CO 487 - Applied Cryptography

Cameron Roopnarine

Last updated: January 11, 2021

Contents

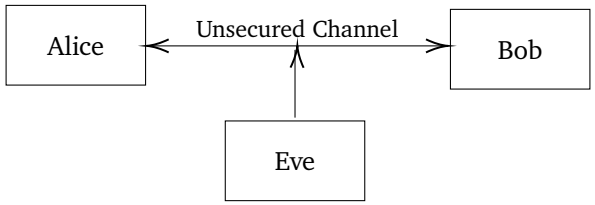
V1: Introduction

V1a: Course Preview

What is Cryptography?

DEFINITION 1.1.1: Cryptography

Cryptography is about securing communications in the presence of *malicious* adversaries.



In the basic communications model we have two parties: Alice and Bob. They are exchanging messages over an *unsecured* channel such as the internet. By *unsecured* we mean that their communications might be under attack by an adversary who we call Eve, or eavesdropper.

Eve can certainly read data exchanged between Alice and Bob, but Eve can also modify data, inject data, delete data, and so on. Therefore, the term Eve can be a bit misleading since Eve can do *a lot* more than just eavesdrop. For the remainder of the course, when the term Eve is used please do not think of Eve as an eavesdropper, but rather as a very powerful, malicious, unpredictable, and evil adversary. What kind of security can Alice and Bob hope to achieve in the presence of such strong adversaries?

Fundamental Goals of Cryptography

1. *Confidentiality*: Keeping data secret from all but those authorized to see it.

   When Alice sends Bob a message, even though Eve can read all the transmitted data, Alice and Bob are the only parties who can read the actual underlying message. In practice, confidentiality is achieved through the use of encryption.

2. *Data integrity*: Ensuring data has not been altered by unauthorized means.

   When Bob receives a message from Alice, Bob has the assurance that the message wasn't modified during transmission.

3. *Data origin authentication*: Corroborating the source of data.

   When Bob receives a message supposedly sent by Alice, Bob can verify that the message indeed was sent by Alice.

4. *Non-repudiation*: Preventing an entity from denying previous commitments or actions.

Broadly speaking, this is preventing an entity from denying previous commitments or actions. In the physical world, non-repudiation is provided through the use of handwritten signatures. Cryptography though can be used to provide the same security as handwritten signatures, but in the digital world.

We'll see in the remainder of the course that cryptography can be used to provide confidentiality, data integrity, data origin authentication, and non-repudiation.

It should be evident that these four security services provided by cryptography are very important in many applications where sensitive data might be communicated or stored. Let's look at some examples.

EXAMPLE 1.1.2: Email

Firstly, when Alice connects to the website, she needs the assurance that she really is visiting g-mail's website, not the website of an impersonator. Next, when she enters her email address and password, these are transmitted over the internet and so need to be secured through the use of encryption. Lastly, when Alice uploads and downloads emails, these are also transmitted over the internet and so have to be encrypted and authenticated.

EXAMPLE 1.1.3: Online Shopping

Cryptography helps engender trust in e-commerce applications for both consumers and vendors.
This is accomplished by encrypting and authenticating all sensitive financial and personal data that might be transmitted between Alice and Bob.

EXAMPLE 1.1.4: Automatic Software Upgrades

Cryptography helps in guaranteeing the safety of automatic software upgrades.
Imagine Microsoft broadcasting upgrades of its operating system to hundreds of millions of devices around the world. When such a device receives an upgrade, it needs the assurance that the software upgrade indeed is from Microsoft and hasn't been modified during transmission. In practice, these assurances are provided through the use of digital signatures.
Note: Confidentiality is not required in this scenario because the software upgrades themselves are not considered to be private.

EXAMPLE 1.1.5: Cell Phone Service

When you make a call on your cell phone the connection is established to a nearby cell phone tower. The cell phone tower needs the assurance that you have the authorization to use the cell phone service. Furthermore, your voice data needs to be encrypted and authenticated as it is transmitted over the air to the cell phone tower.

EXAMPLE 1.1.6: WiFi

When you connect to a WiFi network, your cell phone has to authenticate itself to the WiFi router to prove that it has the authorization to use the WiFi service. Furthermore, all internet communications between your cell phone and the router have to be encrypted and authenticated.

EXAMPLE 1.1.7: Bluetooth

When two devices such as your cell phone and your headset establish a Bluetooth connection this means that all data transmitted between the two devices is encrypted and authenticated using cryptographic mechanisms.

EXAMPLE 1.1.8: Messaging

Cryptography is used to secure instant messaging, most famously by WhatsApp which provides very strong *end-to-end encryption*. End-to-end encryption is when *only* the two users who are exchanging messages can read them. In particular, WhatsApp itself cannot read the exchange messages.

Hopefully, these examples convince the reader that cryptography is extremely useful and important in securing all kinds of applications where sensitive data might be communicated or stored.

One shouldn't think of Alice and Bob as human beings but rather as two communicating devices.

Table 1.1: Communicating Parties

Alice	*Bob*	*Communications channel*
person	person	telephone cable
person	person	cellular network
person	website	internet
iPhone	wireless router	wireless
iPhone	headphones	wireless
car's brakes	another car	wireless
smart card	bank machine	financial network
smart meter	energy provider	wireless
military commander	satellite	space

Secure Web Transactions

EXAMPLE 1.1.9: Secure Web Transactions

Cryptography is used to secure web transactions via *Transport Layer Security*.

DEFINITION 1.1.10: Transport Layer Security (TLS)

The cryptographic protocol used by web browsers for secure web transactions for secure access to amazon, g-mail, Facebook, etc.

TLS is used to assure an individual user (*client*) in this case Alice, of the authenticity of the website (*server*), in this case g-mail, he or she is visiting, and to establish a *secure communications channel* for the remainder of the session.
All data communicated between Alice and the website need to be encrypted and authenticated which is accomplished using *symmetric-key cryptography*.

DEFINITION 1.1.11: Symmetric-key Cryptography

The *client* and *server*, a priori, share some *secret* information k, called a *key*.

Once Alice and Bob have such a secret key, they can subsequently engage in secure communications by encrypting their messages with Advanced Encryption Standard (AES) and authenticating the resulting ciphertexts with HMAC. We'll study AES and HMAC later on in the course.

Question: Alice and Bob establish the shared secret key k in the first place?

Alice cannot simply select k and transmit it over the internet to Bob because then the eavesdropper would learn k.

This is where *public-key cryptography* is very useful.

DEFINITION 1.1.12: Public-key Cryptography

Communicating parties a priori, share some *authenticated* (but non-secret) information.

Now, to establish a secret key, the client Alice selects the secret session key k, and encrypts it with the servers *RSA public key*, and sends the resulting ciphertext over the internet to Bob. Only Bob the server can decrypt the resulting ciphertext, because decryption requires the use of the corresponding RSA private key which only Bob knows. And so Bob decrypts the ciphertext to recover k, and k is known only to Alice and Bob.

Question: How does Alice obtain an authentic copy of the server's RSA public key?

Bob cannot just send the public key to Alice over the internet because Eve could intercept this public key and replace it with her own public key.

This problem is solved using signature schemes.

DEFINITION 1.1.13: Signature Scheme

The server's RSA public key is *signed* by a *Certifying Authority* (CA) using the **RSA signature scheme**.

The client can verify the signature using the CA's RSA public verification key which is embedded in Alice's browser.

In this way, the client obtains an authentic copy of the server's RSA public key.

The TLS protocol

1. When a client first visits a secured website, the server transmits its *certificate* to the client.
   - The certificate contains the server's identifying information (e.g. website name and URL) and RSA a public key, and the RSA signature of a *certifying authority*.
   - The certifying authority (e.g. Verisign) is trusted to carefully verify the server's identity before issuing the certificate.
2. Upon receipt of the certificate, the client *verifies* the signature using the certifying authority's public key, which is embedded in the browser. A successful verification confirms the *authenticity* of the server and of its RSA public key
3. The client selects a random *session key* k, *encrypts* it with the server's RSA public key, and transmits the resulting ciphertext to the server.
4. The server *decrypts* the ciphertext to obtain the session key, which is then used with symmetric-key schemes to *encrypt* (e.g. with AES) and *authenticate* (e.g. with HMAC) all sensitive data exchanged for the remainder of the session.
5. The establishment of a secure link is indicated by a *closed padlock* in the browser. Clicking on this icon reveals the server's certificate and information about the certifying authority.

TLS is one of the most successful security technologies ever deployed, but is TLS *really* secure?

TLS: Potential Vulnerabilities

1. The cryptography is weak (e.g. AES, HMAC, RSA).

   Recall from MATH 135 that the security of RSA is based on the presumed difficulty of factoring large numbers. No one knows how to factor large numbers efficiently today, however there is a possibility that such an algorithm might be discovered tomorrow.

2. *Quantum attacks* on the underlying crpytography.

We do know how to factor large numbers very quickly on quantum computers,although it isn't known as yet whether one can actually build large-scale quantum computers.

3. Weak random number generation.

   In TLS, browsers select random session keys. In practice, selecting random numbers is a very difficult problem in software applications.

   Many deployments of TLS have been found to be insecure in the past 20 years due to poor random number generation.

4. Issuance of fraudulent certificates.

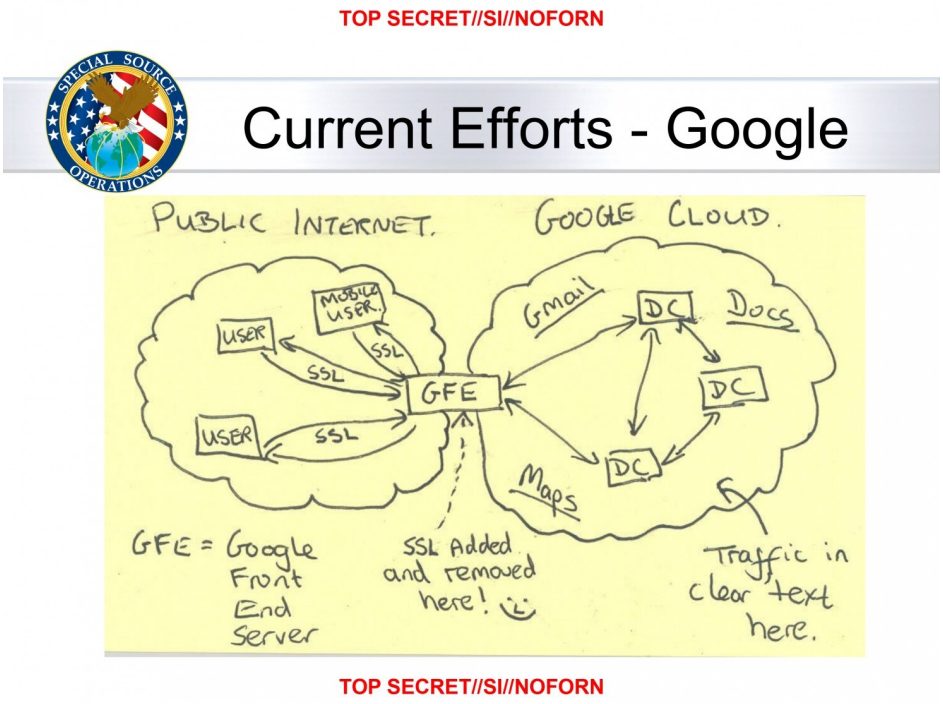
   - In 2001, Verisign erroneously issued two Class 3 code-signing certificates to a person masquerading as a Microsoft representative.

     In principle, this person could have generated malware, signed it with its RSA private key, broadcast this malware to computers around the world, and the computers would have accepted the malware as originating from Microsoft. This problem was caused by *human error*, not due to any deficiency in the underlying cryptography.

5. Software bugs (both inadvertent and malicious).
6. *Phishing* attacks.
7. TLS only protects data during transit. It does *not* protect your data when it is collected at the server.

   - Many servers store large amounts of credit card data and other personal information.

8. The National Security Agency (NSA).



We give an example from the Snowden revelations. In 2013, Edward Snowden, an NSA employee, leaked a large number of highly classified NSA documents to the press. One such document was headed "Current Efforts Google."

On the left, the slide showed the public internet. So it showed users connecting using TLS, also known as SSL, to Google's front-end servers. So all data exchange between users and Google servers are protected

using TLS.

However, the slide also had the statement "SSL added and removed here" with a smiley face, and the note that "traffic in cleartext here."

So Google's front-end servers would then transmit the decrypted data to its data centers around the world. These transmissions took place in the so-called Google cloud, where the data centers were connected by private links. In particular, the data transmitted between these links were not encrypted. As it turns out, NSA had access to one of these links between two Google data centers and NSA was capturing and storing gigantic amounts of user data transmitted between these two data centers.

Therefore, even though TLS itself was secure, what was not secure was the way in which data was stored and transmitted in Google's back end.

Cryptography in Context

These examples of potential vulnerabilities in TLS tell us that cryptography must be studied in a context, and that broader context is information security, now known as Cybersecurity. Cybersecurity is a very broad field and it's difficult to give it a precise definition. Nonetheless, here's one attempt.

DEFINITION 1.1.14: Cybersecrity

Cybersecurity is comprised of the concepts, technical measures, and administrative measures used to protect networks, computers, programs and data from deliberate or inadvertent unauthorized access, disclosure, manipulation, loss or use.
Also known as *information security*.

Table 1.2: Cybersecrity Includes the Study of:

Computer security	*Network security*	*Software security*
Security models and polices	Internet Protocols and their security	Detecting and preventing buffer overflows
Secure operating systems	Viruses and worms	Programming languages and compilers
Virus protection	Denial-of-service (DoS) attacks	Specifying and enforcing security policies
Auditing mechanisms	Firewalls	Digital rights management
Risk analysis	Intrusion detection systems	Code obfuscation
Risk management	Wireless communications	Software tamper resistance
		Trusted computing

Cryptography $\neq$ Cybersecrity

- Cryptography provides some mathematical tools that can assist with the provision of cybersecurity services. It is a *small*, albeit an *indispensable*, part of a complete security solution.
- *Security is a chain*
  - Weak links become targets; one flaw is all it takes.
  - Cryptography is usually not the weakest link. However, when the cryptography fails the damage can be catastrophic.
  - This course will focus on cryptography.

Syllabus

Cryptography primitives

- Symmetric-key encryption
- Hash functions
- Message authentication
- Authenticated encryption
- Public-key encryption
- Signature schemes
- RSA
- Elliptic curve cryptography
- Key establishment
- Post-quantum cryptography

Cryptography deployments

- IEEE 802.11 WEP/WPA2
- Google's KMS
- GSM security
- QQ browser
- FIDO U2F
- Bluetooth security
- Key management (PKIs)
- Web security protocols (TLS)
- Signal protocol (WhatsApp)
- Cryptocurrencies (Bitcoin)

V1b: Course Outline

About the Course

- Coverage will favour breadth at the expense of depth
  - For depth, try the optional readings
  - See also: *CO 485* (Mathematics of Public-Key Crypto)
  - See also: *CS 458* (Computer Security and Privacy)
- This course is not a traditional textbook course!
  - *Watching the video lectures* is strongly recommended
  - There are no good sources of "practice questions"
  - *Your job*: Identify and understand the important (technical and non-technical) *concepts* presented in the lectures
- Optional text: Understanding Cryptography
  - Available for free download from the UW library
  - Optional readings are posted on the course website

Learning Outcomes

1. Understand the fundamental cryptographic tools of symmetric-key encryption, message authentication, authenticated encryption, hash functions, public-key encryption, and signatures;
2. Appreciate the challenges with assessing the security of these tools;
3. Gain exposure to how these cryptographic tools are used to secure large-scale applications;
4. Understand why key management is an essential process that underpins the security of many applications.

Course Administration

- Website: http://learn.uwaterloo.ca
  - Course outline and *weekly schedule*.
  - Videos, slides, assignments & solutions, handouts.
  - Optional readings.

- Assistance:
  - Instructor and TA online office hours (see LEARN).
  - Piazza (see the Course Outline for the password).
  - Participation is *strongly encouraged*, but optional.
- Communications: All important course announcements will be sent to you by *email*. (Apologies in advance for the spam.)
- Assignments (50%): submitted via Crowdmark.
  - Due dates: *Jan 29*, *Feb 12*, *Mar 5*, *Mar 26*, *Apr 9*.
- Quizzes (15 / 7.5 / 0%): *Feb 26* and *Mar 19* (on LEARN).
- Final Assessment: (35 /42.5 / 50%).

Academic Integrity

- Assignments:
  - No collaboration permitted on one or two questions.
  - Collaboration permitted (and *encouraged*) on the other questions (but solutions must be written up independently).
- Quizzes:
  - *Open book*: slides, your notes, assignments, solutions (ONLY).
- Final Assessment:
  - *Open book*: slides, your notes, assignments, solutions (ONLY).
- General: no cheating, Course Hero, Chegg, online discussion groups, solutions from previous course offerings, etc.
- Policy 71 *will be enforced*.

Should You Take This Course?

Course philosophy: CO 487 is an *elective course* for everyone.

As such, it is intended to be:

- *interesting*,
- *fun*,
- *relevant*, and
- *not-too-difficult*.

Amount of Material

Course content:

- 65%: Material you need to know for quizzes and the final exam.
- 25%: Show-and-tell, including “Crypto Cafe.”
- 10%: Cryptography gossip.

Guidance for the quizzes and the final exam:

There will not be any "unreasonable" expectations of what you need to memorize for the quizzes and the final assessment. For example, you do *not* need to memorize the descriptions of RC4, the WEP security protocol, ChaCha20, AES, AES-GCM, SHA-256, RSA-OAEP, the elliptic curve addition formulae, ECDSA, the Bluetooth security protocol, the Signal protocol, etc. If asked any questions on such topics, then all the relevant background information will be provided. You also don't need to memorize any *trivia*, e.g. historical notes, dates, spelling out of acronyms, etc.

V2: Symmetric-Key Cryptography

V2a: Basic Concepts

DEFINITION 1.1.15: Symmetric-key Encryption Scheme (SKES)

A **symmetric-key encryption scheme** (SKES) consists of:

- M: the plaintext space,
- C: the ciphertext space,
- K: the key space,
- a family of encryption functions, E_k: $M \to C$, $\forall k \in K$,
- a family of decryption functions, D_k: $C \to M$, $\forall k \in K$, such that $D_k(E_k(m)) = m$ for all $m \in M$, $k \in K$.

Using a SKES to Achieve Confidentiality

1. Alice and Bob agree on a *secret key* $k \in K$ by communicating over the *secure channel*.
2. Alice computes $c = E_k(m)$ and sends the ciphertext c to Bob over the *unsecured channel*.
3. Bob retrieves the plaintext by computing $m = D_k(c)$.