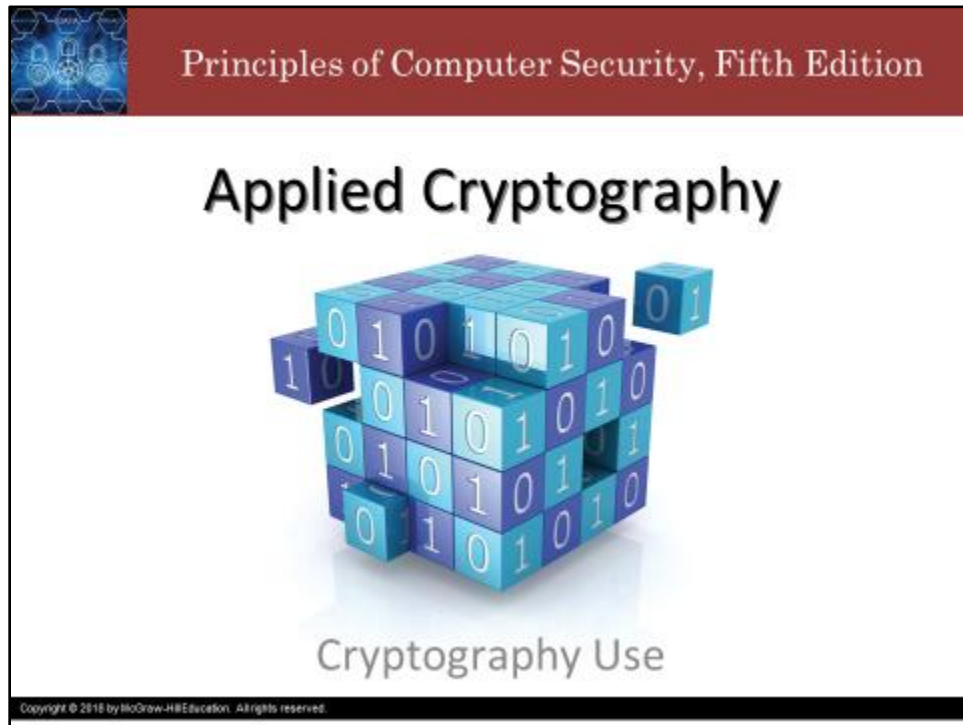



## Applied Cryptography: Cryptography Use

Slide 1



Howdy! In this video, we discuss the use of cryptography.



Principles of Computer Security, Fifth Edition

## Cryptography Use

- The best way to secure data online
- Components of security:
  - CIA
  - AAA
    - Crypto addresses all except availability
- Cryptographic algorithms:
  - Key exchange, key management, encryption, digital signatures, digital rights management, intellectual property protection, ...

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

The best way to secure data online with current technology is to use cryptography.

The components of security include:


The CIA triad: Confidentiality, integrity, availability

And the triple As: authentication, authorization, accounting (or nonrepudiation)

cryptography addresses all components except availability

Cryptographic algorithms play a key role in:

Key exchange, Key management (including key escrow); encryption, digital signatures, digital rights management, intellectual property protection, and more



Principles of Computer Security, Fifth Edition

## Confidentiality

- Encryption.
- Symmetric encryption favored due to speed and size.
  - Asymmetric is used for short messages, e.g. key exchange
- Secrecy of data ensured by:
  - Algorithm strength
  - Key length

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

*Confidentiality* typically comes to mind when the term *security* is brought up. Confidentiality is the ability to keep some piece of data a secret. In the digital world, encryption excels at providing confidentiality. In most cases, symmetric encryption is favored because of its speed and because some asymmetric algorithms can significantly increase the size of the object being encrypted. Asymmetric cryptography also can be used to protect confidentiality, but its size and speed make it more efficient at protecting the confidentiality of small units for tasks such as digital key exchange. In all cases, the strength of the algorithms and the length of the keys ensure the secrecy of the data in question.



Principles of Computer Security, Fifth Edition


## Integrity

- Integrity is crucial to security
- Alice and Bob need to know that messages are not altered in transmission.
- Hashing, Digital Signatures, Encryption

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Integrity is a crucial component of message security. When a message is sent, both the sender and recipient need to know that the message was not altered in transmission. This is especially important for legal contracts—recipients need to know that the contracts have not been altered. Signers also need a way to validate that a contract they sign will not be altered in the future. Message integrity will become increasingly important as more commerce is conducted digitally. The ability to independently make sure that a document has not been tampered with is very important to commerce. More importantly, once the document is “signed” with a digital signature, it cannot be refuted that the person in question signed it. This is known as non-repudiation, which also belongs to the accounting component of security.

Integrity mechanisms employ a combination of cryptographic methods, including hashing, encryption, and digital signatures. For example, hash functions are used to compute message digests, which protects the integrity of the message by making it easy to determine whether any part of the message has been changed. The message now has some extra information (the hash value) to tell the users to resend the message if it was intercepted and interfered with.



Principles of Computer Security, Fifth Edition

## Authentication

- Alice and Bob need to know who they are talking to.
- Integrity of origin.
- Authentication requires proof.
- Hashing, Digital Signatures

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Authentication can be thought of as a special case of integrity, but one which deals with identity rather than data.

Authentication is the matching of a user to an identity through previously shared credentials.

In this sense, authentication deals with the integrity of the origin of data.


Identification is when you claim to be someone. For example, I identify as Dr. Ritchey.

Authentication is when you prove that claim. I can prove that I am Dr. Ritchey by showing you credentials that establish that I am Ritchey (like a government issued ID) and that I am doctor (like my doctorate degree).

In society, we typically take people at their word. When someone introduces themselves to you with a certain identity, you typically trust the veracity of that identity (unless you are wary of some sort of bamboozle, and you should, since social engineering is a violation of that trust).

In security, we trust reluctantly. In a secure environment, every interaction would need to be preceded by authentication. Even if I've talked you a hundred times before, I'm going to check your ID before I talk to you for the hundred and first. Complete mediation. Check every access.

Hashing and digital signatures are the most common cryptographic functions used for authentication.



Principles of Computer Security, Fifth Edition

## Authorization

- Access policy & control.
  - Authorize := define an access control policy
- Encryption, authentication.


Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Authorization is the security function of specifying access privileges of subjects to objects (like system resources and other subjects).

“To authorize” means to define an access policy.

The primary mechanism for enforcing an access policy is access control.

There are many ways to implement access control, but encryption and authentication (which often entails other crypto functions) are commonly used to support access control systems.



Principles of Computer Security, Fifth Edition


## Accounting

- Nonrepudiation
  - Sender cannot later deny that they sent the message.
  - Important in electronic exchanges of data.
  - Cannot be implemented with symmetric algorithms.
  - Digital Signatures
- Auditing
  - Keep track of what users do
    - Can include non-repudiation
  - Hashing, encryption, digital signatures

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

An item of some confusion, the concept of nonrepudiation is actually fairly simple. Nonrepudiation means that the message sender cannot later deny that they sent the message. This is important in electronic exchanges of data, because of the lack of face-to-face meetings. Nonrepudiation is based upon public key cryptography and the identification of a subject with their private key (only Alice knows Alice's private key). A message signed using your private key, which nobody else should know, is an example of nonrepudiation. A third party can check your signature using your public key and disprove any claim that you were not the one who actually sent the message (since only you could have signed it). Nonrepudiation is tied to asymmetric cryptography and cannot be implemented with symmetric algorithms.

Another aspect of accounting is a bit more general than non-repudiation, and that is auditing, which is keeping track of what happens in the system. Auditing is an application of the principle of compromise recording, and is also a very lightweight version of complete mediation (where the mediation is simply to record that the access occurred). The audit logs must be kept secure, and cryptography has many of the tools needed for the job. High-level functions for protecting the audit logs include authorization, and authentication.



Principles of Computer Security, Fifth Edition

## Digital Signatures

- Save the trees (and the ink).
- Protect integrity.
- Use hash functions and asymmetric cryptography.
  - Sign: `encrypt(hash(document), private_key)`
  - Verify: `hash(document) == decrypt(signature, public_key)`

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Digital signatures could be the key to truly paperless (and inkless) document flow.

But, even more critical than that, is that unprotected digital documents are very easy for anyone to change.


If a document can be edited after an individual signs it (which is always the case), it is important that any modification can be detected and refuted.

To protect against document editing, hash functions are used to create a digest of the message that is unique and easily reproducible by both parties. This protects the message integrity.

The properties of asymmetric encryption allow anyone to use a person's public key to generate a message that can be read only by that person, as this person is theoretically the only one with access to the private key.

In the case of **digital signatures**, this process works the same but with the keys swapped. When a user can decrypt the hash with the public key of the originator, that user knows that the hash was encrypted by the corresponding private key. This use of asymmetric encryption is a good example of nonrepudiation, because only the signer would have access to the private key. This is how digital signatures work, by using integrity and nonrepudiation to prove not only that the right person signed the digital document, but also that the digital document was not altered after being signed.





Principles of Computer Security, Fifth Edition

## Digital Rights Management (DRM)

- Protect intellectual property from unauthorized use.
- Examples:
  - broadcast stream of digital satellite TV
  - Software as a Service (SaaS)
  - Hardware authentication tokens

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

This is a broad area, but the most concentrated focus is on preventing piracy of software or digital content.

Before easy access to computers, or the “digital revolution,” the content we came in contact with was analog, such as print media

A common example of **DRM** that is mostly successful is the broadcast stream of digital satellite TV.

Since the signal is beamed from space to every home in North America, the satellite TV provider must be able to protect the signal so that it can charge people to receive it.

Smartcards are employed to securely hold the decryption keys that allow access to some or all of the content in the stream.

This system has been cracked several times, allowing a subset of users free access to the content; however, the satellite TV providers learned from their early mistakes and upgraded new smartcards to correct the old problems.


Similar to companies that provide satellite TV service, companies that provide Software as a Service rely on a subscription basis for profitability.

If someone could pay for a single license and then distribute that to hundreds of employees, the provider would soon go out of business .

Many systems in the past have been cracked because the key was housed inside the software.

Another example is hardware token USB keys that must be inserted into the machine for the software to decrypt and run.

Placing the keys in hardware makes an attack to retrieve them much harder, a concept that is employed in the Trusted Platform Module; in fact, one of the primary complaints against the TPM is its ability to enforce DRM restrictions.



Principles of Computer Security, Fifth Edition

## Cryptographic Applications


- Many programs to encrypt data conveniently on your personal computer.
  - Pretty Good Privacy (PGP), GNU Privacy Guard (GPG)
  - File system-level encryption (Ext4, Encrypting File System)
  - Full disk encryption (BitLocker)
  - Database encryption: AES
  - Network: VPNs

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

There are free, open-source, and also non-free, programs that allow you to conveniently use cryptographic algorithms on your personal computer. The programs support a wide range of crypto applications, from email security, to file encryption, all the way down to full disk encryption. Databases also support encryption at several levels. The de facto standard for database encryption is to use AES. Many people who are concerned about online privacy use a VPN (a virtual private network) to protect their network traffic. VPNs use encryption and obfuscation to hide the content and destination of your traffic. Not using a VPN, especially when travelling or connecting to untrusted access points, is risky.

*Pretty Good Privacy (PGP)* is mentioned in the textbook because it is a useful protocol suite. GNU Privacy Guard (GPG) is an open source openPGP-compliant program.

PGP applications can be plugged into popular e-mail programs to handle the majority of day-to-day encryption tasks using a combination of symmetric and asymmetric encryption protocols. We'll talk a bit more about PGP in another video.



Principles of Computer Security, Fifth Edition

## Use of Proven Technologies

- “Never roll your own crypto”
  - Relevant xkcd: <https://xkcd.com/153/>
- Foundational elements:
  - Proven cryptographic libraries
  - Proven cryptographically correct random number generators
- Good crypto is approved only after lengthy test and public review.
  - E.g. AES, SHA

Copyright © 2018 by NoStarch.com. All rights reserved.

It is important to use proven technologies when setting up cryptographic protections.

Foundational elements associated with a solid use of cryptography include:

Proven cryptographic libraries


Proven cryptographically correct random number generators

Most good cryptographic algorithms are approved for use only after a lengthy test and public review phase, as was the case for AES and SHA.

This point is a fundamental principle and great truth of applied cryptography: never roll your own crypto. Any person can invent a security system so clever that they can't think of how to break it. It takes a lot of time and effort, on a global scale, to vet a crypto algorithm. One does not simply sit down one day and bang out a secure algorithm. It takes many people and many years.

As a great truth, its opposite is also a great truth: you should roll your own crypto. Why? So you'll learn through failure. The development of strong crypto is an intense and enduring process of failure. The strength of a crypto algorithm is measured in the infeasibility of attack, how much work the attacker must do. Designing strong crypto requires deep knowledge of cryptanalysis, which is developed by breaking cryptosystems (exploiting weaknesses which the designer was unable to see themselves). So, roll your own crypto! But don't ever use it for a security-critical function. Pass it around your friends and have some fun and learn what you can about it's weaknesses. Do this for long enough with patient, talented and experienced crypto-friends, and you'll be well on your way to becoming a novice cryptographer. Until then, you (and I) are the noobiest of noobs and we should be content to sit in our sandboxes eating sand and making weak crypto and learning all that we can from experts and experience.

Slide 12



Principles of Computer Security, Fifth Edition

## Attribution

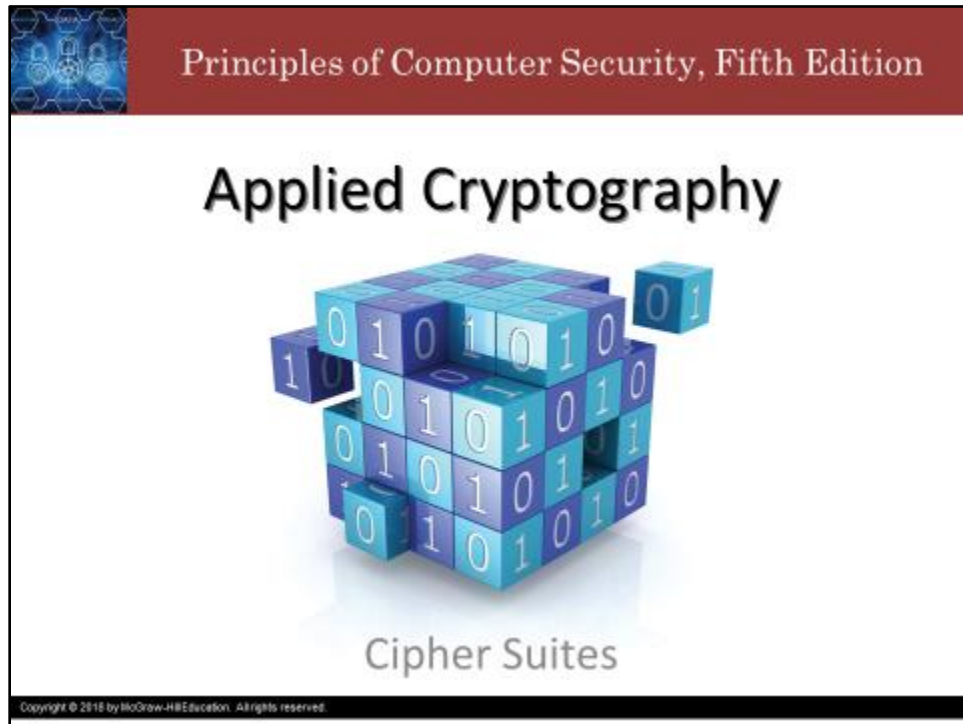
- The course slides are based on slides developed by the textbook authors, W. "Art" Conklin, Ph.D. (University of Houston), and Greg White, Ph.D. (University of Texas at San Antonio).
- These slides remain the intellectual property protected by US Copyright law of the authors and the textbook company.
- There have been some changes to the slide deck.

Copyright © 2018 by McGraw-Hill Education. All rights reserved.


Thank you and take care.

## Applied Cryptography: Cipher Suites

Slide 1



Howdy! In this video, we discuss cipher suites.



## Principles of Computer Security, Fifth Edition

### Cipher Suites

- Cryptography is applied as a set of functions.
  - Key agreement, encryption, digital signatures, hashing.
- **Cipher suite** := an arranged group of algorithms
  - E.g. TLS Cipher Suite Registry

▼ Connection:

Protocol version: "TLSv1.3"

Cipher suite: "TLS\_AES\_128\_GCM\_SHA256"

Key Exchange Group: "x25519"

Signature Scheme: "ECDSA-P256-SHA256"

- Don't use weak or deprecated algorithms

Copyright © 2018 by NoStarch.com. All rights reserved.


In many applications, the use of cryptography occurs as a set of functions.

Different algorithms can be used for key exchange, encryption, digital signatures, and hashing.

The term **cipher suite** refers to a pre-arranged group of algorithms, such as the TLS Cipher Suite Registry.

Once a TCP connection is established, the end points can secure that connection using TLS, which stands for Transport Layer Security, the transport layer is the layer of the network at which TCP operates, so TLS runs just above that at the application layer. TLS is what puts the S in the HTTPS protocol. TLS starts with a handshake, just like TCP. We'll see more details about TLS in another video, but for now, let us suffice it to say that it is during this handshake that the client and the server agree on the set of crypto algorithms to use in order to secure their connection. They must agree on several things, including the key agreement algorithm, authentication algorithm, the symmetric cipher and key size, and the hash algorithm to use. The set of algorithms selected for the TLS connection are the cipher suite which is used for that connection. The cipher suite my browser and the Gmail server agreed on is AES in Galois Counter Mode with SHA256 as the hash function and 128-bit keys. If I dig a bit deeper, I can find that the signature algorithm is the Elliptic Curve Digital Signature Algorithm using the P256 curve and SHA256 and the key-exchange was done using Elliptic Curve Diffie-Hellman using Curve25519.

Over time, ciphers can become vulnerable to attacks. End users typically don't curate which algorithms their application supports. That is a decision most people are happy to leave to the developers and maintainers. You, as a future, or possibly even current, developer, operator, and/or maintainer of secure systems, are the kind of person who may get to make those decisions. It is important to keep the list of supported algorithms up to date and to be quick to remove algorithms which become weak or deprecated.




Principles of Computer Security, Fifth Edition

## Secret Algorithms

- Algorithms can be broken into two types
  - Those with published details
  - Those where the steps are kept secret
- Secrecy can help thwart reverse engineering
- Most secure algorithms have survived the onslaught of attackers
- Best algorithms are published for peer review

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Not all crypto algorithms follow the principle of open design. Proprietary algorithms and algorithms used by the military and intelligence communities are not public. While secrecy (security by obscurity) can make it harder to break the algorithm, if the security relies in any way on that secrecy, then the algorithm is essentially insecure. For this reason, best practice is to never use algorithms whose details are kept secret. This is a corollary to the rule to never your own crypto: don't partake in anyone else's home-rolled crypto, either. You don't know what they put in there, either on purpose or by mistake. The best crypto algorithms are those which have been published for peer review and have survived the tests of time and intense cryptanalysis. All of the algorithms approved for use in TLS fall into this category.



Principles of Computer Security, Fifth Edition


## Key Exchange

- **Key exchange** is the central foundational element of a secure symmetric encryption system.
- In asymmetric systems, the key exchange problem is one of key publication.
- The Diffie-Hellman key exchange is one example of secure key exchange via digital methods and mathematical algorithms.
- See also: Public Key Infrastructure

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

A necessary first step (or maybe not exactly first, but it's one of the first steps), is to exchange keys. For symmetric crypto, this must be done before communication can occur. Without the shared secret, Bob cannot decrypt Alice's messages. In asymmetric crypto, key exchange is either solved by a protocol like Diffie-Hellman or transformed into key publication, as is the case for public key algorithms like RSA. When public keys are used, those keys need to be stored somewhere. The standard solution is to have everyone keep their own public keys and send them as part of the secure connection setup phase. That makes things easier since now there doesn't need to be anyone whose job it is to store public keys (but those types of servers do still exist, such as repositories of PGP keys for secure email). When I send you my public key, how do you know it's really my key? How do you know that I am me and not someone else masquerading as me who has sent their own key to ensnare us both in a man-in-the-middle attack? That is a great question, and is the topic of another module on public key infrastructure. The answer, since I can't bear to leave you in suspense, is to have a trusted entity, called a certificate authority, digitally sign my public key to certify that the identity associated with my public is me. To verify that it is really my public key, you simply examine the certificate that includes my identity, public key, and the certificate authority's signature, and you verify that their signature is valid. If you trust the certificate authority, then you will also trust that the key is mine. Your computer and your browser have a list of certificate authorities that they trust by default.





Principles of Computer Security, Fifth Edition

## Key Escrow

- The loss of a key can happen due to many reasons:
  - It might simply be lost; the key holder might be incapacitated or dead; the software or hardware might fail.
- **Key escrow** refers to keeping a copy of the encryption key with a trusted third party.
  - Can be used to retrieve your key in case of emergency
  - Can be used by law enforcement
  - Can negatively affect your security

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

What do you do if you lose your keys? You look for them. Your physical keys, like your car keys or the keys to your home and office, exist in the physical world and, if you look hard enough or long enough, you can eventually find them.

Digital keys are not so forgiving. Since they exist as mere bits, and usually as encrypted bits, it may not be possible to recover a lost key.

Losing a digital key is like accidentally melting your car keys. You may still have some metal, but it's no longer the keys to your car.


If you forget the passphrase to unlock your secure keystore, the key is locked up until the cryptography can be broken, and as you may know, that could be millennia.

This has raised the question of **key escrow**, or keeping a copy of the encryption key with a trusted third party. Like giving a friend a spare key to your house or car in case you lose yours.

Theoretically, this third party would only release your key to you, or your official designate in the event of your being unable to get the key yourself.

However, just as the old saying from Benjamin Franklin goes, "Three may keep a secret if two of them are dead.", anytime more than one copy of the key exists, the security of the system is broken. The extent of the insecurity of key escrow is a subject open to debate, and will be hotly contested in the years to come.

Additionally, with computer technology being miniaturized into smartphones and other relatively inexpensive devices, criminals and other ill-willed people have begun using cryptography to conceal communications and business dealings from law enforcement agencies. Because law enforcement agencies have not been able to break the encryption in many cases, government agencies have begun asking for mandatory key escrow legislation. Apple has been in the news recently for their public refusal to weaken the security of the data protection mechanisms on their devices.



Principles of Computer Security, Fifth Edition


## Session Keys

- A **session key** is a symmetric key used for encrypting messages during a communication session.
  - Generated from random seeds
  - Used for the duration of a communication session
- Session keys offer the advantages of:
  - Symmetric encryption, speed, strength, simplicity
  - Significant levels of automated security

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

A **session key** is a symmetric key used for encrypting messages during a communication session. Session keys are generated from random seeds and are used for the duration of a communication session, such as the time you spend browsing a website.

Session keys offer the advantages of symmetric encryption like speed, strength, and simplicity. And, with key exchange protocols, they also offer significant levels of automated security, such as the ability to automatically update the session after some period of time during long-lived connections.




Principles of Computer Security, Fifth Edition

## Ephemeral Keys

- **Ephemeral keys** are cryptographic keys that are used only once after they are generated.
- Examples of applications are:
  - Ephemeral Diffie-Hellman (EDH) key exchange
  - Elliptic Curve Diffie-Hellman Ephemeral (ECDHE)
- Provides perfect forward secrecy.

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

**Ephemeral keys** are cryptographic keys that are used only once after they are generated. Diffie-Hellman is the commonly used key exchange protocol, and the most common application of it is to generate ephemeral keys during the TLS handshake. Since each key is only used exactly once, ephemeral keys provide perfect forward secrecy. Cracking one message, or recovering one key will not jeopardize any of the other keys or messages. Every time you make a new secure connection request (which can happen several times when loading a single webpage), a new ephemeral key is generated, used once, and then discarded. This significantly increases Eve's work factor at minimal expense to Alice and Bob.



Principles of Computer Security, Fifth Edition

## Key Stretching


- **Key stretching** is a mechanism that takes what would be weak keys and “stretches” them to make the system more secure against brute-force attacks.
- A typical methodology involves increasing the computational complexity by adding iterative rounds of computations
- Common forms employed today are:
  - **Password-Based Key Derivation Function 2 (PBKDF2)**
  - **Bcrypt**

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Key stretching techniques are used to make a possibly weak key, like a human-generated password or passphrase, more secure against a brute-force attack by increasing the time and space resources required to test each possible key.

One common way of doing this is to apply a crypto algorithm, like a hash or encryption, many times. For example, GPG hashes a password 65536 times before checking it. When the correct password is entered, the time it takes to run the 65000 hashes feels almost negligible. But, the 100s of milliseconds it takes to run all those hashes for each guess makes a brute force attack infeasible since every test requires 65000 hashes.

A particularly useful application of key stretching is key derivation functions which are used to convert passwords and ephemeral keys into longer keys or a sequence of keys to be used for other cryptographic operations.



Principles of Computer Security, Fifth Edition


## Transport Encryption

- **Transport encryption** is used to protect data that is in motion.
- When data is being transported across a network, it is at risk of interception.
- When utilizing the TCP/IP protocol, TLS is the preferred method of managing the security at the transport level.

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Data should be protected at rest, in transit, and in use. Transport encryption is the application of encryption to protect data in transit.

The main threat to data in transit is interception, which could violate any or all three of the CIA triad. As the data traverses the network, it can be read, modified, delayed, or destroyed. The TCP protocol takes care of reliable transport, getting data from point A to point B, taking care of availability as best it can. But, it does nothing to protect the confidentiality or integrity of the data (it does some integrity, hence the reliable transport, but the checksum is trivial for an attacker to update). That is where TLS comes in. TLS protects the data against interception. The data inside the TLS connection is encrypted and authenticated, which greatly complicates the attacker's goals of reading or modifying the data.



Principles of Computer Security, Fifth Edition

## Common Use Cases


- Low-Power Devices: mobile phones, laptops
- Low Latency: stream cipher
- High Resiliency: resume operations after disruption
- Supporting Confidentiality: cryptography
- Supporting Integrity: Message Authenticating Codes
- Supporting Obfuscation
- Supporting Authentication: verify identity
- Supporting Nonrepudiation: verify authenticity

Beware of Magic Crypto Fairy Dust

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Cryptographic services are used in such a wide variety of applications that it might be easier to compile a list of places where crypto is not used. Maybe your toaster. Unless it's a cylon.

Crypto is certainly a workhorse. It supports at least 5 of the CIA AAA goals. But, it is important that designers, builders, operators, and users not treat it as if it were a panacea for all problems. You cannot sprinkle magic crypto fairy dust on your system and then claim mission accomplished. That's not how this works. That's not how any of this works.



## Principles of Computer Security, Fifth Edition

### Attribution

- The course slides are based on slides developed by the textbook authors, W. "Art" Conklin, Ph.D. (University of Houston), and Greg White, Ph.D. (University of Texas at San Antonio).
- These slides remain the intellectual property protected by US Copyright law of the authors and the textbook company.
- There have been some changes to the slide deck.

"Cryptography is like investing in an armored car to carry money between a customer living in a cardboard box and a person doing business on a park bench"

- Gene Spafford

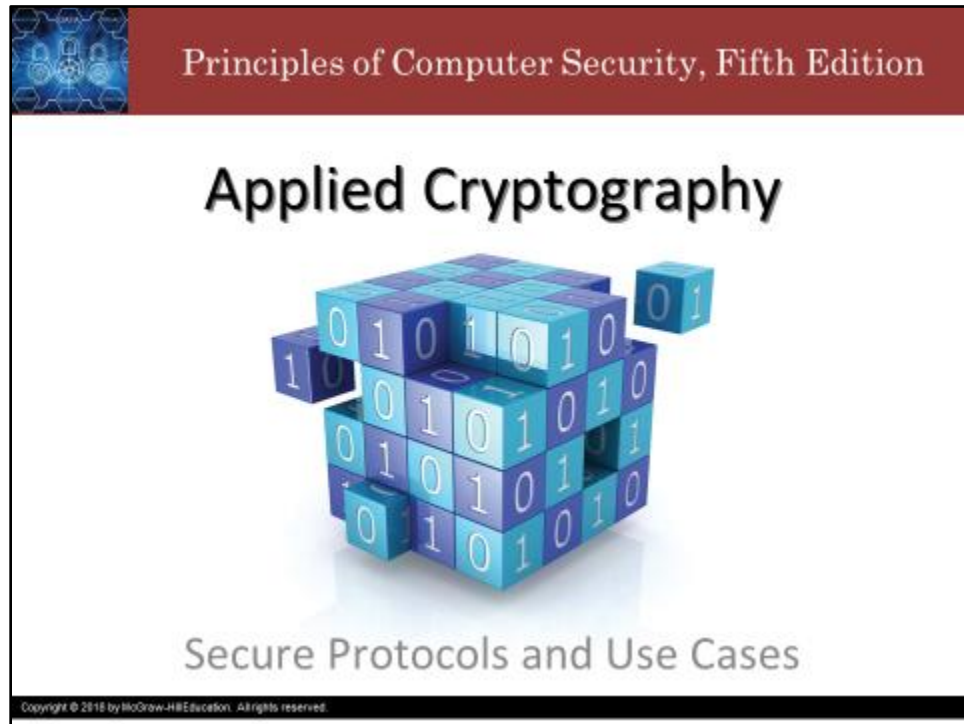
Copyright © 2018 by McGraw-Hill Education. All rights reserved.

I want to leave you with a quote attributed to Gene Spafford, legendary professor of computer science at Purdue University, speaking about the usage of cryptography: "Cryptography is like investing in an armored car to carry money between a customer living in a cardboard box and a person doing business on a park bench"

Thank you and take care.


## Applied Cryptography: Secure Protocols and Use Cases

Slide 1



Howdy! In this video, we discuss secure protocols and their use cases.





Principles of Computer Security, Fifth Edition

## Secure Protocols

- SSL, TLS
- HTTPS
- DNSSEC, DoT, DoH
- PGP, GPG
- S/MIME, IMAP, POP, SMTP
- SSH
- FTPS, SFTP
- LDAPS
- SRTP
- SNMPv3
- IPsec

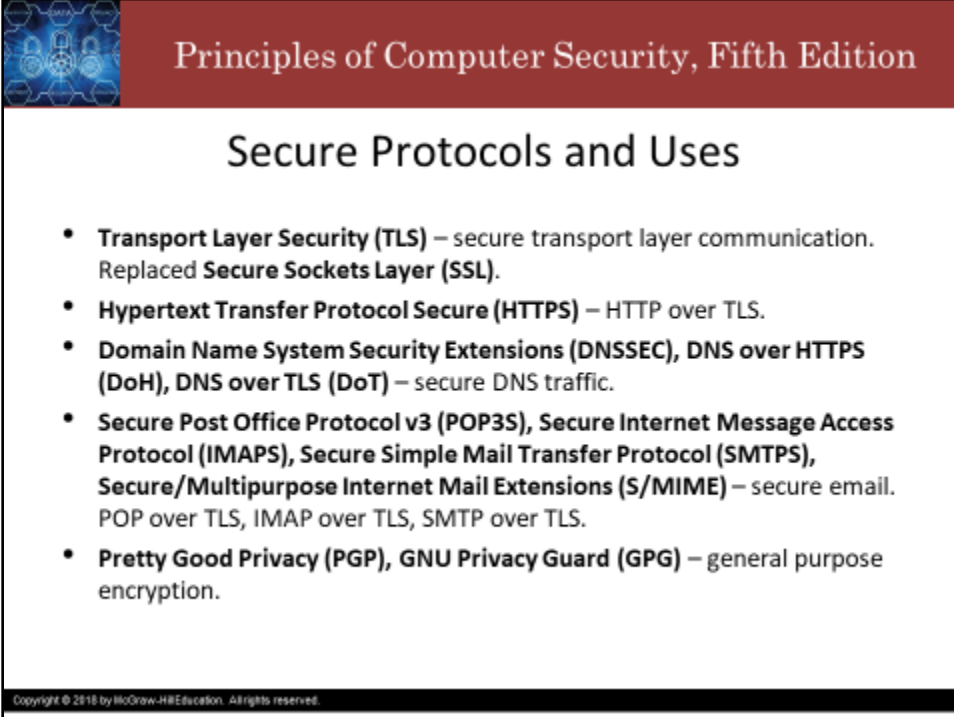
Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Protocols act as a common language, allowing different components to talk using a common set of functions. Many different protocols exist, all of which are used to achieve specific communication goals.

In the early days of networked computer systems and the Internet, when many of the basic protocols were first defined, security was not high on the list of concerns, if it was on the list at all. Most of the security was either physical or based almost entirely on trust. As the networks grew, the assumptions of trust fell away which necessitated the development of new and more secure protocols. By that time, however, some protocols were already so deeply embedded in the design and bones of the Internet, they could not be replaced with a secure protocol and instead the security had to be either bolted on at a higher level or had to be content being a bridesmaid until the bride dies (by which I mean, being on the sideline and available to dance, but not the center of attention... by which I mean the secure protocols were available to be used, but they could not be mandated for universal use due legacy systems that did not or could not implement them). The TCP and IP protocols are the main examples of this scenario. TCP and IP were not designed with security in mind, but they were enormously successful and are the de facto standards for transport and internet layer communications, respectively. Unfortunately, due to their design, they cannot be replaced by a secure version without breaking legacy system which rely on them. We basically need to redesign the Internet in order to make it secure below the application layer. So, instead, secure protocols are built on top of TCP, so their data which, is secured and authenticated, ride inside TCP packets, which are not. It's like wearing a bulletproof vest while driving your care through an active warzone. Your bodily integrity is assured, but your vehicle's not so much.

On this list are just a small number of the many protocols which have been developed to secure network communications of various types.

Slide 3



**Principles of Computer Security, Fifth Edition**

## Secure Protocols and Uses

- **Transport Layer Security (TLS)** – secure transport layer communication. Replaced **Secure Sockets Layer (SSL)**.
- **Hypertext Transfer Protocol Secure (HTTPS)** – HTTP over TLS.
- **Domain Name System Security Extensions (DNSSEC), DNS over HTTPS (DoH), DNS over TLS (DoT)** – secure DNS traffic.
- **Secure Post Office Protocol v3 (POP3S), Secure Internet Message Access Protocol (IMAPS), Secure Simple Mail Transfer Protocol (SMTPS), Secure/Multipurpose Internet Mail Extensions (S/MIME)** – secure email. POP over TLS, IMAP over TLS, SMTP over TLS.
- **Pretty Good Privacy (PGP), GNU Privacy Guard (GPG)** – general purpose encryption.

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

**Transport Layer Security (TLS)** is the successor of the now-deprecated **Secure Sockets Layer (SSL)**. It is an application layer cryptographic protocol designed to secure communications over a computer network at the transport layer. Several versions of the protocol are widely used in applications such as email, instant messaging, and voice over IP, but its use as the security layer in HTTPS is the most publicly visible.

The Hypertext Transfer Protocol (HTTP) is an application layer protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web, where hypertext documents include hyperlinks to other resources that the user can easily access, for example by a mouse click or by tapping the screen in a web browser. When you run HTTP over a TLS connection, you get **HTTP Secure (HTTPS)**. HTTPS is now more widely and frequently used than the original non-secure HTTP. For myself, if a site doesn't support HTTPS, I can't visit it. My browser won't let me because I configured it to use HTTPS everywhere and to block all unencrypted requests. The Domain Name System (DNS) is a hierarchical and decentralized naming system for computers, services, or other resources connected to the Internet or a private network. Its most prominent function is the translation of domain names to IP addresses so you can simply remember `tamu.edu` instead of `165.91.22.70`. DNS traffic is sent in plaintext and is vulnerable to spoofing.

The **Domain Name System Security Extensions (DNSSEC)** is a suite of extensions for securing data exchanged in the DNS. DNSSEC provides cryptographic authentication of data, authenticated denial of existence, and data integrity, but not availability or confidentiality. DNSSEC attempts to add security while maintaining backward compatibility. Other protocols for securing DNS are DNS over HTTPS and DNS over TLS. Firefox use DNS over HTTPS by default. Many public DNS servers, including Google's, now support DNS over TLS.

The Post Office Protocol (POP) is an application-layer Internet standard protocol used by e-mail clients to retrieve e-mail from a mail server. POP version 3 (POP3) is the version in common use. The Internet Message Access Protocol (IMAP) is an Internet standard protocol used by email clients to retrieve email messages from a mail server over a TCP/IP connection. Virtually all modern e-mail clients and servers, like Gmail and Outlook, support IMAP and POP3, which are the two most prevalent standard protocols for email retrieval. POP3 and IMAP can be run over TLS to cryptographically protect the data. In that mode, they are referred to as **Secure Post Office Protocol v3 (POP3S)** and **Secure Internet Message Access Protocol (IMAPS)**.

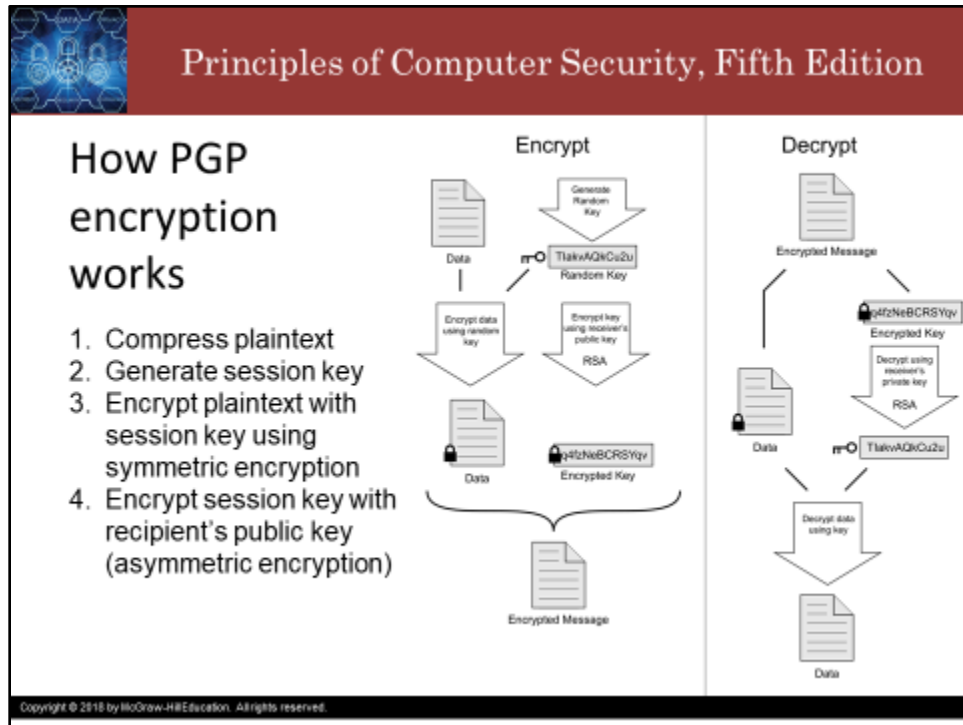
The Simple Mail Transfer Protocol (SMTP) is an internet standard communication protocol for electronic mail transmission. POP3 and IMAP are used by end users and their mail client. SMTP is the protocol used by the mail servers themselves and other message transfer agents use to send and receive mail messages. Like POP3 and IMAP, SMTP is not natively secure, but can be run over TLS to protect the data. It is worth noting that even email servers within the same logical network should use secure communication channels since there is a risk of an attacker being on the network and eavesdropping on mail transfers, which would bypass the security provided by running the client-side POP3 and IMAP protocols over a secure channel. Google got burned by the NSA on this one a few years ago. They assumed that data within their own network, going between their own servers, would be secure, but the NSA had penetrated Google's network and installed sniffers that sucked up the unencrypted data as it passed between Google servers and exfiltrated it to the NSA for analysis. They didn't need to break TLS, they just waited until Google dropped it's guard and decrypted the data and then YOINK! All your email are belong to NSA now. So, now Google encrypts everything, from end-to-end and at rest. Which is what they should have been doing all along. Remember to apply defense in depth and protect data throughout it's entire lifecycle.

Multipurpose Internet Mail Extensions (MIME) is an Internet standard that extends the format of email messages to support text in character sets other than ASCII, as well as attachments of audio, video, images, and application programs. In addition to email, HTTP also uses MIME formatting.

**S/MIME (Secure/MIME)** is a standard for public key encryption and signing of MIME data. S/MIME functionality is built into the majority of modern email software and interoperates between them.

**Pretty Good Privacy (PGP)** is an encryption program that provides cryptographic privacy and authentication for data communication. PGP is used for signing, encrypting, and decrypting texts, e-mails, files, directories, and whole disk partitions and to increase the security of e-mail communications.

**GNU Privacy Guard (GnuPG or GPG)** is a free-software replacement for the proprietary PGP cryptographic software suite.



PGP encryption is a 4-step process of compression, key-generation, symmetric encryption, and public-key encryption. Compression is used to decrease the size of the plaintext and remove patterns to make cryptanalysis (which can exploit those patterns) more difficult. The session key is generated using a key derivation function and a pseudorandom number generator (or some source of natural randomness, like mouse movements and keystrokes). The session key is ephemeral, it will only be used once. Once the plaintext is compressed and the session key is generated, the compressed data is encrypted using a symmetric key algorithm like AES. Then, the session key is encrypted using the recipient's public key so that only they can decrypt it. The ciphertext and the encrypted session key form the PGP-encrypted message which can then be emailed or transmitted in some other way to the recipient. Decryption goes in reverse: use your private key to decrypt the session key, use the session key to decrypt the ciphertext, decompress the data to obtain the plaintext message.

As a side note: TLS 1.3 removed support for compression due to the CRIME exploit, which stands for Compression Ratio Info-leak Made Easy, which uses a chosen plaintext attack to systematically infer the original plaintext through changes in the observed ciphertexts due to the compression of the chosen plaintexts.

Because it uses both symmetric and public-key crypto, PGP is a hybrid cryptosystem. The point is to get the best of both worlds: fast encryption for the largest part of the data by using symmetric encryption and authentication and authorization to decrypt by using public-key encryption. Each of the steps can use any of a number of supported algorithms, similar to TLS cipher suites, except that little prior negotiation is required, the PGP header contains the

information required to determine which algorithms were used. Following the principle of open design, knowing how PGP works and which algorithms were used does not weaken the security.

PGP can also be used to add a digital signature to the message. This adds integrity of origin so the recipient will be able to verify that the sender is they claim to be.


There's a neat story in the history of PGP that I quite enjoy. You can find it in the PGP article on Wikipedia under the section "Criminal Investigation".

Shortly after its release in 1991, PGP encryption was very popular and so, naturally, found its way outside the United States. In early 1993, PGP's creator, Phil Zimmermann, became the formal target of a criminal investigation by the US Government for exporting munitions without a license. You see, back then, any cryptosystem with a key length larger than 40 bits was considered a munition within the definition of US export regulations; Since PGP uses keys with at least 128 bits, it qualified under this regulation. If prosecuted and found guilty, Zimmermann would be facing up to one and a half million dollars in fines and up to ten years in prison.

Now, this is where it gets fun. Upon learning of the investigation, Zimmermann opted for an aggressive response. He published the entire source code of PGP in a hardback book which was widely distributed. Anybody, anywhere, who wanted their own copy of PGP could copy the code from the book, or scan the pages using OCR, to create a set of source code files. They could then use the freely available and not export controlled GNU Compiler Collection to compile the source code into executable machine code, the PGP program itself. Thus, PGP was available anywhere in the world. The principle of this approach was devilishly simple: export of munitions is restricted, but the export of books is protected by the First Amendment.

In 1996, the US Department of Justice closed the investigation of Zimmermann without filing criminal charges against him or anyone else. Around that same time, two other cases (not involving PGP) did go to trial where it was decided that cryptographic source code was protected speech under the first amendment. Due in part to Zimmermann's gamble and to the other cases, US export regulations regarding cryptography were substantially relaxed and modernized throughout the second half of the 1990s. PGP encryption no longer meets the definition of an export-controlled weapon, and can be exported internationally. More importantly, what that means is that I am allowed to teach PGP, and other cryptographic algorithms, to all of you, some of whom are not US citizens.

(img source: [https://en.wikipedia.org/wiki/Pretty\\_Good\\_Privacy#/media/File:PGP\\_diagram.svg](https://en.wikipedia.org/wiki/Pretty_Good_Privacy#/media/File:PGP_diagram.svg))



Principles of Computer Security, Fifth Edition

## Secure Protocols and Uses

- **Secure Shell Protocol (SSH)** – remote connections to a server
- **Secure Real-time Transport Protocol (SRTP)** – deliver audio and video over IP networks .
- **Secure Lightweight Directory Access Protocol (LDAPS)** – LDAP over TLS.
- **FTPS** – FTP over TLS.
- **SFTP** – SSH FTP (not FTP over SSH).
- **Simple Network Management Protocol version 3 (SNMPv3)** – manage devices on IP networks.
- **Internet Protocol Security (IPsec)** – secure IP packets.
  - IPsec transport mode – only payload is secured.
  - IPsec tunnel mode – entire packet is secured. Used for VPNs.

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

The **Secure Shell Protocol (SSH)** is a cryptographic network protocol for operating network services securely over an unsecured network. Typical applications include remote command-line, login, and remote command execution, but any network service can be secured with SSH.

The **Secure Real-time Transport Protocol (SRTP)** is a network protocol for securely delivering audio and video over IP networks. It has a sister protocol, SRTCP – Secure Realtime Transport Control Protocol. Together, they are like a secure version of TCP for real time applications.

The **Lightweight Directory Access Protocol (LDAP)** is an open, vendor-neutral, industry standard application protocol for accessing and maintaining distributed directory information services over an Internet Protocol (IP) network. Directory services play an important role in developing intranet and Internet applications by allowing the sharing of information about users, systems, networks, services, and applications throughout the network. LDAP is insecure, but can be secured using SSL or TLS. LDAPS is commonly known as LDAP over SSL. Now that SSL is deprecated, you could secure LDAP by running it over a TLS connection. This protocol, LDAP over TLS is non-standard but supported by most LDAP servers.

The **File Transfer Protocol (FTP)** is a standard communication protocol used for the transfer of computer files from a server to a client on a computer network.

**FTPS** is the implementation of FTP over a TLS connection.

The **SSH File Transfer Protocol (or Secure FTP, SFTP)** is a network protocol that provides file access, file transfer, and file management over any reliable data stream. It is not FTP over SSH, but is actually its own protocol which runs over SSH or TLS (or any secure channel).

The **Simple Network Management Protocol (SNMP)** is an Internet Standard protocol for collecting and organizing information about managed devices on IP networks and for modifying that information to change device behavior. Devices that typically support SNMP include cable modems, routers, switches, servers, workstations, printers, and more. Versions 1 and 2 of SNMP do not provide security, but version 3 does.


**Internet Protocol Security (IPsec)** is a secure network protocol suite that authenticates and encrypts packets of data to provide secure communication between two hosts over an IP network.

IPSec is an Internet-layer end-to-end security scheme. Other secure protocols operate above the Internet layer, such as TLS which runs at the Transport Layer (actually its application layer, but other application layer programs use it as if it was a transport-layer protocol) and SSH which runs at the Application layer.

IPSec can be run in either a host-to-host transport mode, or in a network tunneling mode.

In transport mode, only the payload of the IP packet is encrypted or authenticated. The routing information (the IP addresses) are left in plaintext, but cannot be modified if authentication is used since any change to the header will invalidate the hash and only the end points have the key required to compute the hash correctly.

In tunnel mode, the entire IP packet is encrypted and authenticated. It is then encapsulated into a new IP packet with a new IP header. Tunnel mode is used to create VPNs. Unlike transport mode, tunnel mode allows routers to change the IP addresses of the encapsulating packet since the authenticated data (the entire IPSec packet) is carried in the payload.



## Principles of Computer Security, Fifth Edition

### Attribution

- The course slides are based on slides developed by the textbook authors, W. "Art" Conklin, Ph.D. (University of Houston), and Greg White, Ph.D. (University of Texas at San Antonio).
- These slides remain the intellectual property protected by US Copyright law of the authors and the textbook company.
- There have been some changes to the slide deck.

The Illustrated TLS Connection: <https://tls13.ulfheim.net/>

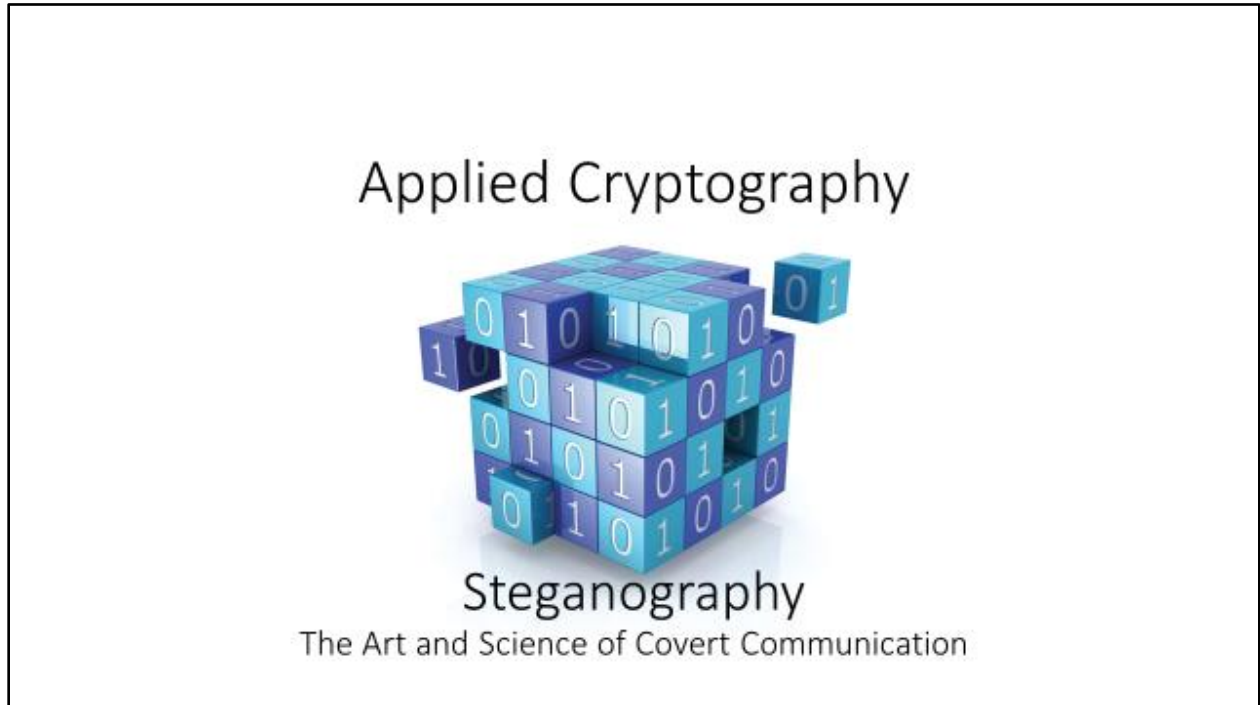
Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Thank you, and take care!



## Applied Cryptography: Steganography

Slide 1



Howdy! In this video, we discuss steganography.

My PhD dissertation was on steganography, and I think it's a really fun field, I think it's more fun than crypto, actually.

## The Prisoners' Problem

- Alice and Bob are imprisoned under watchful warden Walter
  - Allowed to communicate indirectly through the warden
  - Warden must be able to read all correspondence
  - At any sign of trickery, communication channel is cut
    - i.e. cryptography is prohibited
- Alice and Bob need to communicate privately
  - Plan their escape, get their stories straight, share disharmonizing information
- How can they do it?

**Steganography!**

The model we work with is known as the prisoner's problem.

Alice and Bob are imprisoned under the watchful eye of Warden Walter.

They can talk to each other, but all communication goes through the warden who can read everything.

At the first sign, the first whiff, of something fishy, the warden will cut their communication. This means no crypto unless the warden also has the key.

Nonetheless, Alice and Bob need to communicate privately.

How can they do it?

The answer, of course, is steganography!

## Steganography: Art and Science

- Literally: covered writing
  - στεγανός (*steganos*) – covered, concealed, protected
  - γράφειν (*graphein*) – writing
- Cryptography protects *what* Alice sends to Bob
  - Obfuscate the contents of the message to make it unreadable
- Steganography protects *when* and *if* Alice sends data to Bob
  - Disguise the message to make it undetectable or untraceable
- Techniques both Physical and Digital
  - Disguise information as innocuous channel-usages

The word steganography comes from the Greek *steganos*, meaning covered, and *graphein*, meaning writing. So, literally, covered writing.

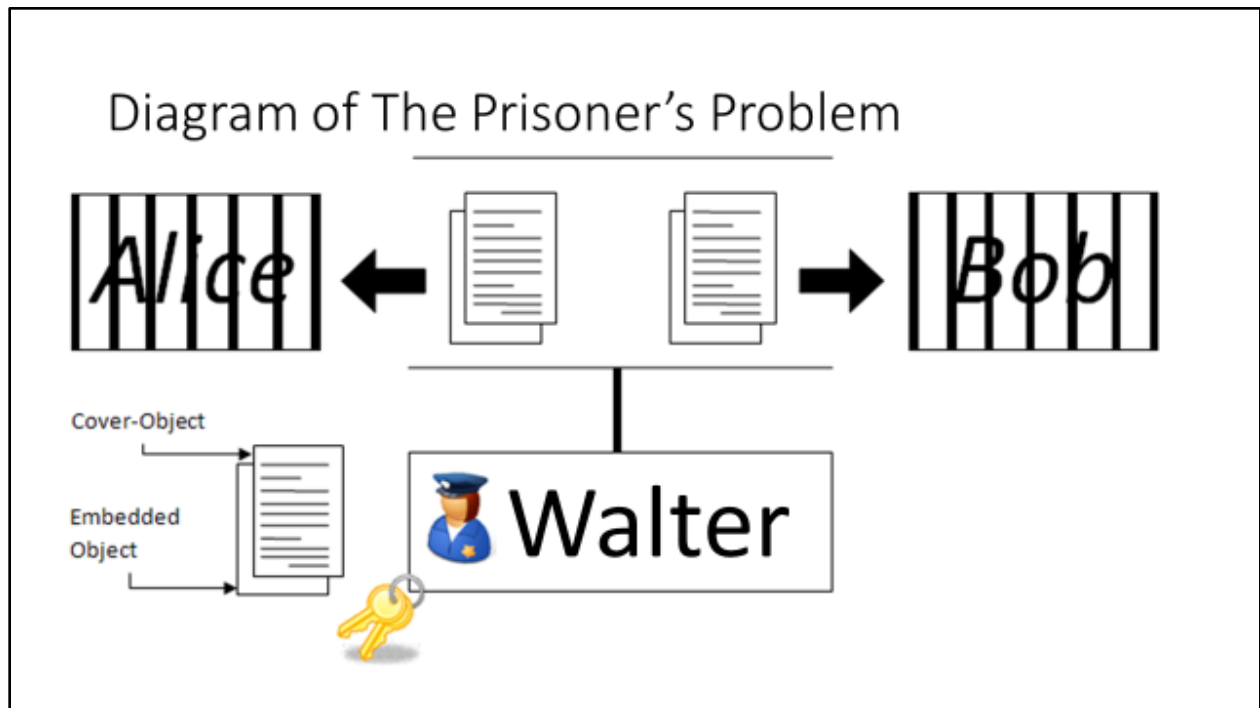
Whereas cryptograph protects what Alice sends to Bob,

Steganography protect when and if Alice sends anything to Bob

Like Crypto, Stego techniques are millennia old and can be both physical and digital.

The main ides is to disguise communications in such a way that a 3<sup>rd</sup> party observer is unaware and unsuspecting that any communication is occurring at all, other than what is overt and clear.

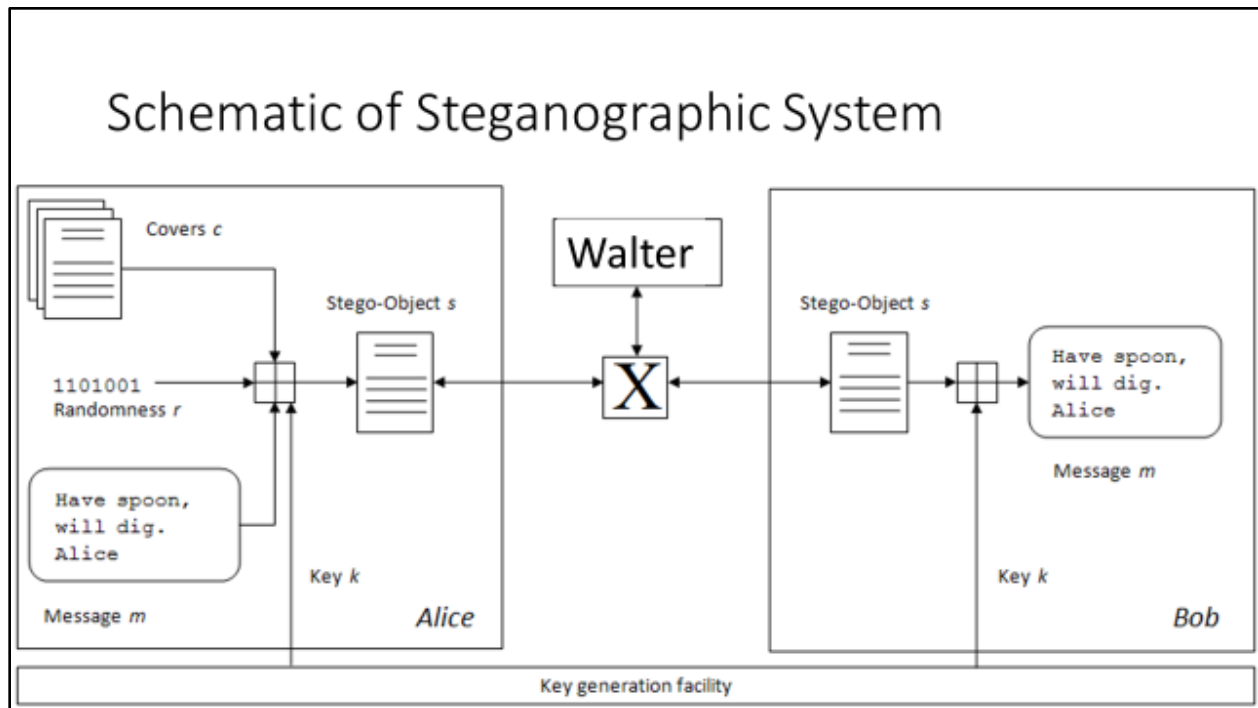
There is both an art and a science to this.



This is a diagram of the prisoner's problem.

We can see Alice and Bob in jail, and the warden, ever watchful in the middle.

Alice needs to send a message to Bob that Walter will see and believe is exactly what it seems but is really a cover for secret communication.



This is a schematic showing the components involved in sending a steganographically hidden message to Bob.

Alice and Bob share a key (or they don't and Alice can use Bob's public key, whatever, it's 2021, public-key stego should be a thing by now).

Alice takes her message, combines it with some randomness and a set of cover objects to generate, in some way, a steganographic object (analogous to a ciphertext), which she sends to the warden to examine and forward to Bob.

The warden, for now, is assumed to only engage in inspection, and no modification. If the warden's inspection does not yield sufficient evidence of sneaky behavior on the part of Alice, then they will forward the message to Bob, believing the message to be only what it seems to be: some innocuous object or message.

Bob takes the object, applies the inverse steganographic operation with his key to recover the hidden message.

So, how can they do it?

What method can they actually use to pull this off?

## A Few Examples of Steganography

- Wax tablets
- Scalp tattoo
- Bacon's cipher
- Grille ciphers
- Invisible ink
- Microdots
- Blinked Morse code
- **LSB manipulation**
- **GIF + Zip concatenation**
- Polymorphic code
- Packet delays
- Ordering of lists
- Extra spaces in documents
- Zero-width characters
- CPU load/temp
- File locks
- **Games**
- And SO MUCH MORE!

It turns out Alice and Bob are somewhat spoiled for choice. Throughout history, and especially in the modern age of computers, there are many examples of successful steganographic systems.

Some of the earliest examples, like writing on the substrate of a wax tablet and then replacing the wax and writing some innocuous cover message, or shaving the messenger's head, tattooing the message on their scalp, waiting until the hair grows back, and sending them on their way, are quite literal in their interpretation of the genre: covering the hidden message so it cannot be seen. As steganography developed, the methods upped their hide and seek game considerably.

I am going to teach you the basics of 3 modern steganographic techniques.

## The Fundamental Theorem of Steganography

Where there is *choice*,  
there is *information*.

Before we jump into the concrete examples, I want to put an idea into your mind and ask that you keep it there throughout this presentation.

I call this the fundamental theorem of steganography. Although, I think it needs a qualifier, like “modern” in front the steganography, since this applies to steganography which is not so orthodox in it’s adherence to the original roots of the name. Literally hiding things is a bit like security by obscurity, which is not secure on its own. I think steganography has grown past that stage now.

Anyways, my fundamental theorem of modern steganography states that where there is choice, there is information.

This is simply an adaption of Claude Shannon’s information theory.

But, for me, when I realized this, it really crystalized my ideas about the development of steganography and where it is headed.

## Categories of Steganography

- **Steganography by modification**
  - Substitution systems
  - Transform domain techniques
  - Spread spectrum techniques
  - Statistical methods
  - Distortion techniques
- **Steganography by cover synthesis (“synthetic steganography”)**
  - Generate a cover object to modify, never release the original
  - Generate a stego-object such that the construction of the object encodes the information

Steganography can be divided in several categories, but I like to make the first split at a high level between methods which work by modifying objects and methods which work by creating objects.

Classical stegosystems use modification. They take existing objects and they modify them, typically by substitution. It's easy to detect the use of stego in these methods if you have access to the original object, or even another version of the object.

The kind of stegosystems in which I am most interested work differently. They generate a new object to hide the secret data. So, we can still use the classical techniques of modification, but we don't have to worry about comparison to an original or any other version. Synthetic cover objects are ephemeral. But, we can also create the object in such a way that the object itself encodes the secret data. So, we release the one and only object. There is no original. There will be no other versions in any meaningful sense. The composition of the object, or it's construction, is a function of the hidden data.

Do you remember the fundamental theorem? Where there is choice, there is information.



## Concatenate GIF and Zip

- GIF files store metadata in the header
- Zip files store metadata in the footer



- Concatenate a GIF with a Zip to make a file which is both a GIF and a Zip.
  - Windows: `copy /b cover.gif+secret.zip new_cover.gif`
  - Unix: `cat secret.zip >> cover.gif`
  - File acts like GIF by default, but can be opened by a Zip-reader (e.g. 7-Zip) to reveal the secret payload

My first example of stego for you is to concatenate a GIF and ZIP.

GIF is an image format which stores its metadata in a header, at the beginning of the file

ZIP is a compression format which stores its metadata in a footer, at the end of the file.

This means we can smoosh a GIF and a ZIP together into a single file and a GIF-reading program can still read the file as a GIF and a ZIP-reading program can read the file as a ZIP.

Due to the way operating systems guess which program to use to open a file, the file will act like a GIF by default, so the hidden data is the ZIP archive appended at the end.

This technique is easy to detect because the file size on disk will not match the file size in the GIF header. Also, it's a file which has a GIF header and a ZIP footer. It ain't foolin' nobody who knows what to look for.

## LSB Manipulation



- A digital image is made up of picture elements (*pixels*)
- Each pixel contains information about the color of that tiny region of the image
  - Grayscale: 8 bits give 256 shades of gray
  - RGB: 24 bits give 256 shades of each of Red, Green, and Blue
- The low order bits of each pixel control color variation which is beneath the perceptual threshold of the human visual system
  - E.g. 150-156 are imperceptibly different shades of the same color
- Thus, low order bits can be modified without any perceivable degradation of image quality
- Overwrite the low order bits with bits of the secret messages

The next example is the paradigmatic example given when anybody talks about steganography: manipulating the low order bits of an image. This technique is also applicable to other media files, like audio and video.

The idea is to replace the low order bits, even just the lowest bit in the 1s place, with the message data.

Because the human visual perception system is not able to distinguish between shades which are too close together, such small changes do not visually affect the image.

This idea is inspired by the classical stego game of hide and seek. It just hides the data in the low-order bits and hopes no one will think to look there.

Unfortunately, while these modifications may be imperceptible to a human, a computer sees the differences plain as day. In fact, it's worse than that, since there are now steganalysis methods which can detect changes to an image even at the level of 1 change per 100 bits by using interpixel relationships.

But, the idea is simple and the effect is impressive... to a human.

## LSB Manipulation

```
def embed(image, secret):
    height, width = len(image), len(image[0])
    index = 0
    for row in range(height):
        for col in range(width):
            if index >= len(secret): return
            image[row][col] -= image[row][col] % 2
            image[row][col] += secret[index]
            index += 1
```

Here is Python-ish code for an algorithm to embed some secret bitstring in an image (which is stored as a 2-dimensional array of greyscale pixel values).

As you can see, the work is done in the loops. Every pixel has its least significant bit obliterated and replaced by a bit of message until the entire message is embedded, at which point no other modifications are made.

One average, given a compressed or encrypted secret, which we expect would have uniformly random and uncorrelated bits, and a greyscale image, which we hope also has random and uncorrelated least significant bits due to noise in the camera or natural variations in color, we should expect that about half of the bits get flipped.

## LSB Manipulation

```
def extract(image):  
    height, width = len(image), len(image[0])  
    secret = list()  
    for row in range(height):  
        for col in range(width):  
            secret.append(image[row][col] % 2)  
    return secret
```

This is the corresponding extraction algorithm.

It traverse the image in the same order as the embedding algorithm and simply peels off the least significant bit of every pixel to recover the hidden data.

## LSB Manipulation

- Cover Image: lena.bmp
  - 8-bit grayscale
  - 512x512 pixels
- Capacity:
  - 1 bit per pixel:
    - 262,144 bits of secret
    - 32,768 bytes of secret
  - 2 bits per pixel
    - 524,288 bits of secret
    - 65,536 bytes of secret



Let's see it in practice.

This is a greyscale picture of Lena (pronounced Lenna). This is one of the most used images in computer history.

We're going to hide some data in it.

If we use 1 bit of every pixel, we can hide 32 kilobytes of data, which is not insignificant.



Can you tell the difference?

The real test is: can you determine whether the image on the previous slide is the cover or the stego image?

## Quiz!

What is the capacity of a 12 megapixel RGB image?

- 1 megapixel is 1 million pixels
- Each 24-bit RGB pixel is three 8-bit pixels for the Red, Green, and Blue components
- Assume the 2 lowest order bits of each color are beneath the perceptual threshold
- 8 bits per byte, 1000 bytes per kilobyte, 1000 kilobytes per megabyte

**1.5 - 9 megabytes**

Pop quiz time.

What is the capacity of a 12 megapixel RGB image?

Pause the video and work out your answer. When you're ready to check your work, hit play.

If you use only 1 bit from each pixel, you can hide 1.5 megabytes. If you use 2 bits of every color of each pixel, you can hide 9 megabytes.

## Synthetic Steganography

- Modification-based steganography has a **serious drawback**
  - Two versions → at least one of them has been modified
  - Original is known → every modification can be detected
- Synthetic steganography **avoids this problem** in one of two ways:
  - Generate a novel cover object to modify exactly once and never release the original
    - Possible that modification can still be detected even without access to other versions
  - Generate an object which **encodes the secret** in the way the object was constructed
    - Requires the generation process to be reversible so that objects can be deconstructed

Modification-based steganography has a serious drawback.

If two different versions of the same object are discovered, then we know that at least one of them has been modified.

If the original is known, then every modification can be detected and sometimes the secret can even be extracted.

The important thing for steganography is detection. The secret itself can be secured with cryptography, so even if the hidden data can be extracted, the content is still protected. But, the mere fact that there was something hidden at all is enough for the warden to bring down the hammer on Alice and Bob. Synthetic steganography attempts to avoid this vulnerability to comparison in one of two ways:

We can generate a novel cover object to modify exactly once and never release the original.

It is possible, however, that the modification can still be detected even without access to other versions.

Or, we can generate an object which encodes the secret in the way the object was constructed.

This approach requires that the generation process is reversible so that objects can be deconstructed. Synthetic steganography using encoding is similar to modern cryptographic techniques in that the output is a function of the input. The key difference is that a stegotext has to make sense in some way – it has to look like or act like the cover media it is impersonating –, whereas a ciphertext does not have this requirement.



## Tic-Tac-Stego: Covert Channels in Combinatorial Games

- Combinatorial games:
  - 2 players
  - Perfect information
  - No chance
  - Examples: Tic-tac-toe, Checkers, Chess, Go, Dots-and-boxes,...
- Attractive for synthetic steganography
  - The stego-object is the sequence of moves, which is built up by both players over the course of the game
  - The channel is interactive. Warden must allow Alice's moves to reach Bob (and vice versa) or else the game cannot be played
  - Generating reasonable next moves is relatively easy
  - The context of the game provides a plausible explanation for the choice of action

<https://dl.acm.org/doi/10.1145/2510126>

This brings us to my last example for you, something I call Tic-Tac-Stego, which is steganography in games.

In particular, the kind of games I like to use are called combinatorial games.

2 players, no hidden information, no element of chance, games like tic-tac-toe, chess, go, and many others.

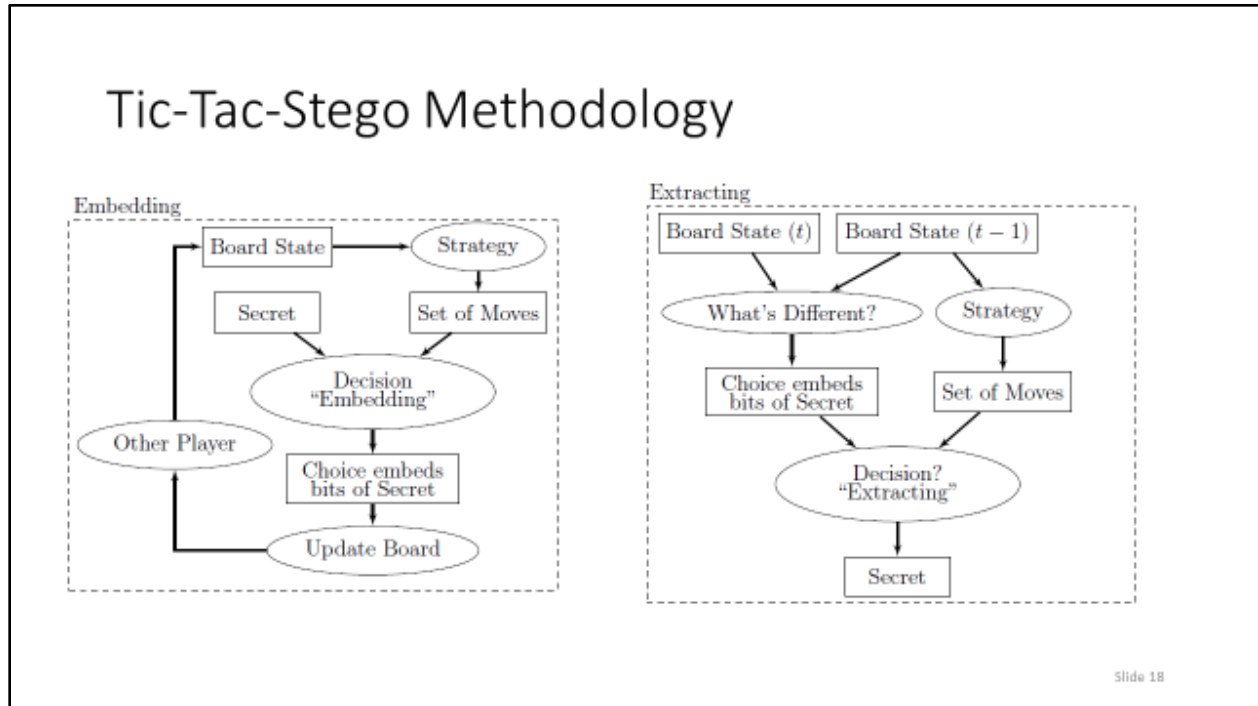
What makes this class of games attractive as a target for synthetic steganography

is that there is a logical structure to the game, with rules that must be followed, and the players make decisions within that structure.

Also, the game is constructed over a sequence of moves, and therefore cannot be built at all unless those moves are communicated between the players.

It is not hard to generate all possible next moves, and it may even be straightforward to select a set of reasonable or good moves from the set of all possible moves. And, since there are two players and each must make a choice at every step, the channel runs in both directions.

And, finally, the game itself is the reason for the communication. Alice has plausible deniability that her move means more than what it is. She's just playing a game with Bob, that's all. What's so suspicious about that!?



This figure describes how this stegosystem works.

Alice has a secret that she wants to embed in a game with Bob.

She starts with the board in its initial state, if she has the first move, or with Bob's first move on the board if Bob had the first move.

Then she applies some strategy to select a set of reasonable or good moves from the set of all possible moves.

Then she takes some bits of her secret data that she wants to send to Bob and she uses those bits to make a decision as to which move to select in some deterministic and reversible way.

That choice embeds those bits into the game.

She can update the board with her move and send it to Bob to make his next move, or she can simply send Bob her move for him to update his own local copy of the board, like playing Chess by mail.

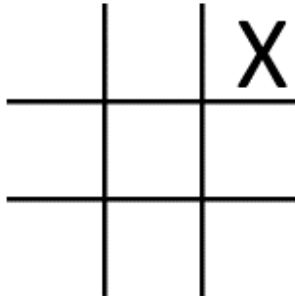
On the receiving end, Bob can extract the bits of Alice's secret by determining which move she made (which is easy if she would just send him the move directly instead of sending him a whole new board state), and then applying the strategy which he knows Alice applies (the strategy does not need to be secret, even the warden can know it). Then, given the set of moves from which Alice's was drawn, Bob determines the bits which Alice's move encodes, in some deterministic way (the inverse of the operation Alice used to go from bits to move). Bob can do this extraction online, at every step of the live game, or he can apply the process to a game record, a sequence of moves in some database. In this latter case, Alice could have generated the whole game herself and posted it for Bob to find later.

The warden gets to examine the board state at every step of the game, same as Bob, and must decide at each step if the game seems authentic, or if there is something fishy going on. A real-world example of this is when people cheat in online games. Veteran players can sense that something is not right when they play against a cheater. Something about the moves their opponent makes clues them into the possibility that their opponent is not really the one making the moves, that some other process is at play.

## Slide 19

### Example: Tic-tac-toe

- In her first move, Alice has 9 options
  - Label each with a bit string
    - Top-left = 000
    - Top-middle = 001
    - **Top-right = 010**
    - ...
    - Bottom-left = 110
    - Bottom-middle = 1110
    - Bottom-right = 1111
  - Pick the move corresponding to the next bits of the secret
    - Secret: "R" = 01010010
    - Pick top-right cell



Slide 19

Let's now work through an example of this method applied to the game of Tic-Tac-Toe.

At each step, the players choose an open cell to place their mark. In this game, Alice will play X and Bob will play O.

In her first move, Alice has 9 options. Let's assume that Alice applies the null strategy and so will choose her move from the set of all legal moves.

We label each move with a bit string, counting up from 000 at the top left, to 110 at the bottom left.

The last 2 cells, numbers 8 and 9, actually get 4-bit codes: 1110 and 1111.

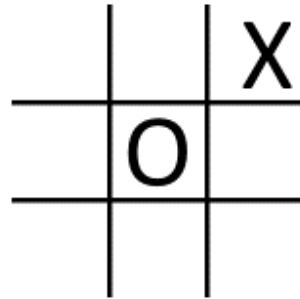
Alice's secret is the letter 'R', which has binary representation 01010010.

The first three bits of the secret are 010, which correspond to the top right cell.

So Alice marks her X in the top right cell.

## Example: Tic-tac-toe

- Bob knows that Alice had 9 options
  - Can assign the same labels
  - Can see which choice Alice made
  - Can extract the bits
    - 010
- Bob has 8 options
  - Depending on strategy, can send up to 3 bits
  - Suppose his strategy is to play optimally
    - Required to play in the middle → 0 bits of capacity



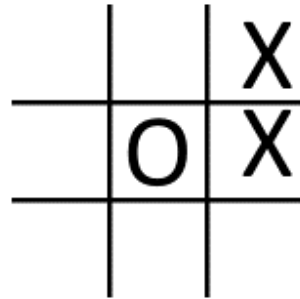
Slide 20

Bob sees Alice's move and he knows she chose from 9 options. He can assign the same bitstrings to each move that Alice did and see that Alice's move corresponds to the bits 010. So, he writes those down and makes his move.

Let's suppose that Bob's strategy is to play optimally. Given that Alice put her move in the corner, Bob's only move, out of the 8 legal moves, is to place his mark in the center. Since Bob had only 1 move, his choice encodes no bits.

## Example: Tic-tac-toe

- Alice knows Bob's strategy
  - Knows he only had 1 option → receives no bits
- Alice has 7 options and needs to send 10010
  - TL = 00
  - TM = 010
  - ML = 011
  - **MR = 100**
  - BL = 101
  - BM = 110
  - BR = 111



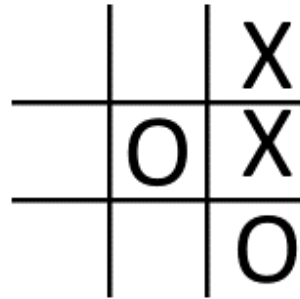
Back to Alice. She sees Bob's move, knows his strategy. Knows he had no choice but to play in the center and so knows that Bob's move encodes no bits.

For her next move, Alice has 7 options and has 5 bits left to send.

She can send 3 of them with the move on the middle right of the board, which encodes the bits 100.

## Example: Tic-tac-toe

- Bob knows that Alice had 7 options
  - Can assign the same labels
  - Can see which choice Alice made
  - Can extract the next bits
    - 010 100
- Bob's strategy requires him to block Alice's win
  - Plays in bottom-right
  - Sends no bits

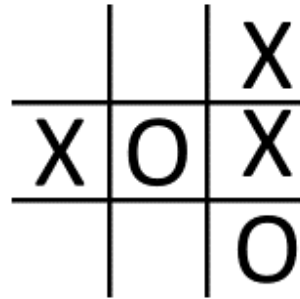


Again, Bob sees the move Alice made, labels the moves in the same way as Alice did and sees that Alice's move corresponds to the bits 100. So he writes those down next to the bits from the previous move.

And again, Bob has only one good move: he must block.

## Example: Tic-tac-toe

- Alice knows Bob's strategy
  - Knows he only had 1 option → receives no bits
- Alice has 5 options and needs to send 10
  - TL = 00
  - TM = 01
  - **ML = 10**
  - BL = 110
  - BM = 111



Alice knows Bob had only 1 move to play, so gets no bits.

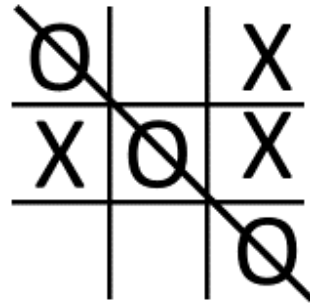
Alice now has 5 options and 2 more bits to send.

Her last two bits, 1 0, correspond to the move middle left.

This is not a good move, but she must play it because it's not about winning, it's about sending a message.

## Example: Tic-tac-toe

- Bob knows that Alice had 5 options
  - Can assign the same labels
  - Can see which choice Alice made
  - Can extract the next bits
    - 010 100 10 = R
- Bob can force a win by playing anywhere
  - 4 options → 2 bits
  - For sake of brevity, Bob will play in top-left (00)
  - Bob could play somewhere else to give Alice some more time to send bits



Slide 24

Bob sees Alice's move and maps it back to the bits, getting 10. Bob has 8 bits, and he can see now that these 8 bits are the ASCII code for the letter R.

Bob's next move is winning no matter where he goes. So, according to his strategy of playing optimally, he has 4 options, which means he can send 2 bits.

Let's just say Bob wants to send 0s, so he picks the first move, the top left, which is an immediate win.

When Alice sees that she lost, she'll be consoled by the fact that she sent 8 bits to Bob and Bob sent 2 bits to her, a total of 10 bits transmitted. If Alice has more bits to send, she can start a new game.



## Quiz!

Suppose Alice and Bob are both using the Null gameplay strategy to covertly communicate through the following game of Tic-tac-toe:

		x	x	x   x	o   x   x	o   x   x	o   x   x	o   x   x
x	x   o	x   o	x   o	x   o	x   o	x   o	x   o   o	x   o   o
			o	o	o	o   x	o   x	o   x   x

What are the secret bits sent from Alice to Bob? From Bob to Alice?

**A: 011 01 01 11 "k"1**

**B: 011 11 00 0 "x"**

I have another quiz for you.

Suppose Alice and Bob are both using the null gameplay strategy (which means no filtering of moves, every legal move is an option) to covertly communicate through the game of Tic-tac-toe shown on the slide

Alice playing X, Bob is playing O, Alice goes first.

What are the secret bits sent from Alice to Bob? From Bob to Alice?

Pause the video now and see if you can extract the bits from the game.

Alice sent 9 bits: 011 01 01 11, which is ASCII k and an extra 1

Bob sent 8 bits: 011 11 00 0, which is ASCII x.

That's a total of 17 bits.

And, if you look at the quality of gameplay, Alice and Bob actually played fairly well. At least, no missed a win or failed to block a win.

## Capacity of Tic-Tac-Stego

- In general, **steganographic capacity** is the number of bits of secret that can be hidden in the cover object
  - Often expressed as an **embedding rate**
  - The capacity of an image is typically expressed in bits per pixel
  - **Embedding density** is the ratio of the size of the payload to the size of the object
  - LSB image-stego at 1 bpp has an embedding density of 0.125, or 1 bit of secret per 8 bits of cover
- What is the steganographic capacity of **gameplay**?
  - 1 game of Tic-tac-toe was able to transmit 17 bits of secret in 9 moves
    - That's 1.8889 bits of secret per move
  - It takes less than 19 bits to store an entire game of Tic-tac-toe
    - That gives an embedding density of approximately 0.90, or 9 bits of secret per 10 bits of cover

In general, steganographic capacity is the number of bits of secret that can be hidden in the cover object.

It is often expressed not as an absolute number, but as an embedding rate so that it generalizes to object of different sizes.

The capacity of an image is typically expressed in bits per pixel.

The embedding density is the ratio of the size of the payload to the size of the object.

So, for example, LSB image-stego at 1 bpp hides 1 bit of secret in every 8 bits of cover and therefore has an embedding density of 0.125.

In the case of a combinatorial game, this analysis shows that games can have very high embedding rates.

One game of Tic-tac-toe was able to transmit 17 bits of secret in 9 moves, which is almost 1.9 bits of secret per move.

An entire game of Tic-tac-toe can be transmitted or stored in less than 19 bits.

That gives an embedding density of approximately 0.90, or 9 bits of secret per 10 bits of cover. 90% of the space is both game and hidden data.

## Summary

- **Steganography**: art and science of secret communication.
- Steganography and Cryptography are **complementary**.
- **Two kinds** of steganographic systems:
  - Modification-based: modify an existing object
  - Synthetic: create a new object from scratch
- **Any digital object** can be used for covert communication.
  - Image, audio, video, text, files, games, etc.
  - The fundamental theorem: **where there is choice, there is information**.
- **Steganographic security**: how difficult it is to detect hidden data.
- **Steganographic capacity**: how many bits of secret can be hidden.
- Steganography is **fun!**

To wrap up, let's review some conclusions.

Steganography is the art and science of secret communication.

It is complementary to Cryptography: they can be used separately or in conjunction.

There are two kinds of steganographic systems: modification-based stego which modifies an existing object, and synthetic, which creates a new object from scratch.

Any digital object can be used for covert communication, including images, audio, video, text, files, games, et cetera.

This follows from the fundamental theorem of steganography which states that "where there is choice, there is information."

Steganographic security depends on the difficulty for the attacker to detect that the object contains hidden information. There is actually more to it than that, including the integrity of the data.

Steganographic capacity measures how many bits of secret can be hidden and is typically reported as a rate or a density.

And, finally, I hope you will agree that steganography is fun!

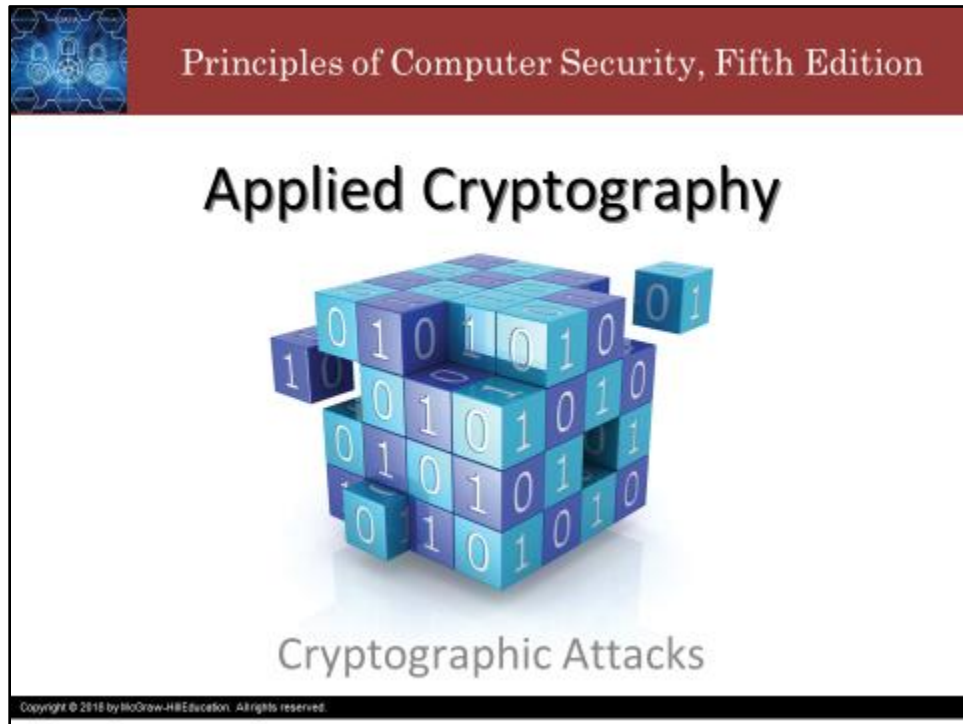
If you have questions or want to learn more, I'm always happy to talk about steganography.

Thank you and take care!




## Applied Cryptography: Cryptographic Attacks

Slide 1



Howdy! In this video, we introduce a few types of attacks on cryptographic algorithms.



Principles of Computer Security, Fifth Edition

## Information Available to the Attacker

- Ciphertext-only
- Known-plaintext – ciphertexts with corresponding plaintexts
- Chosen-plaintext/ciphertext – have one, can get the other for a set of texts
- Adaptive chosen-plaintext/ciphertext – have one, can get the other for a sequence of texts
- Related-key – chosen-plaintext with 2 unknown but related keys

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Attacks can be classified based on what type of information the attacker has available.

We typically assume, by the principle of open design or Shannon's maxim, or Kerckhoff's principle, that the attacker knows the algorithm but does not know the key.

In a ciphertext-only attack, the attacker only has access to the ciphertext. This would be the case, for example, with ciphertext which is captured by eavesdropping.


In a known plaintext attack, the attacker has a set of ciphertexts for which they know the corresponding plaintext. This would be the case for plaintext/ciphertext pairs which are captured after they are decrypted. Linear cryptanalysis is one example.

In a chosen plaintext attack, the attacker can pick a set of plaintexts to be encrypted with an unknown key to obtain the corresponding ciphertexts. A chosen ciphertext attack is the same, except the attacker picks ciphertexts and gets the plaintext back. This is the case when the attacker has access to an encryption or decryption oracle, some service which performs the cryptographic task, but whose internal structure is unknown, like a webserver that has a stored key and will attempt to decrypt anything you send it and will send you the results. This sounds ridiculous, but real-world examples of this attack abound. Differential cryptanalysis is usually a chosen plaintext attack.

An adaptive chosen plaintext or ciphertext attack is the same as the non-adaptive version, but the attacker has the chance to interact with the oracle and choose their next submission based on what they have learned from previous submissions.

A related key attack is like a chosen plaintext attack except that the attacker gets the ciphertext under two different keys. The keys are unknown, but a relationship between them is known, such as they differ in one bit, or they are inverse of each other. The WEP wireless security algorithm failed due to a related key attack because part of the key was known to be the same for all packets.

Slide 3



## Principles of Computer Security, Fifth Edition

### Birthday Attack

- Paradox: with 23 people, >50% probability that some pair have the same birthday.
- Attack: find two plaintexts whose hash values match on  $n$  bits in about  $1.25\sqrt{2^n}$  plaintexts.
- Chosen-plaintext or ciphertext-only

```

'0': b6595fc6ab0dc82cf12099d1c2d40ab994e8410c
'1': 356a192b7913b04c64574d18c28d46e6395428ab
'2': da4b9237bacc0df19c0760cab7aec4a8359010b0
'3': 77de68daecd823babb58ed1c8e14d710e83bb
'4': 1b6453892473a467d07372d45eb05abc2031647a
'5': ac3479d69a3c81fa62e6025c3696165a4e5e6ac4
'6': c1df96ee8cc2b62785275bca39ac26125e278
'7': 902ba3cdal883801594b6e1b452790cc53948fda
'8': fe5dbbcea5ce7e2988b8c69bcfdffe8904abc1f
'9': 0ade70c2cf57f75d0099765f4d720d1fa6c19f4897
'10': b1d5781111d84f7b3fe45a0852e59758cd7a87e5
'11': 17ba0791499db908433b80f37c65bc89b870084b
'12': 7b52009b64fd0a2a45e6d8a939753077792b0554
'13': bd307a3ec329e10a20ff8fb87480823dal14f8f4
'14': fa35e192121eabf3dabf9f5eae6abdcbcb07ac3b
'15': f1abd670358e036c31296e6b3b6c382ac00912
'16': 1574b4db75c78a6fd2251d61e2993b5146201319
'17': 0716d9708d321fba6a00818614779c779925365c
'18': 9e6a55b6b4563e652a23be9d623ca5055c356940
'19': b3f0c7f6bb763af1be91d5e74eabfcb199dc1f1f
--
'27': bc33e4e26e5e1af1408321416956113a4658763
'28': 0a57cb53ba59c46fc4b692527a38a87c78d84028
--
'45': fb644351560d8296fe6da332236b1f8d61b2828a

```

Copyright © 2015 by McGraw-Hill Education. All rights reserved.

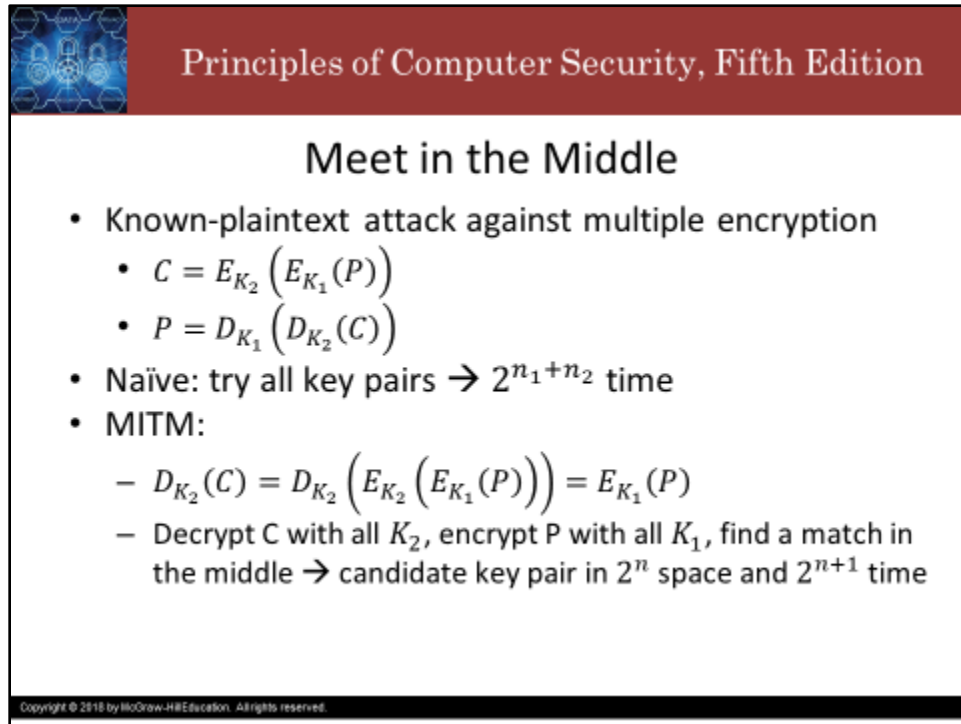
The **birthday attack** is a special type of brute-force attack that gets its name from something known as the birthday paradox, which states that in a group of at least 23 people, the chance that two individuals will have the same birthday is greater than 50 percent.

Applied to cryptanalysis, the attack is often used to find hash collisions: 2 different inputs that have the same hash value.

If I need to find a collision in an 8-bit hash value (or I only care about matching a particular 8-bit block of the hash value), I only need to generate about  $1.25 * \sqrt{256} = 20$  values in order to do it. I hashed the bytes representing numbers as strings, looking for a match in the last byte. I found one after exactly 20 hashes for the strings '8' and '19'. There are matches in other bytes, too. Byte 12 (indexed from 0 starting on the left) collides for the strings '0' and '1'. It looks like, out of those 20 inputs, only '4', '13',

and '16' have no matching bytes with any of the other values. How much further must we look to find a match in any byte of those 3? Well, '13' has a match at byte 6 with '27', '16' has a match at byte 7 with '28'. '4' has a match at byte 1 with '45'.

Slide 4



Principles of Computer Security, Fifth Edition

### Meet in the Middle

- Known-plaintext attack against multiple encryption
  - $C = E_{K_2}(E_{K_1}(P))$
  - $P = D_{K_1}(D_{K_2}(C))$
- Naïve: try all key pairs  $\rightarrow 2^{n_1+n_2}$  time
- MITM:
  - $D_{K_2}(C) = D_{K_2}(E_{K_2}(E_{K_1}(P))) = E_{K_1}(P)$
  - Decrypt C with all  $K_2$ , encrypt P with all  $K_1$ , find a match in the middle  $\rightarrow$  candidate key pair in  $2^n$  space and  $2^{n+1}$  time

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

The meet-in-the-middle attack is a known-plaintext attack against encryption schemes that rely on performing multiple encryption operations in sequence, like Triple DES. It uses a space–time tradeoff to speed up the attack. The meet-in-the-middle attack is the primary reason why Double DES is not used and why a Triple DES key can be brute forced by an attacker with  $2^{56}$  space and  $2^{112}$  operations.

If you want to attack an algorithm which uses 2 encryption steps. You might try to brute force the key by trying all pairs of keys. If the keys are  $n_1$  and  $n_2$  bits long, this takes  $2^{n_1 + n_2}$  time .

The meet-in-the-middle attack uses a more efficient approach. Notice that the first decrypt step yields the encryption of the plain text with key 1. So, what the meet-in-the-middle attacker does is to decrypt the cipher for all possible values of key 2. This takes  $2^{n_2}$  time and  $2^{n_2}$  space to store all of the half-decrypted ciphertexts. Then, they encrypt the known plaintext using all possible values for key 1, which takes  $2^{n_1}$  time and  $2^{n_1}$  space to store the half-encrypted plaintexts. All that is left to do is search for a match between the half decrypted ciphertexts and the half encrypted plaintexts which can be done in  $2^{\min(n_1, n_2)}$  operations. When a match is found, the corresponding key pair becomes a candidate key, which can be verified by using it to decrypt a different ciphertext of a known plaintext. All together this attack takes  $2^{n_1} + 2^{n_2}$  space and  $2^{n_1} + 2^{n_2} + 2^{\min(n_1, n_2)}$  time. If  $n_1 = n_2 = n$ , this simplifies to  $2^{n+1}$  space and  $3 \cdot 2^n$  time. These costs can be reduced by observing that we don't actually need to store both the half-decrypted and half-encrypted data, we only need to keep the whole set of one of



them and the other we can use each as soon as we compute it and then throw it away and move on. This reduces the total cost to  $2^n$  space and  $2^{n+1}$  time.

## Slide 5

Principles of Computer Security, Fifth Edition

### Password Attacks

- Brute force – try every possible password
- Dictionary – try every password in a list
- Rainbow Tables – precompute hash chains to aide in inverting the hash
- Online – real time
  - defeated by restricting and slowing down multiple failed logins
- Offline – any time
  - defeated by strong passwords and salty hashes

<https://haveibeenpwned.com/>  
<https://crackstation.net/>

Copyright © 2018 by IIS/Ozrow-HREducation. All rights reserved.

Password are very commonly used for authentication but are a notorious weak point in security. For that reason, they are a popular target of attack.

The least technical attack is **brute force**, where the attacker simply tries all possibilities to guess the password of an authorized user of the system or network.

Advanced versions of the brute force attack include the dictionary attack and rainbow tables.

A **dictionary attack** reduces the size of the search space by using a dictionary of common, weak, or compromised passwords, thereby significantly speeding up the process of such passwords. Strong passwords are not crackable using a dictionary attack since a strong password will not be in any dictionary.


**Rainbow tables** are precomputed tables of hash values associated with passwords. This can change the search for a password from a computational problem to a lookup problem. This attack is another example of a space-time tradeoff, where the attacker can speed up their attack by doing work and storing it before the attack. The rainbow table attack is very effective against weak passwords, even ones that are not in a dictionary, especially short passwords. The standard

defense against rainbow tables is to add salts to the hashes. A salt value is a random bitstring that gets concatenated to the password, or to a hash of the password, and the whole salty thing gets hashed again. By using large salts and a different salt for every user, the precomputation of the rainbow table is completely nerfed because the attacker would need a rainbow table for every salt value.

Password attacks can be online or offline.

Online attacks are against a live system in real time. Like sitting down at a machine and trying to login. Only, they're typically done by machines themselves, remotely, and they go really fast. Online attacks are easily defended against by limiting the number of failed login attempts and locking the account once that limit is reached and by making the time required to check the password very large (but not so large that users get annoyed by waiting... just a few hundred milliseconds is enough to make the attack's life miserable).

Offline attacks are made directly against the data, usually a password hash dump from a data breach. These kinds of things happen often, and the number of hashes that get dumped can number into the hundreds of millions. Check out [I been pwned dot com](http://Ibeenpwned.com) to see if yours are any of the over 11 billion pwned accounts. In an offline attack, the attacker has all the time they need and they can't be slowed down by interface limitations or access policies. This type of attack is also known as hash cracking. There are companies that provide hash cracking services for a fee. There are also sites that will do it for free for fun and education. The best way to defend against an offline attack is to use a strong password (which means a long password) and to use salted hashes. The user can only control the length of their password and it's complexity (unless the system idiotically constrains them into making a weak password, which does actually happen). The system designers must design in the use of salted hashes for authentication. And they must use and store that data securely. There have been data breaches where it was discovered that passwords were stored in plaintext, which makes for a very easy offline attack.



Principles of Computer Security, Fifth Edition

### A few other examples


- Collision Attack – find two inputs  $x, y$  with the same hash value  $h(x) = h(y), x \neq y$ 
  - Chosen-prefix collision attack – given prefixes  $p1, p2$ , find suffixes  $m1, m2$  such that  $h(p1 // m1) = h(p2 // m2)$ 
    - here, // means concatenate
  - See also: birthday attack
- Downgrade Attack – make server use a low- or no-security mode of operation.
- Replay Attack – listen to valid traffic and repeat it later

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

A collision attack tries to find two different inputs that produce the same output from a hash function, i.e. find a hash collision. The collision attack is a special case of a chosen-prefix collision attack which tries to find values that can be concatenated to the end of the target chosen inputs such that the hash values of the extended values are equal. If the prefixes are empty, then we get the classic collision attack. The best defense against collision attacks is to use strong hash functions.

A downgrade attack is when the attacker takes advantage of a server which supports backward compatibility and forces it to downgrade the security of the connection to a lower or nonexistent state, such as using SSL instead of TLS, or using standard HTTP instead of Secure HTTP. The best defense against downgrade attacks is to remove backward compatibility.

A replay attack is a network attack in which the attacker captures some valid data in transit, like authentication packets, and then resends it later to fool the target into thinking the data from the attacker is a valid response. There are many ways to defend against replay attacks. A general countermeasure is to give every connection a unique random session ID, which gets included in each encrypted message. If an attacker captures some traffic and tries to replay it later, the target will reject the data because the session ID will be incorrect.



## Principles of Computer Security, Fifth Edition

### Attribution

- The course slides are based on slides developed by the textbook authors, W. "Art" Conklin, Ph.D. (University of Houston), and Greg White, Ph.D. (University of Texas at San Antonio).
- These slides remain the intellectual property protected by US Copyright law of the authors and the textbook company.
- There have been some changes to the slide deck.

Rule 1 of cryptanalysis: check for plaintext.

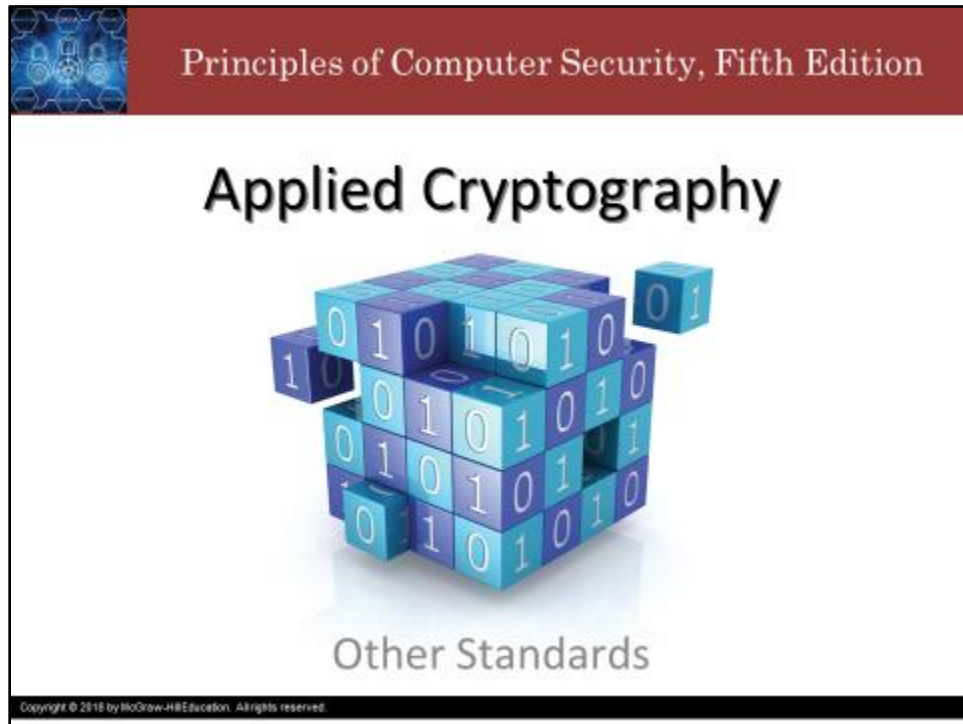
— *Robert Morris*

Copyright © 2018 by McGraw-Hill Education. All rights reserved.


Thank you and take care.

## Applied Cryptography: Other Standards

Slide 1



Howdy! In this video, we introduce a few of the many additional standards associated with information security in general.



## Principles of Computer Security, Fifth Edition

### Federal Information Processing Standards

- FIPS Pubs / FIPS
- Issued by US Gov't through NIST
- Used by non-military US Gov't agencies and contractors
- Examples
  - (withdrawn 2005) FIPS 46-3 – DES, 3DES
  - FIPS 140-2 – Security Requirements for Crypto Modules
  - FIPS 180-4 – SHA-1, SHA-2
  - FIPS 186-6 – DSS
  - FIPS 197 – AES
  - FIPS 198-1 – Keyed HMAC
  - FIPS 202 – SHA-3

<https://www.nist.gov/itl/fips>

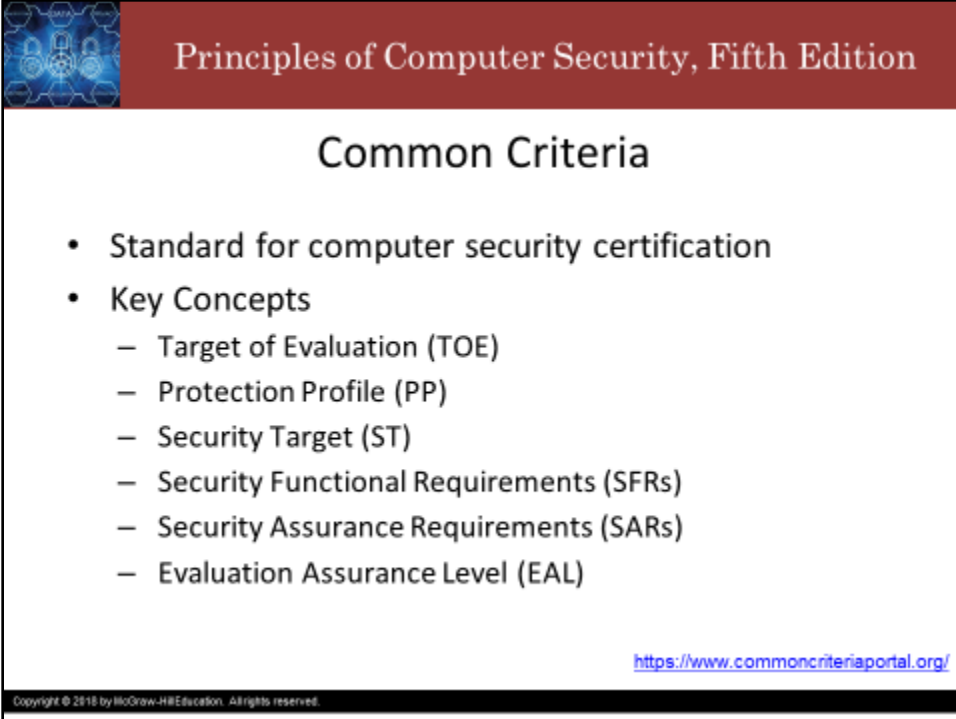
Copyright © 2018 by McGraw-Hill Education. All rights reserved.

The **Federal Information Processing Standards Publications (FIPS PUBS)** or simply **FIPS** describe various standards for data communication issues.

FIPS are developed by NIST (the National Institute of Standards and Technology) when the US government needs a standard but there isn't a recognized industry standard they can use.

They are published publicly for use by non-military American government agencies and government contractors.

There are currently 9 active FIPS, including the Secure Hash Standard, the Advanced Encryption Standard, and the Digital Signature Standard.



Principles of Computer Security, Fifth Edition

## Common Criteria

- Standard for computer security certification
- Key Concepts
  - Target of Evaluation (TOE)
  - Protection Profile (PP)
  - Security Target (ST)
  - Security Functional Requirements (SFRs)
  - Security Assurance Requirements (SARs)
  - Evaluation Assurance Level (EAL)

<https://www.commoncriteriaportal.org/>

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Common Criteria provides assurance that the process of specification, implementation and evaluation of a computer security product has been conducted in a rigorous and standard and repeatable manner at a level that is commensurate with the target environment for use.

Common Criteria maintains a list of certified products, including operating systems, access control systems, databases, and key management systems.

The **Target of Evaluation** is the product or system that is the subject of the evaluation.

The **Protection Profile** identifies the security requirements for a class of security devices relevant to the user for a particular purpose.

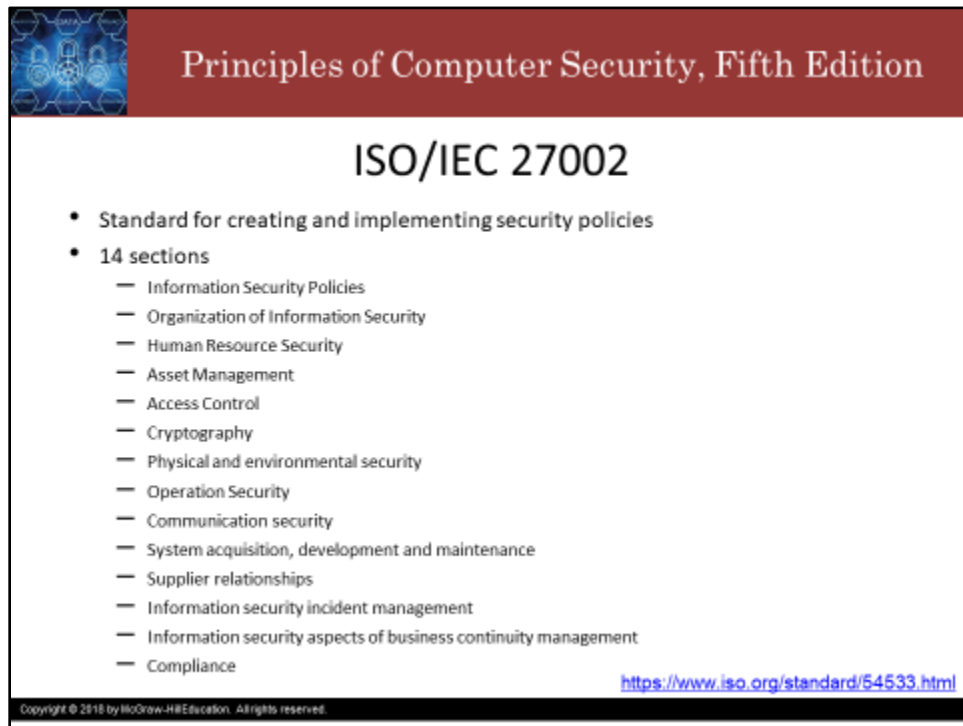
The **Security Target** identifies the security properties of the T O E.

**Security Functional Requirements** specify individual security functions which may be provided by a product.

**Security Assurance Requirements** describe the measures taken during development and evaluation of the product to assure compliance with the claimed security functionality.

The **Evaluation Assurance Level** is a numerical rating describing the depth and rigor of an evaluation with E A L 1 being the most basic and therefore cheapest to implement and evaluate and E A L 7 being the most stringent and most expensive.

Slide 4



**Principles of Computer Security, Fifth Edition**

## ISO/IEC 27002

- Standard for creating and implementing security policies
- 14 sections
  - Information Security Policies
  - Organization of Information Security
  - Human Resource Security
  - Asset Management
  - Access Control
  - Cryptography
  - Physical and environmental security
  - Operation Security
  - Communication security
  - System acquisition, development and maintenance
  - Supplier relationships
  - Information security incident management
  - Information security aspects of business continuity management
  - Compliance


<https://www.iso.org/standard/54533.html>

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

ISO/IEC 27 0 0 2 gives guidelines for organizational information security standards and information security management practices including the selection, implementation and management of controls taking into consideration the organization's information security risk environments.

The standard covers a wide range of policies divided into 14 chapters, which you can see here include things like access control, physical security, compliance, and, of course, cryptography.





## Principles of Computer Security, Fifth Edition

### Attribution

- The course slides are based on slides developed by the textbook authors, W. "Art" Conklin, Ph.D. (University of Houston), and Greg White, Ph.D. (University of Texas at San Antonio).
- These slides remain the intellectual property protected by US Copyright law of the authors and the textbook company.
- There have been some changes to the slide deck.

FIPS 160 C (Programming Language)  
Published 13 Mar 1991  
Withdrawn 25 Feb 2000

Copyright © 2018 by McGraw-Hill Education. All rights reserved.

Thank you and take care.