rules." *New York Times*, July 6, 2020.

Media Attributions

- atlas-of-surveillance © Electronic Frontier Foundation is licensed under a CC BY (Attribution) license
- surveillance-strategies © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- prism_slide_5 © JSvEIHmpPE is licensed under a Public Domain license
- surveillance-tactics © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license

PART 3: DEFENDING SOCIAL MOVEMENTS (IN THE US)

Defending against Surveillance and Suppression

We recommend that you read the chapters “Mechanisms of Social Movement Suppression” and “Digital Threats to Social Movements” before reading this chapter.

What You'll Learn

1. What threat modeling is
2. Strategies for reducing threats to your digital security

You may hear that there is no such thing as perfect digital security, and we agree. The surveillance capabilities of a well-resourced adversary are nearly limitless, and those that we described in “Digital Threats to Social Movements” barely scratch the surface. However, not all risks are equal, not all surveillance tools are equally likely to be used, and there is a lot that an individual and a group can do to reduce the threats due to surveillance.

We can model a digital security threat in terms of the following relationship:

$$\text{threat} \propto \frac{(\text{surveillance capabilities}) \times (\text{suppression risk})}{\text{effort required to obtain data}}$$

In this model, **surveillance capabilities** refers to your opponent’s level of resources, as discussed in the chapter “Digital Threats to Social Movements.” **Suppression risk** refers to the ways in which your opponent may try to undermine you, as discussed in the chapter “Mechanisms of Social Movement Suppression.”

It is important to keep in mind that surveillance supports suppression both indirectly and directly. Many of the examples we gave in the chapter “Mechanisms of Social Movement Suppression” were indeed supported by surveillance:

- The **direct violence** meted out on Black Panther Party leader Fred Hampton through a

targeted assassination was supported by detailed knowledge of his schedule and apartment layout.
- The US Department of Justice issued threats of sanction through the **legal system** against those individuals organizing the protests of Donald Trump's inauguration and requested to obtain all website traffic information of an organizing web page (described at the end of the chapter "Anonymous Routing").
- Steven Salaita's **employment deprivation** was a result of the monitoring of his Twitter activity.
- The **deception** used by the FBI against Mohamed Mohamud began with the monitoring of Mohamud's email.

Reducing the Threat

We can reduce digital security threats by *decreasing* surveillance capabilities or suppression risk or by *increasing* the effort required to obtain one's data.

Reducing Surveillance Capabilities

Most activists have little immediate control over surveillance capabilities. However, there are a number of laudable efforts to regulate surveillance with some success, such as the banning of face recognition and CSS in certain jurisdictions. But unless your social movement work is aimed at trying to ban or limit surveillance, going down this route would take you away from your goals.

Reducing Suppression Risk

Likewise, activists have little control over suppression risk. You could minimize the risk of suppression by reducing the threat to your opponent, but then you would be succumbing to the chilling effect.

Increasing the Effort Required to Obtain Your Data

That leaves us with increasing the effort required to obtain your data, which is the focus of the remainder of this book. While protecting all data is important (the more your opponent knows about you, the better they can undermine you), we encourage putting any additional effort in protecting your data toward the most protective strategies. So to *guide* that effort, you should keep in mind the surveillance capabilities of your opponents and their likely modes of suppressing your efforts. To this end, focus on protecting data that

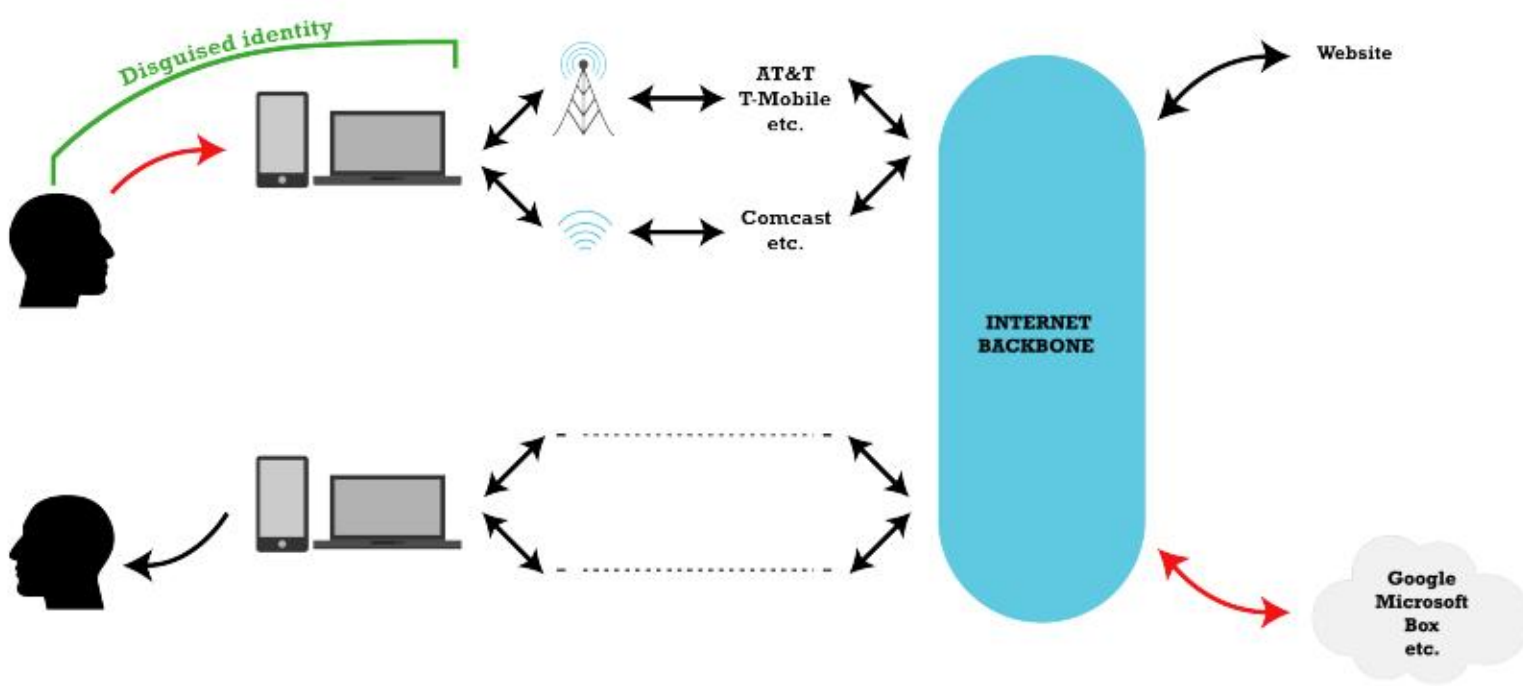
1. could most likely be used to suppress your efforts and

2. is most vulnerable to surveillance.

Understanding point 1 will be through a deep understanding of the efforts and opponents of a given social movement. To consider point 2, we need to understand where your data is (described below) and how to protect it (which will be discussed in the remaining chapters of this book).

Where Is Your Data?

We take different protective strategies depending on where data is vulnerable. Your information becomes data when it is put on a device (e.g., a cell phone or laptop) and then may be transmitted through the internet via service providers. We distinguish here between websites where you may be browsing or cloud providers where your data may be held (from Google to Facebook).



Where your data is

In the remaining chapters, we discuss how to protect where your data is. In the chapter "Security Culture," we discuss how to decide whether your information becomes data (when you have control over it) and whether to store your data in the cloud—that is, whether you want your data to transmit over the red arrows. In the chapter "Protecting Your Devices," we discuss how to protect data that is held on devices that you have control over (e.g., your laptop and cell phone). In the chapter "Protecting Your Communications," we discuss how to protect your data while it transmits from you to your destination, be that a website, cloud provider, or another person. In the chapter "Protecting Your Remote Data," we discuss how to protect data that is held in the cloud if you have made the decision to do so.

We then discuss how to protect your identity—that is, how to be anonymous or pseudonymous online and break through censorship—in the chapter "Protecting Your Identity." Finally, we discuss how to select digital security tools in the conclusion and give the principles we use for our recommendations.

In Context: Edward Snowden

In the years leading up to 2013, Edward Snowden collected data from his workplaces (mostly NSA subcontractors) that he had access to in his role as a systems administrator. Snowden's leaks of troves of classified material illustrated just how advanced and broadly deployed the surveillance tactics of many of the world's most powerful governments were. However, in order to make these disclosures, Snowden was up against a powerful adversary: the National Security Agency itself.

Snowden was unlikely to achieve long-term anonymity—his goal was to keep his behaviors (collecting information) and goal (whistleblowing) unknown for long enough to leak the information to journalists, who would responsibly report on it, and hopefully long enough to get to a safe haven, where he could live in freedom. It took months for Snowden to set up an encrypted communications channel with Glenn Greenwald (a journalist known for fearless, deep reporting), this being in the days before "plug-and-play" end-to-end encrypted messaging apps. But once the reporting on Snowden's disclosures started, he knew his identity would be discovered and unmasked himself. Snowden didn't end up where he had hoped (Latin America). His US passport was canceled during his flight from Hong Kong (where he disclosed his leads to Glenn Greenwald) to Russia, preventing him from further air travel. Snowden was able to claim asylum in Russia.

However, Snowden was very successful in his whistleblowing, with the reporting lasting for years after and with numerous changes to our communications: encryption is more commonly available now, so much so that many people don't even know when their conversations are end-to-end encrypted.

What to Learn Next

- Security Culture

External Resources

- AnarchoTechNYC. "Persona Based Training Matrix." June 9, 2020.
- Electronic Frontier Foundation. "Your Security Plan." Surveillance Self-Defense, August 1, 2014.
- Snowden, Edward J. Permanent Record, 2019. Metropolitan Books.

Media Attributions

- where-your-data-is © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license

Security Culture

We recommend that you read the chapter "Defending against Surveillance and Suppression" before reading this chapter.

What You'll Learn

1. What social movement security culture is
2. Why security culture is essential to digital security

Social movements aware of the history of informant-driven suppression by State and private adversaries have developed what is termed *security culture*. This term refers to information-sharing agreements and other group practices intended to minimize the negative impacts of infiltration, surveillance, and other suppressive threats to the group, its work, its membership, and broader social movements; that is, *security* here means something much broader than digital security. The term *culture* indicates an aspiration for security principles and practices to become reflexive and intuitive. The ideal security culture helps a group to safely and easily communicate and bring in new members (if desired) while avoiding excessive paranoia or cumbersome procedures and policies.

Although perspectives and practices on security culture vary widely, some important widespread principles you should adhere to are the following:

1. Share information on a need-to-know basis.
2. If you are organizing with others, get to know your group members as well as possible.
3. Avoid gossip and rumors.

Security Culture Meets Digital Security

Let's explore some of these elements in detail and how they relate to digital security.

Need to Know: Minimize Information Sharing and Digitizing

The first principle of keeping secrets is to minimize the number of people who must be trusted to keep them. Of course there is a spectrum of information sensitivity, from public announcements to open meetings, from in-development press releases to specific places and times of direct actions. Deciding what information needs to be protected and being careful to protect it is only part of the picture; people also need to accept that they won't have access to sensitive information unless they need it to do their work.

From a digital security perspective, this also means deciding what information becomes digitized. (Do you really need a Google Doc listing all the people who plan on attending a protest? Do you really need to post identifiable photos of people who showed up? Do you need those posts to be public and geolocated?) Limiting the amount and extent of information sharing dovetails with good digital security practices because no platform or means of communication can be considered perfectly secure.

Before taking specific digital security measures (such as using possibly complicated end-to-end encrypted technology), consider what information needs to be stored, be shared, or even exist in a digital format—perhaps (absent a global pandemic) we should be meeting and discussing our ideas in person as much as possible. Keep in mind that any digital information is extremely easy to copy, and so even a strong encryption can only protect information to the extent that every human with access to it can be trusted. Not even a perfectly designed secure app or digital platform can stop information from being compromised by an infiltrator or defector within a group.

Get to Know: Vetting and Trust Building

Get to know the people that you work with so that you can trust them with whatever risks you decide to take together. But when you decide to digitize information, you are potentially welcoming more "people" (well, corporations and the State) into your organizing circle. If your group uses, for example, Gmail for communications among group members, then Google also has all those emails, and those emails can be easily subpoenaed by the State. So you should be ready to trust that any entity has access to your *unencrypted* data, whether that entity is a human with whom you interact, your internet service provider, your cloud storage provider, or your email provider.

Don't Gossip or Spread Rumors

Social movements in the past have been crushed by gossip and rumors, with the State using our human weaknesses to engage in gossip and rumors to its advantage, as we discussed in the chapter "Mechanisms of Social Movement Suppression": the use of snitch jacketing, agent provocateurs, and

false propaganda as tactics of deception depend on social movement participants believing the source and repeating information.

For digital security, we can aim to authenticate the source of information. This is particularly important online, where one can more easily pretend to be someone one isn't, either through low-tech means (such as fake accounts or stealing an account) or high-tech means (such as redirecting network traffic). We will discuss authenticating digital sources in the chapter "Protecting Your Communications" and the conclusion, "Selecting Digital Security Tools."

But a very basic consideration is one's use of social media, where gossip and rumors abound and where the details of our personal lives make infiltration unnecessary to get to know what your weaknesses might be. Social media platforms should only be used to publicly distribute information, and conversations there should never be considered private.

These protective actions have the potential to protect you from social media monitoring, subpoenas and search warrants, and doxxing.

In Context: Saint Paul Principles

Leading up to the 2008 Republican National Convention in Saint Paul, Minnesota, different social movements came together around the opposition to the Republican Party's support of the war in Iraq. The coalition of protest groups adopted the following principles ahead of the convention in order to make space for different groups' views and strategies and reduce the risks one group is facing from affecting another group:

1. Our solidarity will be based on respect for a diversity of tactics and the plans of other groups.
2. The actions and tactics used will be organized to maintain a separation of time or space.
3. Any debates or criticisms will stay internal to the movement, avoiding any public or media denunciations of fellow activists and events.
4. We oppose any state repression of dissent, including surveillance, infiltration, disruption and violence. We agree not to assist law enforcement actions against activists and others.

These rules have become known as the "Saint Paul Principles" and have been adopted by many coalitions of groups in the years since. The principles elevate notions of security culture from an intragroup level to an intergroup level. They are designed to help different groups come together if they have the same ultimate aim but may disagree on how to get there and to increase the success that the overarching movement will be successful in their agreed-upon aim.

What to Learn Next

- Protecting Your Devices
- Protecting Your Communications
- Protecting Your Identity

External Resources

- Wikipedia. "2008 Republican National Convention." January 11, 2021.
- Activistsecurity.org. *A Practical Security Handbook for Activists and Campaigns*. Civil Liberties Defense Center, May 2007.
- Sprout Anarchist Collective. "What Is Security Culture?" 2012.

Protecting Your Devices

We recommend that you read the chapters “Passwords” and “Digital Threats to Social Movements” before reading this chapter.

What You’ll Learn

1. Common ways in which phones and computers are compromised
2. Strategies for protecting phones and computers

The amount of data you keep on your phone and laptop is staggering. Contacts, emails, photos, documents, calendars, tax returns, banking details, and, in the case of a smartphone, often a detailed history of your location as long as you’ve had that phone. A lot of this data you will also share with your cloud storage providers (i.e., Apple, Google, Dropbox), but that is the focus of the chapter “Protecting Your Remote Data.” Here we focus on protecting the data that you keep with you on your laptops and cell phones from a remote or physical attack.

Physical Attacks

By a physical attack, we mean that your adversary would first gain physical access to your device through loss, theft, or confiscation. You may lose your phone at an inopportune moment that places it in the hands of an adversary rather than a Good Samaritan, or your adversary may steal your phone. More likely your phone may be confiscated while crossing a border or during an arrest, either planned or unplanned.

Those who were swept up in the mass arrests during the protests of the presidential inauguration on January 20, 2017 (J20), had their phones confiscated and subject to search by a tool from the Israeli company Cellebrite, which extracts all information on a device (phone or computer) and all remote accounts that device has access to (e.g., Google, Facebook, Dropbox). In the article “How to Protect Yourself from the Snitch in Your Pocket,” one J20 defendant described the eight thousand pages of data that a Cellebrite tool extracted from his confiscated cell phone; he received this information from his lawyer as they prepared for his defense:

- A list of all my contacts, including phone numbers and emails that contacted me that were not stored in my phone, with a count of how many times I called, messaged, or emailed them or was called, messaged, or emailed by them.
- The number of emails I received, sent, and drafted to specific email addresses and how many shared calendar events I had with those email addresses. The number of incoming/outgoing/missed calls from each number and if they were my contacts, and how long total calls were between me and a number. Whether they were in my contacts, and if so what nickname I call them in my phone.
- The number of SMS texts received/sent/drafted to a number. The content of all texts, even if they were deleted, including drafts.
- Whatsapp contacts, their "usernames" (i.e. the phone number attached to their account), and how many chats/calls took place between me and them.
- All apps, when they were installed/deleted/last used/purchased, and what permissions they had.
- Audio files that were stored in Google Drive, as well as any podcasts, voice memos, and ringtones. Timestamps for their creation/deletion/modification/last access.
- All calendar events, attendees invited, location tags, etc.
- Traditional call log info you might expect.
- Date and time of all cell towers my phone had ever connected to and their location, conveniently linked to Google Maps. A world map marking all cell towers accessed by my phone.
- Chats from Signal, WhatsApp, SMS, Google Hangouts, TextSecure, GroupMe, and Google Docs; a list of all participants in those chats; text body content; whether it was read or unread, with a timestamp for sent and read; if it was starred; if it was deleted; all attachments. These chats were also from years ago, before I even had a smartphone.
- All information for my contacts, including whether the contact was deleted or not.
- Web browser cookies.
- Any document ever opened on my phone, including text documents, attachments, Google docs, and those created by apps.
- Emails and email drafts, including all sending information, entire text content, and up to 16 attachments.
- Images/photos/videos along with their created/accessed timestamp and any metadata.
- Ninety-six random tweets from one of my Twitter accounts, some from as far back as 2013.
- A list of all wifi networks that my phone ever connected to, their passwords, hardware identifiers, and when I connected to them.
- The last five times my phone was turned on, including twice two months after I lost access to it.
- Web history and web and Playstore search history.
- A list of every word ever typed into my phone and how many times that word was typed, including email addresses as words, and words I added to the dictionary so they wouldn't continue to be autocorrected to something else.
- What they call my "timeline": every action (texts, calls, emails, web history, app usage

including maps searches, connections to wifi networks or new cell towers, etc.) with timestamp to be easily sorted.

What Can I Do?

A detective testifying at a J20 trial noted that among the phones that had encryption enabled, he was only able to access basic device information and not the contents of the phone storage. iPhones and Android devices that are running up-to-date operating systems have encryption enabled by default, while Apple and Microsoft computers need to have this enabled. Encrypting your device is not, however, a panacea. The encryption that protects your device is only as strong as the password protecting it.

Device-encryption passwords regrettably suffer from a convenience-security trade-off. A passphrase (as described in the chapter “Passwords”) may need to be composed of six or more words to sustain a physical attack, but such a passphrase is cumbersome to type in frequently. There are a few options, all with trade-offs. For phones or laptops, you can modify your settings to change how often you need to enter your password, passphrase, or unlock code. (Note that encryption is only in effect when a screen lock is enabled.) Or you could modify the strength (length) of your password, passphrase, or unlock code depending on your situation. However, these strategies rely on knowing when your situation requires higher levels of security and consistently strengthening your security when needed.

For phones and some laptops, one can often choose between a typed passphrase or biometric input (such as a fingerprint). A fingerprint is more convenient than a typed password. For the purposes of encryption, your fingerprint will be paired with a passphrase (which should be as strong as you can manage). However, if your device is confiscated by law enforcement, your fingerprint may be compelled from you. So when the risk of device confiscation is high, one should still consider removing the ability to biometrically unlock your device.

There are additional protections against physical interference that one might consider. A privacy screen can obscure an eavesdropper from seeing the passwords (and anything else) you type. Faraday bags can prevent your phone from transmitting or receiving information; among other things, this can prevent your phone from recording location information. The ability to remotely wipe your phone is provided by major cell phone manufacturers, and while it may relinquish control over your device to the same corporations that may share your information with your adversaries (as we discuss in the chapter “Protecting Your Remote Data”), it may be a useful tool in certain situations.

Remote Attacks

By a remote attack, we mean that an adversary would access the data on your phone or laptop through an internet or data connection. There are companies that design and sell the ability to infect your device (usually focusing on smartphones) with malware that would allow their customer (your adversary, be it a corporate or state agent) to gain remote access to some or all your information.

For example, Citizen Lab uncovered wide and varied use of the spyware Pegasus created and sold by another Israeli company, NSO Group. Coupled with some social engineering to convince the target to click on a link, the spyware grants the ability to turn on and record from the phone's camera and microphone, record calls and text messages (even those providing end-to-end encryption), log GPS locations, and send all this information back to the target's adversary. Citizen Lab reported that Pegasus attempts were made against Ahmed Mansoor, a human rights defender based in the United Arab Emirates, and twenty-two individuals in Mexico ranging from politicians campaigning against government corruption to scientists advocating for a state tax on sugary drinks.

What Can I Do?

Remote attacks, such as those sold by NSO Group, rely on flaws in computer software known as *zero-days*. Such flaws are unknown to the software provider (e.g., Apple or Microsoft). Until they are known (which happens on "day zero"), there is no chance that the software provider could have fixed or patched the vulnerability, and so there is no chance that a victim could protect themself. Computer security is often a cat-and-mouse game. The products of malware and spyware creators (such as NSO Group) are only good so long as the targets (or more accurately, companies like Apple, Google, and Microsoft) don't know about the malware being deployed. As soon as they do, they fix their products so that the malware is no longer effective.

But these product fixes only work if the target (you) updates their device. So the lesson here is to *install all security updates as soon as they are available*. Unfortunately, smartphones do not receive security updates indefinitely, with particular devices (e.g., Nokia 5.3) only being supported by an operating system (e.g., Android) for a few years. You can check if an Apple or Android phone is receiving security updates through the settings.

Many malware products require phishing for installation on the target's device: convincing the target to click on a link or opening a file (in either an email or a text message). So the second thing you can do is *be wary of what you click on*. Do you know the sender? Are you expecting something from the sender? Does anything seem, well, fishy? In fact, it was vigilance that led Ahmed Mansoor to avoid spyware infection: he sent the phishing text along to Citizen Lab, which led to their reporting on the abuse of spyware from NSO Group.

Finally, *be wary of the apps you install and what permissions you grant them*. Does a flashlight app need access to your contacts and camera? Do you really need to install that game created by an

unknown software creator? Every app you install is a potential vector for malware, so it is a good opportunity to practice minimalism.

In Context: Compromising Protesters' Phones

In September 2020, it came to light that the Department of Homeland Security was "extracting information from protestors' phones" during the extended protests during the summer of 2020 in Portland, Oregon. Purportedly using a novel cell phone cloning method, the government was able to intercept communications to the phones of protestors. While this is disturbing and likely illegal, the details of the attack remained classified. However, we can try to infer likely methods of attack and likely protective practices.

If the cloning method requires a physical attack, most likely the compromised phones are those that were confiscated in prior arrests during the summer. However, this limits the surveillance potential to only the phones of arrestees and allows them either to no longer trust their phones or to factory reset their phones to remove possible malware.

On the other hand, if the cloning method can be done remotely, this greatly expands the number of phones that might be compromised and has no signaling event.

In either case, the use of in-transit and end-to-end encryptions would still protect phone communications (but perhaps not metadata), as you can learn about in the chapter "Protecting Your Communications."

What to Learn Next

- Protecting Your Communications
- Protecting Your Identity

External Resources

- *Earth First! The Journal of Ecological Resistance*. "How to Protect Yourself from the Snitch in Your Pocket." Winter 2017–18. (A digital version of the article is available at protestarchive.org)
- Marczak, Bill, and John Scott-Railton. "The Million Dollar Dissident: NSO Group's iPhone Zero-Days

Used against a UAE Human Rights Defender." Citizen Lab, August 24, 2016.
- Schiano, Chris. "Criminalizing Dissent: Contested Evidence Introduced in J20 Trial Testimony." Unicorn Riot, November 30, 2017.
- Klippenstein, Ken. "Federal Agencies Tapped Protesters' Phones in Portland." Nation, September 21, 2020.
- Vice Media Group. "Phone Crackers." Accessed February 9, 2021.

Protecting Your Communications

We recommend that you read the chapters "The Man in the Middle," "Passwords," and "Digital Threats to Social Movements" before reading this chapter.

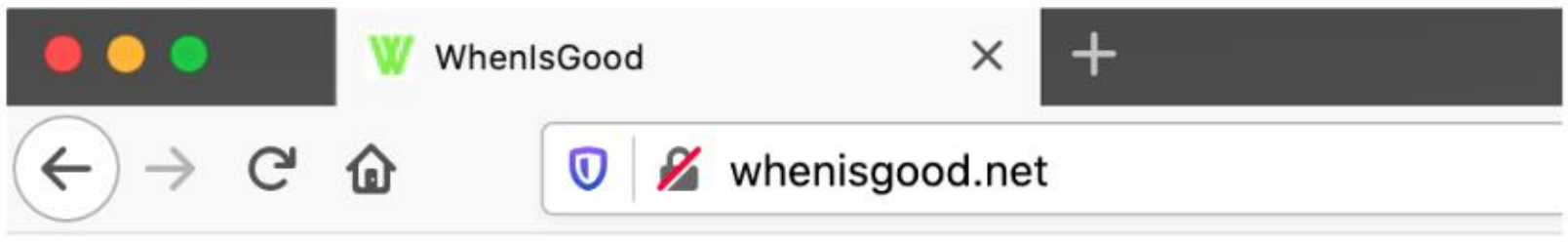
What You'll Learn

1. The difference between in-transit and end-to-end encryption
2. Who has access to your information when not using an encryption
3. Who has access to your information when using an in-transit encryption
4. Who has access to your information when using an end-to-end encryption

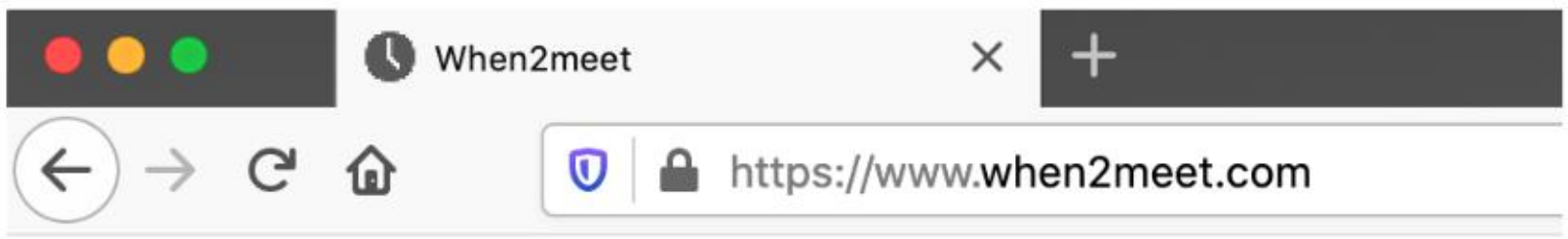
The best way to protect your online communications is through encryption. But not all encryption is equally protective. We will focus on the concepts that distinguish between the degrees of protection.

Encrypted or Not

The most basic version of encrypted communications is in-transit encryption, where your information is encrypted between your computer and a server. In the context of browsing the web, this is the best you can do to protect the content (but not the metadata) of your communications from an adversary. Most web browsers indicate whether your browsing is encrypted by the URL, as illustrated below.



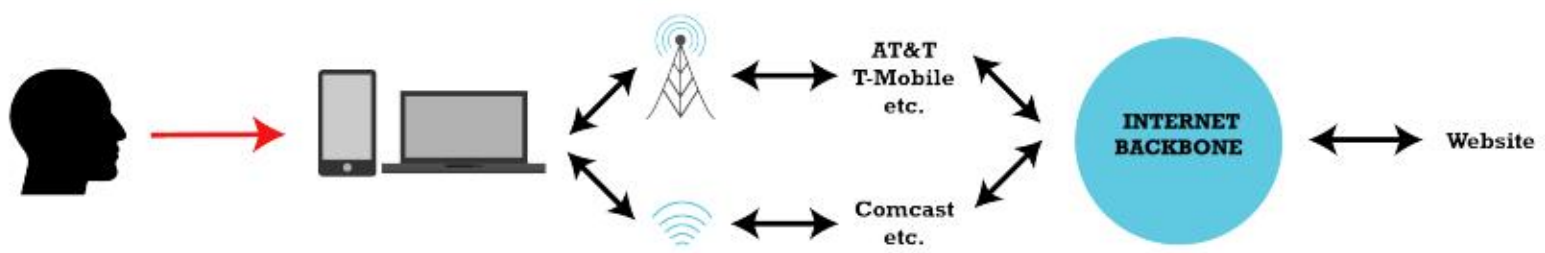
HTTP, not encrypted



HTTPS, encrypted

In the top example, the information is transmitted unencrypted. The full URL in this case is `http://whenisgood.net`, where `http` indicates accessing a web page without encryption. This browser (Firefox) emphasizes this point with a struck-through lock. In the bottom example, the information is encrypted: `https` indicates accessing a web page with encryption, and the `s` stands for "secure." The keys used for this encryption are exchanged between your computer and the whenisgood servers using the Diffie-Hellman key exchange, as described in the chapter "Exchanging Keys for Encryption."

Using `http`, every entity on the path between you and the website pictured below can access the content of your web browsing (such as the pictures being loaded and any information you might type into a web form). Further, anyone snooping on the communications between the entities on these paths (such as between Comcast's network and the internet backbone) may also have access to your browsing content. We qualify this with *may* because the communications between two entities on this path may be encrypted. For example, the communications between a cell phone and a cell tower are encrypted in most cases.



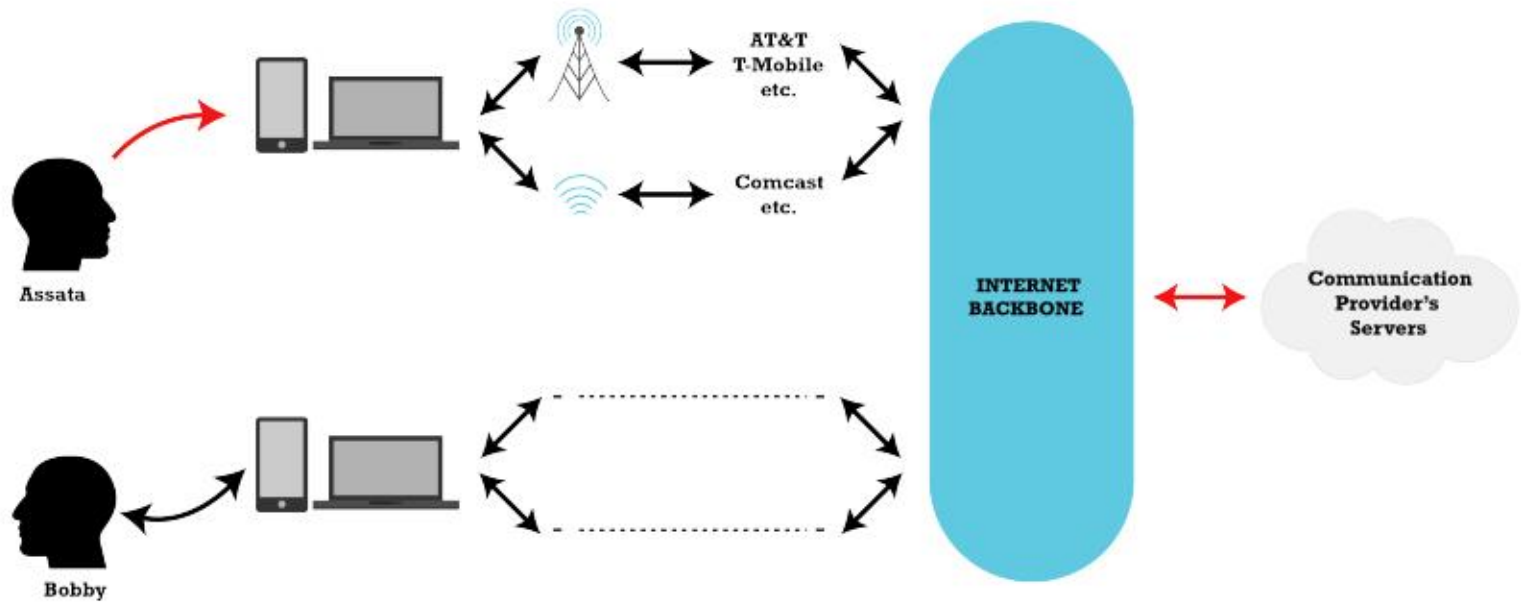
Who has access to your browsing data

By contrast, when you use `https`, *only* you and the website (technically, the servers that are hosting the website) have access to the *content* of your browsing. We specify *content* here because certain metadata would still be known by entities on the path between you and the website, such as the basic URL of the website, the amount of time you spend browsing the website, and the amount of information you are downloading from the website.

In-Transit Encryption

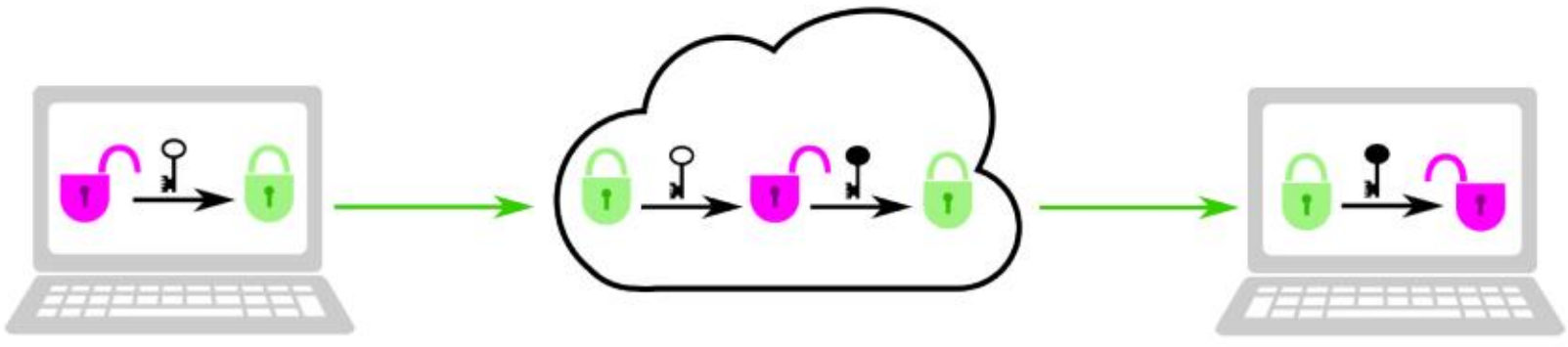
When we communicate with another person by email, instant messaging, or video chat, those

communications are (most often) routed through the communication provider (e.g., Google servers for email or Microsoft servers for Skype calls), as pictured below. Nowadays, those communications are usually encrypted but most often only encrypted between you and the communication provider. That is, while the entities and eavesdroppers along the path between Assata and the communication provider's servers (center) and between the communication provider's servers and Bobby do not have access to the content of your communications (but can glean metadata), the communication provider *does* have access to the content of your communications.



Who has access to your communication data

We call this in-transit encryption because the content is encrypted while it is in transit between Assata and the communication provider and between the communication provider and Bobby. The in-transit encryption keys are generated separately for each part of the path between Assata and Bobby, pictured below. The provider (center) performs a Diffie-Hellman key exchange with Assata, generating a shared key, and performs a separate Diffie-Hellman key exchange with Bobby, generating a *different* shared key. When Assata sends a message to Bobby through the provider, the message is first encrypted with the key Assata shares with the provider, and then the message is transmitted to the provider. The provider decrypts the message with the key that Assata and the provider share. Then the provider re-encrypts the message with the key that the provider shares with Bobby before transmitting the encrypted message to Bobby. Bobby can then decrypt the message. Therefore, the message only exists in a decrypted state on Assata's and Bobby's devices and the provider's servers; the message is encrypted when it is in transit between these entities.



In-transit encryption

End-to-End Encryption

While in-transit encryption protects your communications from many potential adversaries (such as your internet service provider, the Wi-Fi hotspot, or a snoop along the communication channels), the communication provider still has access to all that information. Even if the provider is not a direct adversary, they may share that information with an adversary (such as through a subpoena or warrant). End-to-end encryption (E2EE) will protect your communications from even the communication provider.

For E2EE (pictured below), Assata and Bobby exchange keys (using a Diffie-Hellman key exchange or a similar procedure). While their communications are routed *through* the communication provider, so long as the provider isn't mounting a man-in-the-middle attack, the communications through the provider are encrypted with a key that only Bobby and Assata have access to. That is, the message only exists in a decrypted state on Assata's and Bobby's devices. Assata's and Bobby's devices are the *end*points of the communication (hence end-to-end encryption).



End-to-end encryption

Authentication

While E2EE is the gold standard, there are further considerations. As mentioned above, end-to-end encryption is only established if the communication provider (or another third-party member) does not mount a man-in-the-middle attack starting at the time of key exchange. However, as we covered in the chapter "The Man in the Middle," if Assata and Bobby verify their keys through independent channels, they can determine whether or not a man-in-the-middle attack has occurred and so whether their communications are truly end-to-end encrypted.

While many apps or services claiming E2EE provide the ability to verify keys, many do not, providing little guarantee to trust the claims of E2EE. Further, for those E2EE apps that do provide the ability to verify keys, most operate on a Trust on First Use (TOFU) basis. That is, communications may begin without verifying keys first. However, while actually doing key verification is the only way to guarantee E2EE, the existence of the *ability* to verify keys is protective against automated man-in-the-middle attacks, as even a small fraction of users verifying keys would catch widespread man-in-the-middle attacks.

And, of course, E2EE only protects the communication between devices—it does not protect the

data that is on the device. E2EE apps should be combined with strong passwords to protect the account or device.

In Context: Multiparty Video Chatting

There are many apps and services for video chatting between two or more people, with varying degrees of security. Here are three illustrative examples:

- Wire provides the gold standard of E2EE. Each user has an account that can be accessed from multiple devices (e.g., laptop and smartphone). There is a public key for each device that is used to establish an encryption key for a session (e.g., a video call), and the fingerprints of these keys can be compared to verify true E2EE. Wire allows E2EE video calls for groups of up to twelve users.
- Zoom allows video calls for much larger groups and does provide E2EE in that a video stream is encrypted and decrypted by the users with the same key. However, this key is established and distributed by the Zoom servers. Since Zoom has access to the encryption key, this cannot be considered true E2EE. Further, there is no mechanism for users to verify the encryption keys. As of summer 2020, Zoom had a proposal for establishing keys for true E2EE, but it has not yet implemented it.
- Jitsi Meet also provides large-group video conferencing, but only using in-transit encryption. However, Jitsi Meet is available to be hosted on any server (including your own, if you are so inclined). There is an instance of Jitsi Meet hosted by May First, a nonprofit that provides technical solutions to social movements and is a trusted third party to many groups. Even though May First has access to these communications, some would prefer to trust May First over a profit-driven solution such as Zoom.

What to Learn Next

- Protecting Your Devices
- Protecting Your Remote Data
- Protecting Your Identity

External Resources

- Blum, Josh, Simon Booth, Oded Gal, Maxwell Krohn, Julia Len, Karan Lyons, Antonio Marcedone, et al. “E2E Encryption for Zoom Meetings.” Zoom Video Communications, December 15, 2020.

Media Attributions

- http-example © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- https-example © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- who-has-access-to-your-data © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- data-access-communication © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- notE2EE © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license
- E2EE © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license

Protecting Your Remote Data

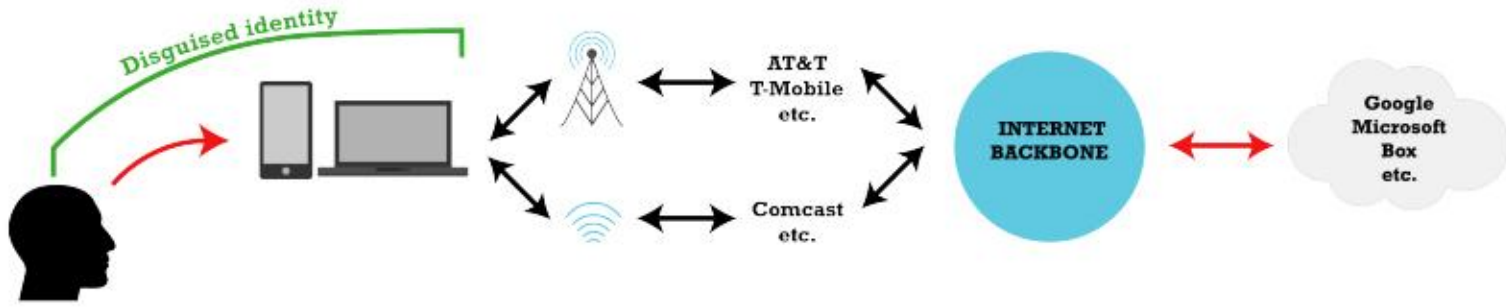
We recommend that you read the chapter "Protecting Your Communications" before reading this chapter.

What You'll Learn

1. Who has access to your data in the cloud
2. What of your data is in the cloud

The cloud is ubiquitous. Since the early 2000s, data is increasingly stored not exclusively (or at all) on your own device but on the servers of the companies that manage your device or operating system or whose services you subscribe to. If that data is not encrypted with a key that you control, that data is at risk for compromise.

Accessing your remote or cloud data or storage is similar to accessing a web page, as pictured below. In most models of accessing cloud storage, the information is protected by in-transit encryption, which would protect your data from potential adversaries along the path from your device to your cloud storage provider's servers (pictured below).



Who has access to your cloud data

However, as discussed in the chapter "Digital Threats to Social Movements," data that is stored remotely (and is not encrypted) is accessible by government adversaries by subpoena or warrant or may simply be shared with third parties. Unfortunately, even if we avoid the most explicit forms of remote data (such as what is offered by Dropbox or Google Drive), many of our devices encourage the remote backup of all our data (such as Apple devices to the iCloud), in some cases making it very difficult to avoid (as for Android devices to a Google account). This includes a potential wealth of

information, including your addresses, calendar, location history, browsing information—potentially anything you do with your computer.

In Context: Trusted or Encrypted Cloud Storage

There are many choices for cloud storage. In the following list, we describe a few options that illustrate the breadth of options, from not encrypted and not trusted, to not encrypted but trusted, to encrypted.

- Google will happily store all your information (email, files, contact information, device backups) for free. Of course, they extract value from this by using your data, but they can't do so if that data is encrypted with a key that only you control (and so it isn't). As we saw in the chapter "Digital Threats to Social Movements," Google returns data in response to roughly 80 percent of subpoena requests.
- The software ownCloud provides Box- or Dropbox-style cloud storage but, like Google's products, only uses in-transit encryption. (An enterprise version of ownCloud does provide some end-to-end encrypted file storage and sharing.) However, ownCloud, like the video-conferencing app Jitsi Meet, is available to be hosted on any server (including your own). Also, like Jitsi Meet, there is an instance of ownCloud hosted by May First, a service provider that is trusted by many. Even though May First has access to your stored data, some would prefer to trust May First over Google.
- CryptPad is a collaborative editing platform that offers an end-to-end encrypted alternative to Google Docs. Documents are accessed by a link that includes the key for decrypting the document, but that key appears after a # in URL—for example, `https://cryptpad.fr/pad/#/2/pad/edit/bpsky2zF5La8sZ_i-6r_cTj9fPL+`. The part of the URL after the # is known as a fragment identifier and is not transmitted to the server but is only used within the browser—in this case, to decrypt a given pad. Since the encryption key is part of the URL, one must take care in sharing such a link (i.e., only share this link over an encrypted channel, such as Signal).
- Keybase has a number of features including an end-to-end encrypted storage system akin to ownCloud or Dropbox. Unlike CryptPad, Keybase offers stand-alone apps (rather than operating in a browser) and handles the management of keys.

What to Learn Next

- Any remaining chapter in part 3

Media Attributions

- where-your-cloud-data-is © OSU OERU is licensed under a CC BY-NC (Attribution NonCommercial) license

Protecting Your Identity

We recommend that you read the chapters "Public-Key Cryptography" and "Anonymous Routing" before reading this chapter.

What You'll Learn

1. The difference between being anonymous and pseudonymous
2. Three distinct ways to use Tor
3. Some things you should never try to do using Tor

In the chapter "Anonymous Routing," we compared and contrasted virtual private networks (VPNs) and Tor as two methods for disguising one's metadata online. This can help one achieve anonymity or pseudonymity, but this is difficult to do over the long term. In this chapter, we will focus on skills for using Tor over VPN, but these lessons apply to using a VPN. One needs to additionally remember, though, that when using a VPN, the VPN provider knows who you are and the metadata of your internet communications (and the content, if it isn't encrypted). While we will focus on using Tor via the Tor Browser, know that there are other applications (such as secure-messaging applications or whole operating systems) that route internet requests through the Tor network.

Anonymity versus Pseudonymity

Before we describe different ways to use Tor, let us consider the difference between anonymity and pseudonymity. These terms are used in different ways in different contexts, and we restrict our use here to online communications and behavior.

Anonymity refers to being without a name or, more generally, without any identifier that could link to you. If you visit a website anonymously today and the same website anonymously tomorrow, the website should not even be able to tell that it was the same person both times. All the website should know is that "someone visited me anonymously yesterday" and "someone visited me anonymously today."

Pseudonymity refers to using a false name, with few or no people knowing your true identity. For example, Samuel Clemens published under the pseudonym Mark Twain, but of course his publisher

and others knew who the true author was. Edward Snowden used the pseudonym Cincinnatus in contacting journalist Glenn Greenwald. Greenwald did not know who was contacting him as Cincinnatus, and because Snowden was using Tor to contact Greenwald, neither did anyone else. However, Snowden's repeated use of the alias Cincinnatus allowed Greenwald to connect different communications he (and fellow journalist Laura Poitras) received from Snowden. We will refer to pseudonymity as allowing one to link different and otherwise anonymous sessions of communications under one persona.

Ways to Use Tor

Tor can be used to hide information about your identity (such as your physical location) and achieve anonymity and pseudonymity. For the novice user, Tor is accessed using the Tor Browser or Tor-compatible apps. For a more advanced user, non-web-browser communications can be routed through Tor by using an operating system (such as Tails or Whonix) that routes all your web traffic through Tor.

Hiding Your Physical Location

Using the Tor Browser as you do any other browser, including accessing emails or social media usernames that are linked to you, will conceal your physical location from those accounts. Each time you open a new Tor Browser tab or window and after a certain delay, Tor will route your web requests via a new location. However, many email and social media platforms will flag your account activity as suspicious if it is being accessed from different locations, as it will appear to be when accessing through Tor. So while it is possible to use Tor all the time and for everything, it may not be practical. If you are able to navigate these difficulties, you will still need to be smart to consistently hide your physical location by *avoiding* the following behaviors:

- Entering identifying information into a website (such as your address)
- Downloading a document that might access some part of the document via the web (like a photo) and opening it outside the Tor Browser (Word documents and pdfs can do this); if you need to access such a document, disconnect from the internet before opening

Achieving Anonymity

Using Tor can help you achieve anonymity. However, you will need to restrict your behavior to ensure you don't leak information that could break your anonymity. To that end, in order to maintain anonymity, you need to *avoid* the following behaviors during your *anonymous session* (in addition to those for hiding your physical location):

- Logging into accounts (e.g., social media, email, banking)
- Visiting your own website repeatedly

Achieving and Maintaining Pseudonymity

If you create a pseudonym that is unrelated to your true identity in order to, for example, post press releases or participate in forums, Tor can help ensure that your pseudonym stays unrelated to your true identity. However, to keep your real and pseudonymous identity separate, you need to *avoid* the following behaviors:

- Accessing different pseudonymous (or your pseudonymous and real) identities in the same session, as this can link these identities
- Accessing a pseudonymous account even once outside of Tor
- Using two-factor authentication with a phone (as your phone, even if it is a "burner," can reveal your physical location)
- Posting media with revealing metadata (such as location)

Note that the longer you attempt to maintain a pseudonymous identity, the more opportunity you give yourself to make a mistake. In addition to the mistakes above, your writing style can be used to identify you using stylometry. The more examples of your writing style that are available (under both your real and pseudonymous identity), the easier it would be to identify you.

Tor Warnings

There are some additional things to be aware of when accessing the internet via Tor.

As with any protective technology, nothing is perfect. If an adversary (Edgar) is able to watch Assata's connection to the Tor network as well her connection leaving the Tor network (to Bobby's website), Edgar will be able to determine that Assata is visiting Bobby's website. This is called an *end-to-end timing attack* or correlation attack.

If you attempt to use applications not designed for Tor over the Tor network, they may leak identifying information, such as screen resolution or a unique set of settings you may have.

Finally, when using Tor, keep in mind it only provides anonymity—for privacy, you need to be accessing web pages using end-to-end encryption via https (though unfortunately not all websites support this).

In Context: Getting the Real Tor Browser

The tools you use to protect yourself online are only useful if they are the real thing. In 2019, it was discovered that a false version of the Tor Browser was being promoted by people interested in (and successfully) stealing Bitcoin. They made the malicious "Tor Browser" available through incorrect domains like tor-browser[.]org and torproect[.]org (instead of the authentic domain, torproject.org). To protect yourself from such mistakes as downloading from the wrong site or to protect yourself from a man-in-the-middle attack that supplies you with a malicious app, apps such as the Tor Browser make it possible for you to check the signature of a download, as we discussed in the chapter "Public-Key Cryptography."

What to Learn Next

- Any remaining chapter in part 3

External Resources

- Hancock, Alexis. "Phony HTTPS Everywhere Extension Used in Fake Tor Browser." Electronic Frontier Foundation, October 31, 2019.
- Tails. Accessed February 9, 2021.
- Tor Project. "Tor Project: Overview." Accessed February 9, 2021.
- Whonix. "Tips on Remaining Anonymous." December 26, 2020.
- Whonix. "Whonix: Software That Can Anonymize Everything You Do Online." Accessed February 9, 2021.

Conclusion: Selecting Digital Security Tools

We recommend that you read this section last.

What You'll Learn

1. Criteria for evaluating and selecting communications tools

There are many different digital security tools from which to choose. Decisions about which to use can be overwhelming. When we recommend a tool, we try to pick one so that we can minimize our trust in its providers. The *required criteria* listed below are selected with this in mind. We include *additional technical criteria* that would be nice to meet but aren't as essential. Finally, there are some *nontechnical criteria* that technology providers offer that can help when making decisions among the myriad of choices.

Rarely will a tool be perfect, as we will illustrate with examples. Part of the selection process is finding the tool that is right for the group that will use it. This might result in compromises, even on the required criteria. And note that not every criterion is applicable to every tool. For example, the Tor Browser provides the ability to anonymously browse the internet, but on its own, it does not aim to provide end-to-end encryption, so the first set of criteria doesn't apply.

Finally, choose a tool carefully and test it out before asking a lot of people to adopt it. There is a social cost to asking people to use something new or change their practices, and you want to minimize how often this happens.

Required Criteria

1. **End-to-end encryption** (as described in the chapter "Protecting Your Communications") allows us to follow the security culture principle of minimizing the number of trusted parties. End-to-end encryption allows you to keep your data from the tool provider (but not necessarily the metadata). Implementing end-to-end encryption is our number-one criterion for protecting

the rights of social movement participants (and everyone).

2. If a tool provides the **ability to authenticate your contacts' keys** via fingerprinting, this allows you to prevent man-in-the-middle attacks, as described in the chapter "The Man in the Middle." Further, if the tool does not provide this fingerprinting ability, the provider of the tool could be mounting man-in-the-middle attacks all the time without anyone being able to tell.
3. Having an **open-source client** allows the broader community of cybersecurity professionals to verify, for example, that end-to-end encryption is indeed robustly implemented. What *source code* and *open source* mean are described in the chapter "Modern Cryptography." By "client," we mean the code for the app that would run on your device as distinctive from software that would run on the provider's servers. Having an open-source server software would also be ideal, but many apps keep that software closed to protect their intellectual property. However, since encryption occurs on the device, it is sufficient to verify how much information the server would see by examining the source code for the client.
4. Having the **ability to authenticate the app** does what it purports: it requires more than just access to the source code. You also need to be able to verify that the app you download is the app that comes from the source code. This requires two steps: (1) being able to reproduce that the app that you run on your device can indeed be made from the published (open-)source code and (2) ensuring that the app or source code is authentically that which the provider makes available (isn't subject to a scam, as in the story at the end of the chapter "Protecting Your Identity" or a man-in-the-middle attack). The former is rarely made available for any user and can be very difficult to do. But the latter can be (and is for many tools) verified using cryptographic signatures, as described in the chapter "Authenticity through Cryptographic Signing."
5. An app should be under **active development** and have **responsive developers**, for if it is not, it will not be keeping up with even the most basic updates like changes in operating systems (e.g., Android, iOS). Further, if a problem is found with an app, it needs to be fixed quickly to keep all the app's users' information safe.

Additional Desirable Technical Criteria

1. A digital security tool should be **cross-platform and otherwise widely accessible**. By "cross-platform," we mean that it would be (ideally and if applicable) able to run on Linux, Mac, and Windows operating systems as well as Android and iOS smartphones. Further, tools should play well with screen readers and other accessibility measures. Compromises on this should not be taken lightly. Perhaps at the moment, everyone in your group has Android devices, and so using an Android-only communication tool might be OK—for now. But what if in the future, someone wants to join the group who doesn't have an Android phone (or any smartphone)?
2. **Security audits** are when a (responsible and respected) third party evaluates the security of a given tool by looking at their code and server practices. For some tools, this is an unreasonable ask. For example, Whonix, an anonymity- and security-focused operating system based on Linux, has not undergone a security audit. But no operating system has undergone a complete

security audit.

3. A tool that provides for **anonymity** or **pseudonymity** is more desirable. This can go as far as being Tor compatible. Or it can make pseudonymous accounts easy (e.g., by not requiring a phone number or email address to register).
4. **Tool defaults** make a big impact on users' practices. We have seen a number of secure messaging apps over the years that don't have encryption enabled by default. How many users will forget to enable encryption when they start a new conversation?
5. The **degree of centralization** of a tool can both impact the quality of service and change how much data over which a single entity has control. For example, either a communication app can route all communications through the provider's server, or the server can be used to initialize the communication, and thereafter the data goes directly between the users (through the internet but not through the provider's server). The latter is called *peer-to-peer* communication and can improve call quality (by shortening the distance in the network) and reduce the amount of metadata available to the provider (such as call length). Another option is to allow federation. Consider, as an example, email: one can send emails between different email providers (e.g., Google to Microsoft). On the other hand, Signal only allows two users to communicate if they are both using Signal and establishing their calls with Signal's servers.

Nontechnical Criteria

1. The **financial model** of a provider can impact the ability to access a tool (if you need to pay for access), the long-term stability of the tool (what happens if they run out of money?), and the motivations of the provider (are they monetizing your data or metadata?). Choices range from free, to freemium (pay for more services), to pay to play.
   If a tool is free to use, one should ask why. Is it run by a large company that can monetize even the metadata (such as Facebook having access to contact networks of WhatsApp)? Or is it running on donations and grants?
2. If a tool is **movement oriented**, this can be either positive or negative. As discussed at the end of the chapter "Protecting Your Communications," an unencrypted video-conferencing option through movement-oriented May First was more trustworthy than through Zoom, which has been known to engage in censorship and data sharing with authorities. On the other hand, a VPN option through movement-oriented Riseup might draw more attention from authorities as opposed to hiding among the users of a more populous VPN.
3. **Provider transparency** can help build trust. Most major companies publish transparency reports, as discussed in the chapter "Defending against Surveillance and Suppression." In many cases, these only underscore how much the companies are willing to share their data with your opponents. Another option is the **warrant canary,** which we discussed in the chapter "Authenticity through Cryptographic Signing."

What to Learn Next

There are a number of guides and resources that delve into more detail for particular users or particular aspects that we recommend for further study:

- Digital Defenders Partnership. “Digital First Aid Kit.” Accessed February 9, 2021.
- Electronic Frontier Foundation. “Security Education Companion.” Accessed February 9, 2021.
- Electronic Frontier Foundation. “Surveillance Self-Defense.” Accessed February 9, 2021.
- Tactical Technology Collective. “The Holistic Security Manual.” Holistic Security. Accessed February 9, 2021.
- Tactical Technology Collective and Frontline Defenders. “Digital Security Tools and Tactics.” Security in a Box. Accessed February 9, 2021.

Creative Commons License

This work is licensed by Glencora Borradile (© 2021) under a
Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC)

You are free to:
Share — copy and redistribute the material in any medium or format
Adapt — remix, transform, and build upon the material

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:
Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
NonCommercial — You may not use the material for commercial purposes.
No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Recommended Citations

APA (7th)

Online:

Borradaile, G. (2021, March 29). *Defend Dissent*. https://open.oregonstate.education/defenddissent/.

Print:

Borradaile, G. (2021). *Defend Dissent*. Oregon State University.

APA (6th)

Online:

Borradaile, G. (2021, March 29). Defend Dissent. Retrieved [Retrieval date e.g. March 30, 2021], from https://open.oregonstate.education/defenddissent/

Print:

Borradaile, G. (2021). *Defend Dissent*. Corvallis, OR: Oregon State University.

MLA (8th)

Online:

Borradaile, Glencora. *Defend Dissent*. 29 Mar. 2021, open.oregonstate.education/defenddissent/.

Print:

Borradaile, Glencora. *Defend Dissent*. Oregon State University, 2021.

MLA (7th)

Online:

Borradaile, Glencora. "Defend Dissent." 29 Mar. 2021. Web. [Retrieval date e.g. 30 Mar. 2021].

Print:

Borradaile, Glencora. *Defend Dissent*. Corvallis: Oregon State U, 2021. Print.

Chicago

Online:

Borradaile, Glencora. "Defend Dissent," March 29, 2021. https://open.oregonstate.education/defenddissent/.

Print:

Borradaile, Glencora. *Defend Dissent*. Corvallis, OR: Oregon State University, 2021.

Versioning

This page provides a record of changes made to this publication. Each set of edits is acknowledged with a 0.01 increase in the version number. The exported files, available on the homepage, reflect the most recent version.

If you find an error in this text, please fill out the form at bit.ly/33cz3Q1

Version	Date	Change made	Location in text
1.0	MM/DD/YYYY		