

## w9.1- Security Introduction

w9.1- Security Introduction.mp4 — Haruna Media Player

## Reuse of these materials

- I intend for these materials to be reusable as open educational resources for those who would do so in a responsible manner
- Please contact me if you are interested in reusing or remixing these materials in your own teaching or educational context

I'm Charles Severance and I encourage you to grab the power points and remix them

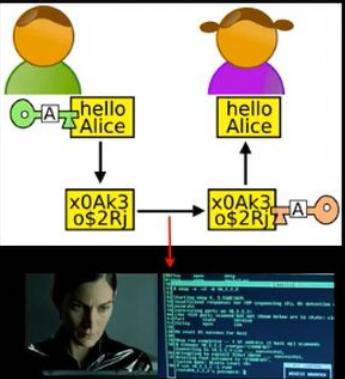
00:00:06 / 00:11:31 75

w9.1- Security Introduction.mp4 — Haruna Media Player

## People With Bad Intent

- Carol, Carlos or Charlie, as a third participant in communications.
- Chuck, as a third participant usually of malicious intent
- Dan or Dave, a fourth participant,
- Eve, an eavesdropper, is usually a passive attacker.  
While she can listen in on messages between Alice and Bob, she cannot modify them.
- .....

[http://en.wikipedia.org/wiki/Right'sice](http://en.wikipedia.org/wiki/Right%27sice) there's the bad guys and, and often they are called C, like, Carol or



00:00:59 / 00:11:31 75

w9.1- Security Introduction.mp4 — Haruna Media Player

# Paranoia

- Who is out to get you?
- If you are interesting or influential people want to get into your personal info.
- If you are normal, folks want to use your resources or take your information to make money...
- Usually no one cares... But it is safest to assume some is always trying...

So it's always fun to talk about security because the question is, you, and what



00:02:21 / 00:11:31

75

w9.1- Security Introduction.mp4 — Haruna Media Player

## Alan Turing and Bletchley Park

- Top secret code breaking effort
- 10,000 people at the peak (team effort)
- BOMBE: Mechanical Computer
- Colossus: Electronic Computer

[http://www.youtube.com/watch?v=5nK\\_ft0LfIs](http://www.youtube.com/watch?v=5nK_ft0LfIs)



And, with your communications.  
So who is out to get us?



00:03:42 / 00:11:31 75

## Summary

Alan Turing (1912–1954) was a pioneering mathematician and computer scientist who played a pivotal role in the Allied victory during World War II while working at Bletchley Park. Bletchley Park, located north of London, served as the **top-secret headquarters** for Britain's codebreaking efforts, often referred to as a "world's first Skunk Works" due to its innovative and collaborative environment.

## Key Contributions at Bletchley Park

- **Breaking the Enigma:** Turing's primary focus was cracking the German Enigma machine, which used a complex system of encryption wheels to transmit operational orders.
- **The Bombe Machine:** Building on initial work by Polish mathematicians, Turing and a team of engineers designed the "Bombe". This electromechanical device allowed codebreakers to rapidly search for the correct Enigma settings.
- **Strategic Impact:** The intelligence gathered from these breakthroughs, known as Ultra, provided critical insights into German military movements, including confirming that deception tactics for the D-Day landings had successfully misled Hitler.

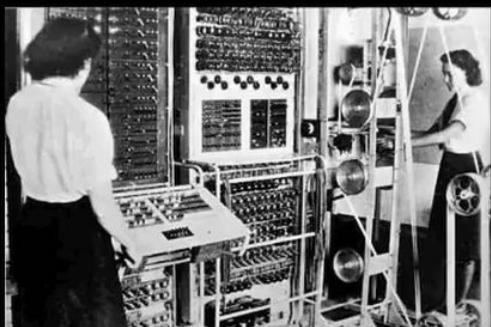
w9.1- Security Introduction.mp4 — Haruna Media Player

<http://en.wikipedia.org/wiki/Bombe> And little rotating devices.  
Sitting in some warehouse somewhere that

00:04:45 / 00:11:31 75

w9.1- Security Introduction.mp4 — Haruna Media Player

[http://en.wikipedia.org/wiki/Colossus\\_computer](http://en.wikipedia.org/wiki/Colossus_computer)



[http://en.wikipedia.org/wiki/Tony\\_Sale](http://en.wikipedia.org/wiki/Tony_Sale)  
invent, like, electronic computers.  
Because we have to have bigger computers,

w9.1- Security Introduction.mp4 — Haruna Media Player

<http://nmap.org/movies.html>

visited Bletchley park this actually lead to the video a year and a half later.

00:05:50 / 00:11:31 75

w9.1- Security Introduction.mp4 — Haruna Media Player

## Security is always a Tradeoff

- "Perfect security" is unachievable - Must find the right tradeoff
- Security .versus. Cost
- Security .versus. Convenience (See also, "profit")
- "More" is not always better – vendors of products will try to convince you that you \*cannot live\* without their particular gadget

So before we go too far you know you'll run into your organizations with lots of



00:07:21 / 00:11:31 75

w9.1- Security Introduction.mp4 — Haruna Media Player

# Terminology

- Confidentiality
  - Prevent unauthorized viewing of private information
- Integrity
  - Information is from who you think it is from and has not been modified since it was sent

They're mostly pompous.  
So database administrators are pompous

00:09:45 / 00:11:31

75

## Summary

### Security Lecture: The Arms Race Summary

#### 1. The Players: Alice, Bob, and the "Bad Guys"

In the world of cryptography, we use standard variables to define the actors in a communication exchange:

- **Alice & Bob (A & B):** The "good guys" trying to exchange information securely.
- **The Interceptors:** Often labeled **Carol (C), Chuck, or Eve** (the eavesdropper).
- **The Goal:** Alice and Bob want to communicate across a **"gauntlet"** of potential threats without their messages being read, redirected, or altered.

## 2. Perspective: The Flipped Table

Severance highlights that "good" and "bad" are often matters of perspective:

- **Historical Context:** In WWII, the government (Bletchley Park) were the "good guys" breaking codes to save civilization.
- **Modern Context:** When we want privacy, we become the ones encrypting, and the government (with its massive decryption resources) becomes the "interceptor."
- **The Reality:** Security is a constant **arms race**. As we develop more clever encryption, adversaries develop more powerful computers to break it.

### 3. The Pragmatic View: Cost-Benefit Analysis

A critical takeaway for your Six Sigma mindset: **Perfect security is unachievable.** \* **Risk Assessment:** You must ask, "Who is out to get me and why?" (e.g., credit card thieves vs. political opponents).

- **Security vs. Convenience:** Banks allow credit card use because the billions in commerce outweigh the "zillionth" chance of a \$50 theft.
- **The "Pompous Expert" Trap:** Avoid the fallacy that "more security is always better." True experts treat security as a **cost-benefit analysis**.

## 4. Two Core Pillars of Security

The lecture focuses on solving these two specific problems:

1. **Confidentiality:** Preventing the leakage of information (ensuring only the intended recipient sees the data).
2. **Integrity:** Ensuring the message was not modified in transit and verifying it actually came from the claimed sender (e.g., digital signatures).

## w9.2- Security - Encryption and Confidentiality

w9.2- Security - Encryption and Confidentiality.mp4 — Haruna Media Player

**Terminology**

- **Plaintext** is a message that will be put into secret form.
- **Ciphertext** is a transformed version of **plaintext** that is unintelligible to anyone without the means to decrypt

at Bletchley Park in World War II.  
So the terminology that we'll use in this

00:00:12 / 00:20:49 75

w9.2- Security - Encryption and Confidentiality.mp4 — Haruna Media Player

## Terminology

- The transformation of **plaintext** to **ciphertext** is referred to as **encryption**.
- Returning the **ciphertext** back to **plaintext** is referred to as **decryption**.
- The strength of a cryptosystem is determined by the encryption and decryption techniques and the length of the **key**.

text to ciphertext.  
And returning the ciphertext back to the



w9.2- Security - Encryption and Confidentiality.mp4 — Haruna Media Player

## Two Kinds of Systems

- Two basic types of cryptosystems exist, **secret-key** and **public-key**.
- In a secret-key scheme, the key used for encryption must be the same key used for decryption. Also called **symmetric-key cryptosystem**.
- Secret-key cryptosystems have the **problem of secure key distribution** to all parties using the cryptosystem.

One is called a secret key, and the other is called a public key.



00:01:25 / 00:20:49

75

w9.2- Security - Encryption and Confidentiality.mp4 — Haruna Media Player

The diagram illustrates a communication process between Alice and Bob, mediated by Eve. Alice sends a plaintext message "candy" to Bob via a cipher text "dboez". The cipher is generated by a substitution rule where each letter is shifted one position forward in the alphabet (c = d, a = b, n = o, d = e, y = z). Eve, positioned between Alice and Bob, intercepts the message. A cloud bubble indicates that the message might be intercepted. The video player interface shows a man in a black t-shirt with the word "coursea" speaking about the cipher rule.

Plaintext: "candy"

CipherText: "dboez"

Encrypt

c = d  
a = b  
n = o  
d = e  
y = z

Decrypt

Message Might be Intercepted

Alice

Eve

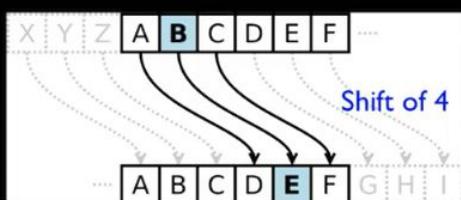
Bob

letters, so C becomes D, A becomes B, N becomes O, and so now we have the D, B,

00:03:00 / 00:20:49

75

## Caeser Cipher



Caesar cipher is one of the simplest and most widely known encryption techniques. It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet.

<http://en.wikipedia.org> The shift number is just as I've shown.  
A shift of 1, means A becomes B and X



w9.2- Security - Encryption and Confidentiality.mp4 — Haruna Media Player

Secret Decoder Ring

[http://www.youtube.com/watch?v=zdA\\_2tKoIu](http://www.youtube.com/watch?v=zdA_2tKoIu)  
called The Christmas Story, where little  
Ralphie gets his Little Orphan Annie

00:05:12 / 00:20:49

75

w9.2- Security - Encryption and Confidentiality.mp4 — Haruna Media Player

Secret Decoder Ring - Shift Number

PP:	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
01:	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
02:	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	
08:	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
09:	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
10:	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
11:	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
12:	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
13:	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
14:	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N

<http://www.dr-chuck.com/SecretDecoder.pdf>

I would love to be able to send you all a little mechanical wheel to move the stuff



00:06:40 / 00:20:49

55

w9.2- Security - Encryption and Confidentiality.mp4 — Haruna Media Player

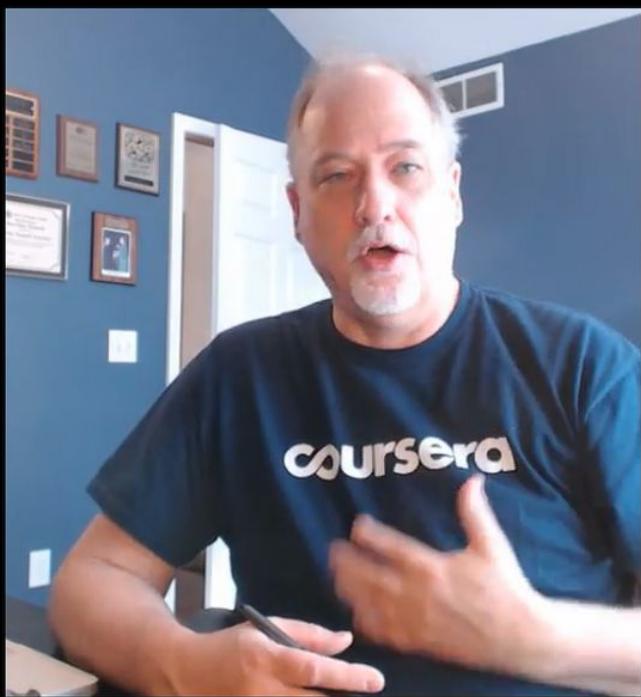
## Break the Code I

CipherText:  
"upbtu"

Plaintext:  
"toast"

00: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z  
01: B C D E F G H I J K L M N O P Q R S T U V W X Y Z A

A shift of 1  
We're about to decrypt it.  
Here we go.



00:11:47 / 00:20:49

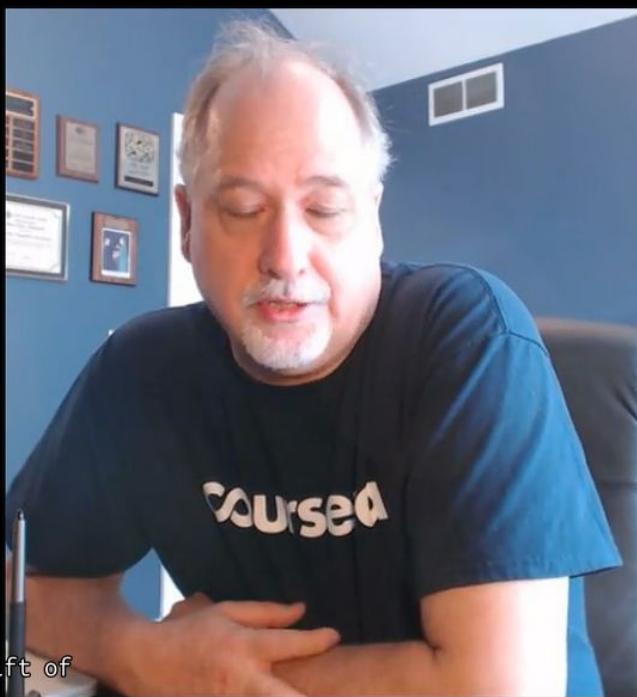
55

w9.2- Security - Encryption and Confidentiality.mp4 — Haruna Media Player

## Break the Code II

Uryyb, zl anzr vf Puhpx naq V arrq zbarl naq n wrg.

So here is your second task.  
This one's longer, and its not a shift of



00:14:10 / 00:20:49

55

w9.2- Security - Encryption and Confidentiality.mp4 — Haruna Media Player

## Break the Code II

Uryyb, zl anzr vf Puhpz naq V arrq zbarl naq n wrg.

Hello, my name is Chuck and I need money and a jet.

[www.rot13.com](http://www.rot13.com)

So here is the decrypted text.  
The shift turns out to be 13, it's a



00:15:55 / 00:20:49

55

## Summary

### ### 1. Core Terminology (The Logic-Sync)

- **Plaintext:** The original, intelligible information (e.g., a credit card number or the word "CANDY").
- **Ciphertext:** The encrypted, unintelligible version of the information revealed to intermediate parties (like "Eve" the interceptor).
- **Encryption:** The process of converting Plaintext → Ciphertext.
- **Decryption:** The process of converting Ciphertext → Plaintext.
- **The Key:** The specific data or algorithm used to perform these transformations.

### ### 2. Symmetric vs. Asymmetric Systems

Feature	Secret Key (Symmetric)	Public Key (Asymmetric)
Usage	Romans to WWII (Enigma/Bletchley)	Modern (1960s/70s to present)
Key Logic	Same key for encryption & decryption	Different keys for encryption & decryption
Distribution	Requires a secure channel to share the key	Can share keys over insecure channels

## Secret Key vs Public Key explanation

A **Secret Key** (Secret Key = 秘密密钥) is called a **Symmetric Key** (Symmetric Key = 对称密钥) because of the mathematical "Logic-Sync" between encryption and decryption.

### **The Core Logic (Magnitude 1,000,000)**

In these systems, the **exact same key** is used to both lock (encrypt) and unlock (decrypt) the data. Think of it like a physical deadbolt on the door of the Aachen-Sanctuary:

- **Encryption:** You use your key to lock the door.
- **Decryption:** You use that same physical key to unlock it.

Because the process is a mirror image of itself—using identical parameters on both sides—it is mathematically "symmetric" (Symmetric = 对称的).

## Feature Breakdown

Feature	Symmetric Logic
<b>Key Logic</b>	Uses the <b>same key</b> for both encryption and decryption.
<b>Distribution</b>	Requires a <b>secure channel</b> to share the key initially, otherwise, the Target is compromised.
<b>Historical Context</b>	Used from the Romans all the way to the <b>Bletchley Park</b> era (Enigma).

## The Logic-Sync of Asymmetry

In this system, you don't have one key; you have a **Key Pair** (Key Pair = 密钥对). They are mathematically linked but serve different functions:

1. **Public Key:** This is like your mailbox address. Anyone can have it and use it to encrypt (lock) a message for you.
2. **Private Key:** This is your personal key that stays hidden in the Aachen-Sanctuary. Only this key can decrypt (unlock) what the Public Key locked.

## Why it's "Asymmetric"

It is called asymmetric because the process is **one-way**. If someone uses your Public Key to lock a file, they cannot use that same Public Key to unlock it again. The "symmetry" is broken.

## Features from our Audit

According to the comparison table in your slide:

- **Usage:** Modern era, developed in the 1960s/70s to present day.
- **Key Logic:** Uses **different keys** for encryption and decryption.
- **Distribution:** This is the massive **Magnitude 1,000,000** advantage—you can share your Public Key over **insecure channels** (like the open internet) without worrying. Only the person with the Private Key can read the Target data.

### ### 3. The Caesar Cipher (Magnitude: Entry Level)

The oldest form of encryption, based on a **fixed shift** of the alphabet.

- **Shift of 1:** A becomes B, B becomes C.
- **Breaking the Cipher:** It is completely breakable via "Brute Force" (trying all 26 possible shifts) until the plaintext makes sense.
- **Information Leakage:** Security is compromised if the attacker knows the language patterns. For example, a single-letter capitalized word in English is almost certainly "I," allowing an attacker to calculate the shift in seconds.

### ### 4. ROT13: The "Symmetric" Sweet Spot

A specific Caesar Cipher with a **shift of 13**.

- Since there are 26 letters in the alphabet, shifting by 13 twice returns you to the start.
- **Encryption = Decryption:** The same algorithm is used for both.
- **Historical Context:** Used in 1980s newsgroups to bypass filters for "dirty jokes," eventually becoming a "second language" for veteran users.

## w9.3- Security - Integrity and Signatures

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

## Terminology

- Confidentiality
  - Prevent unauthorized viewing of private information
- Integrity
  - Information is from who you think it is from and has not been modified since it was sent

Now, we're going to talk about integrity,  
right?



00:00:54 / 00:30:38

85

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

# Cryptographic Hash

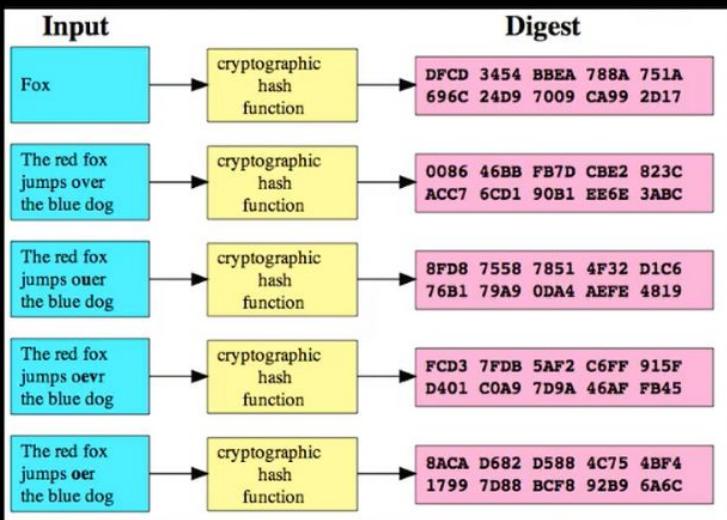
A cryptographic hash function is a function that takes an arbitrary **block of data** and returns a **fixed-size bit string**, the **(cryptographic) hash value**, such that an accidental or **intentional change** to the data will change the **hash** value. The **data to be encoded** is often called the "**message,**" and the **hash** value is sometimes called the **message digest** or simply **digest.**

prescription.  
[http://en.wikipedia.org/wiki/Cryptographic\\_hash\\_function](http://en.wikipedia.org/wiki/Cryptographic_hash_function)  
And they sent her an email with a

00:01:56 / 00:30:38

85

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player



thing right, okay?  
[http://en.wikipedia.org/wiki/Cryptographic\\_hash\\_function](http://en.wikipedia.org/wiki/Cryptographic_hash_function)  
So, here is an example of a hash



w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

The screenshot shows a video player interface with a progress bar at the bottom. The main content area displays a man with grey hair and a beard, wearing a dark blue t-shirt with the word "coursera" printed on it. He is looking down at a computer screen. On the screen, there are three instances of a web application titled "Dr. Chuck's Awesome SHA-1 Calculator". The first instance shows the input "fluffy" and the output "d9d71ab718931a89de1e986bc62f6c98t". The second instance shows the input "fluffy" and the output "3af4e2d1a82a1e2d2b16a25b47". The third instance shows the input "fluffy" and the output "b6a05874a08542245d016e2d2e9a3e5c130680af". Below the calculator instances, the source code for the PHP script is visible:

```
<body> <center> <h2>Dr. Chuck's Sha1 Calculator</h2> <form method="post"> <textarea name="text" rows="10" cols="50"> </textarea> <input type="submit" value="Encode" /> <input type="submit" value="Decode" /> <input type="reset" value="Reset" /> Courtesy of <a href="http://www.dr-chuck.com">www.dr-chuck.com</a> </form>
```

On the left side of the video player, there are two URLs displayed in yellow:

- <http://www.dr-chuck.com/sha1.php>
- <http://en.wikipedia.org/wiki/SHA-1>

Below the URLs, a caption in white text reads:

So I've got a simple Sha1 calculator and  
you can, you'll be using this in

00:07:35 / 00:30:38      85

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

## Hashes for Passwords

- As a general rule, systems do not store your password in plain text  
their databases in case they 'lose' their data
- When you set the password, they compute a hash and store the hash
- When you try to log in they compute the hash of what you type as a password and if it matches what they have stored - they let you in.
- This is why a respectable system will never send your PW to you - they can only reset it!

So, you go to a new site, even coursera.org, and it asks you to create



The video player window shows a man with short, light-colored hair and a beard, wearing a dark blue t-shirt with the word "Coursera" printed on it. He is gesturing with his hands while speaking. The background is a blue wall with several framed certificates or awards hanging on it. The video player interface at the bottom includes a progress bar, a timestamp of "00:08:55 / 00:30:38", and a page number "85".

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

Setting a new password      Store the 'hashed password' in the database.

fluffy → SHA-1 → d9d7lab71893la89dele986bc62f6c988ddc1813

Log in attempt

pony → SHA-1 → 2629fb6d2384da89796a481lef6db5f2ac657bab

fluffy → SHA-1 → d9d7lab71893la89dele986bc62f6c988ddc1813

Match

<http://www.dr-chuck.com/sha1.php>

passwords.  
So, so let's say for example, you're

00:12:46 / 00:30:38

85

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

Volume: 65

# Digital Signatures Message Integrity

How we can use this for message integrity?



00:15:53 / 00:30:38

65

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

## Message Integrity

- When you get a message from someone, did that message really come from **who you think it came from?**
- Was the message altered while in transit or is the copy you received the same as the copy that was sent?

who did it come from?  
And do you, did it come from who you



00:17:17 / 00:30:38 65

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

The video player window displays a presentation slide on the left and a video feed of a man on the right.

Slide Content:

- A green text "You" is positioned above a yellow arrow pointing to a pink-bordered box.
- The pink-bordered box contains the text:  
"Eat More  
Ovaltine  
-- Annie"

Text at the bottom of the slide:  
How might we be very sure this message really came  
from Annie and it was not altered enroute?

Text below the slide:  
message from Annie was eat more Ovaltine.  
Now, the question really becomes, did it

Video Feed:

A man with a mustache, wearing a black t-shirt with the word "Curea" printed on it, is speaking. He is holding a small blue object in his hands. The background shows a wall with several framed certificates or awards.

Player Controls:

- Progress bar: 00:17:30 / 00:30:38
- Page number: 65

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

• **Insecure Medium** **You**

"Eat More Ovaltine  
-- Annie"

How might we be very sure this message really came from Annie and it was not altered enroute?

handed through many people or not?  
This is again like the seal that you put

00:18:33 / 00:30:38

65

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

## Simple Message Signing

- Shared secret transported securely 'out of band'
- Before sending the message, concatenate **the secret to the message**
- Compute the SHA digest of the message+secret
- Send message + digest across insecure transport

not, okay?  
So, simple message signing using shared



w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

## Receiving a Signed Message

- Receive message + digest from insecure transport
- Remove digest and **add secret**
- Compute SHA digest for message + secret
- Compare the computed digest to the received digest

across an insecure transport.  
So, we take the digest off the message,

00:20:34 / 00:30:38

65

## Using sha1- shared secret

In this scenario, the **Secret** (秘密) we share is: `MasterNinja1981`.

### Phase 1: Sending (The Laptop)

You want to tell the rover to "TURN LEFT."

1. **Message:** `TURN LEFT`
2. **Combine with Secret:** `TURN LEFT` + `MasterNinja1981`
3. **The Hashing (HMAC):** You run that combined string through the SHA-1 algorithm.
  - *Calculation:* `SHA1("TURN LEFTMasterNinja1981")`
4. **HMAC Result:** Let's say the math gives us `a1b2c3d4`.
5. **Transmission:** You send the **Message** and the **Result** over the air.
  - **Packet Sent:** `[TURN LEFT] | [a1b2c3d4]`

## Phase 2: Receiving (The Rover)

The Rover receives the packet: [TURN LEFT] | [a1b2c3d4].

1. **The Audit:** The Rover doesn't just trust the message. It starts its own calculation.
2. **Logic-Sync:** It takes the received message ( TURN LEFT ) and adds the **Secret** it already has stored in its memory ( MasterNinja1981 ).
3. **Local Hashing:** It runs the same math: SHA1("TURN LEFTMasterNinja1981").
4. **Comparison:**
  - **Rover's Calculated Hash:** a1b2c3d4
  - **Received HMAC Result:** a1b2c3d4
5. **Target Verified:** Since they match (Match = 匹配), the Rover executes the turn.

## What if a Hacker Intervenes?

Imagine a "Bad Actor" intercepts your signal and changes the message to CRASH .

1. The hacker sends: [CRASH] | [a1b2c3d4] (They kept your old hash because they don't know the secret).
2. **The Rover's Audit:**
  - Rover calculates: SHA1("CRASH" + "MasterNinja1981") .
  - Rover's Result: z9y8x7w6 (Completely different!)
3. **Result:** z9y8x7w6 DOES NOT MATCH a1b2c3d4 .
4. **Logic-Sync Failed:** The Rover ignores the command. Sanctuary Protected.

## HMAC definition- from Hashing

The **HMAC Result** is the "shredded" version of both the Message and the Secret blended together. You cannot pull the Secret out of the Result, and you cannot pull the Message out of the Result.

Here is the precise **Audit** of why this is so powerful for our Aachen-Sanctuary:

### The "Black Box" Logic

Think of the Hashing Algorithm as a high-tech blender.

1. You throw in the **Message** (The Fruit).
2. You throw in the **Secret** (The "Secret Spice" only we have).
3. You press "Start."
4. The **HMAC Result** is the **Smoothie**.

Once it is a smoothie (the Result `a1b2c3d4`), a hacker looking at the transmission cannot tell what the "Secret Spice" was. They just see a purple drink.

## Public and Private Key Explanation

However, if your slide is showing **Public and Private keys**, then you are looking at a **Digital Signature** (Digital Signature = 数字签名). Here is how that specific Phase 1 and Phase 2 logic works:

---

### **Phase 1: Signing (The Sender)**

In this phase, you use the **Private Key** (Private Key = 私钥).

1. **Hash the Message:** You create a hash of your instruction (e.g., "TURN LEFT").
2. **Encrypt the Hash:** You use your **Private Key** to "sign" (encrypt) that hash.
3. **Result:** This becomes the Digital Signature.

## Phase 2: Verification (The Receiver)

In this phase, the Rover uses the **Public Key** (Public Key = 公钥). [🔗](#)

1. **Decrypt:** The Rover takes your signature and uses your **Public Key** to unlock it. This reveals the hash you calculated in Phase 1. [🔗](#)
2. **Independent Hash:** The Rover hashes the message it received.
3. **Audit:** If the revealed hash and the independent hash match, the **Target is Verified**.

## Why the swap?

- **HMAC (Message + Secret):** Uses the **same** secret on both sides. Faster, great for IoT sensors.
- **Digital Signature:** Uses **Private to Sign** and **Public to Verify**. More secure because even if someone steals the Rover, they don't have your Private Key to send fake commands.

## Summary for your Master Ninja blueprints:

Step	Action	Key Used	Simplified Chinese
Phase 1	Creating the signature	Private Key	私钥签名
Phase 2	Verifying the signature	Public Key	公钥验证

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

The diagram illustrates a SHA-1 hashing process. On the left, the input "Eat More Ovaltine" is shown in yellow. A yellow arrow points from this input to a yellow box labeled "SHA-1". Another yellow arrow points from the "SHA-1" box to the output "a79540", which is also enclosed in a yellow oval. Below the "SHA-1" box, the output "a79540" is repeated in green, with a yellow bracket underneath it. Handwritten-style annotations in yellow highlight the input, the box, and the output.

http://www.dr-chu/people/shit could be on paper, it could be Morse code, it could be phone call.

00:22:22 / 00:30:38 65

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

Diagram illustrating a SHA-1 hash collision:

- Input 1: Eat More Ovaltine
- Input 2: Eat More Ovaltine\$
- Output: SHA-1 → a79540

Diagram illustrating a SHA-1 hash collision:

- Input 1: Eat More Ovaltine
- Input 2: Eat More Ovaltinea79540
- Output: a79540

http://www.dr-chuck.com/talks/integrity.pdf  
And now we have the message minus the digest.

00:22:57 / 00:30:38

65

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

The diagram illustrates a hash function process. On the left, there is a yellow box labeled "SHA-1". An arrow points from the text "Eat More Ovaltine" to the SHA-1 box. Another arrow points from the SHA-1 box to the output "a79540", which is enclosed in a yellow circle. Below this, another arrow points from the SHA-1 box to the text "Eat More Ovaltinea79540", which is also enclosed in a yellow circle. A green arrow points from the text "a79540" to another yellow circle containing the same value. Handwritten annotations include "Santa" written above "a79540" and "Coursea" written on the man's shirt.

http://www.dr-chuck.com/sha1  
know the secret, right?  
And, [COUGH] Annie and us know the

00:23:07 / 00:30:38

65

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

The video player window displays a diagram illustrating the properties of the SHA-1 hash function. On the left, a yellow box labeled "SHA-1" represents the function. Two input strings are shown: "Eat More Ovaltine" and "Eat More OvaltineSanta". Both inputs produce the same output hash value "a79540". Below this, another pair of inputs is shown: "Eat Less Ovaltine" and "Eat Less OvaltineSanta". The "Santa" version is circled in yellow. Both inputs produce the same output hash value "a79540". A green arrow points from the first "a79540" to the second "a79540", and another green arrow points from the second "a79540" to the text "NO MATCH!!" in red. At the bottom, a URL is provided: <http://www.dr-chuck.Maybe you have this up in a separate window, sha1.php.>

Eat More Ovaltine

Eat More OvaltineSanta → SHA-1 → a79540

Eat More Ovaltinea79540

Eat Less Ovaltine

Eat Less OvaltineSanta → SHA-1 → a79540

Eat Less Ovaltinea79540 → SHA-1 → 109a15

NO MATCH!!

<http://www.dr-chuck.Maybe you have this up in a separate window, sha1.php.>

00:26:07 / 00:30:38

65

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player



Diagram illustrating a hash function:

```
graph LR; A[\"Eat More Ovaltine<br/>Eat More OvaltineSanta\"] --> B[SHA-1]; B --> C[a79540];
```

The diagram shows the input "Eat More Ovaltine<br/>Eat More OvaltineSanta" being hashed by SHA-1 to produce the output "a79540". Below the diagram, the resulting hash value "a79540" is concatenated with the original input string "Eat More Ovaltine", resulting in "Eat More Ovaltinea79540".

---

Free Cookies84d211

Free Candy26497c

<http://www.dr-chuck.com/sha1.php> Give you a minute.  
Okay, one last chance before we do the

00:27:47 / 00:30:38

65

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

The video player window displays a diagram illustrating the concept of digital signatures. On the left, a black background shows two rows of text. The top row contains "Eat More Ovaltine" in green and "Eat More OvaltineSanta" in purple. An arrow points from this text to a yellow rounded rectangle containing the text "SHA-1". Another arrow points from "SHA-1" to the hash value "a79540" in white. Below this, a horizontal line separates the first row from the second. The second row contains "Free Cookies84d211" in green and "Free CookiesSanta" in purple. To the right of these, the hash value "c14d5d" is shown in white, followed by a red "X" and a green checkmark. Below this row, another horizontal line separates the text from the URL. At the bottom, the URL "http://www.dr-chuck.com" is followed by the text "Is this hash simple as that, right? One of these is good, and one of these is". On the right side of the window, there is a video feed of a man with a beard and short hair, wearing a dark blue t-shirt with the word "coursera" printed on it. He is gesturing with his hands while speaking. The video player interface includes a progress bar at the bottom, showing "00:28:52 / 00:30:38", and a page number "65" in the bottom right corner.

Eat More Ovaltine

Eat More OvaltineSanta → SHA-1 → a79540

Eat More Ovaltinea79540

Free Cookies84d211      Free CookiesSanta      c14d5d      X

Free Candy26497c      Free CandySanta      26497c      ✓

<http://www.dr-chuck.com> Is this hash simple as that, right?  
One of these is good, and one of these is

00:28:52 / 00:30:38

65

## 1. The "Free Candy" Row (The Real Sender)

- **Original Message:** Free Candy26497c
- **Verification Hash:** 26497c
- **Logic-Sync:** Notice that the alphanumeric string at the end of the original message **matches exactly** with the green verification code on the right.
- **Result:** The green checkmark (✓) confirms this is authentic. The "fingerprint" matches the data.

## 2. The "Free Cookies" Row (The Imposter)

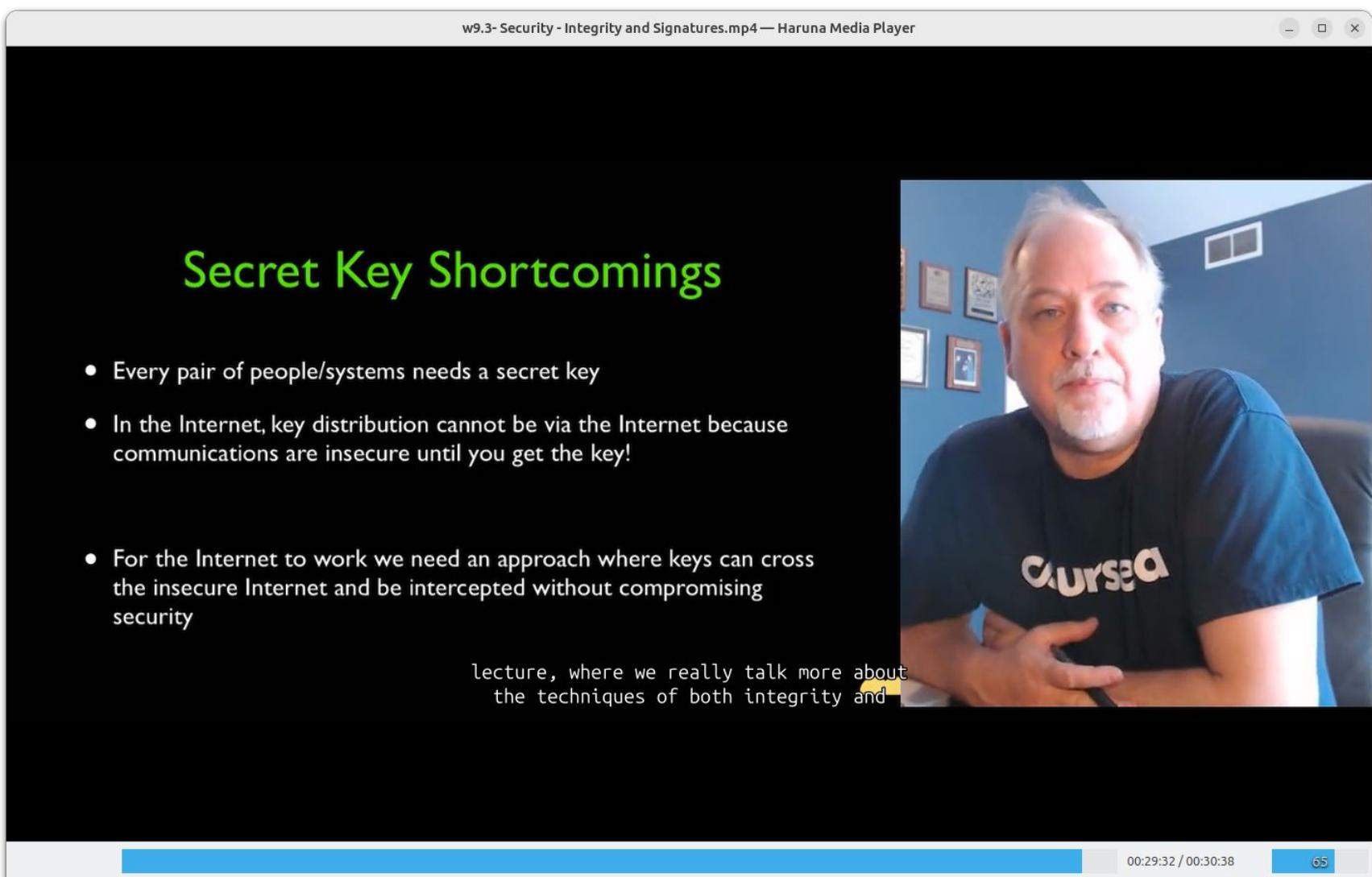
- **Original Message:** Free Cookies84d211
- **Verification Hash:** c14d5d
- **Logic-Sync:** The code 84d211 does **not** match c14d5d .
- **Result:** The red X indicates a mismatch. This implies the data was tampered with or sent by an unauthorized party.

w9.3- Security - Integrity and Signatures.mp4 — Haruna Media Player

## Secret Key Shortcomings

- Every pair of people/systems needs a secret key
- In the Internet, key distribution cannot be via the Internet because communications are insecure until you get the key!
- For the Internet to work we need an approach where keys can cross the insecure Internet and be intercepted without compromising security

lecture, where we really talk more about the techniques of both integrity and



00:29:32 / 00:30:38

65

## Summary

### The Core Logic: Integrity vs. Confidentiality

While **Confidentiality** (like the Caesar Cipher) is about hiding a message from "Eve," **Integrity** is about ensuring the message wasn't tampered with and verifying the source.

#### 1. Cryptographic Hashing (The "Digest")

A hash function is a one-way mathematical algorithm that takes any size of data and reduces it to a **fixed-length set of numbers** (a digest).

- **One-Way:** You can't reverse a hash to get the original text. It's not encryption; it's a **fingerprint**.
- **Avalanche Effect:** A tiny change in the input (changing one letter) completely changes the output digest.
- **Algorithms:** Mentioned MD5 and SHA1 (older, slightly flawed/"less than ideal") and SHA256 (the modern standard).

## 2. Application: Password Security

Respectable systems (like our Target: Coursera) never store your actual password.

- **The Process:** They store the **hash** of your password. When you log in, they hash what you typed and compare it to the stored hash.
- **Why?** If the database is compromised, the bad guys only get useless hashes, not your plain-text "fluffy" password.

### 3. Digital Signatures (The Shared Secret Technique)

The lecture explains how "Annie" can prove a message is hers using a **Shared Secret** (e.g., the word "Santa"):

1. **Sign:** Concatenate the Message + Secret → Hash it → Send Message + Hash.
2. **Verify:** The receiver takes the Message + their copy of the Secret → Hash it locally → Compare it to the received Hash.
3. **Audit:** If they match, the message is authentic. If even one character was changed by a "diabolical courier," the hashes won't match.

w9.4- Bruce Schneier: The Security Mindset

w9.4- Bruce Schneier: The Security Mindset.mp4 — Haruna Media Player

Subtitle scale: 0.5

Bruce Schneier

Resilient Systems

now because it has something  
that nothing else has.

00:00:19 / 00:07:32

65

This video frame shows Bruce Schneier, a man with a white beard and glasses, wearing a blue suit jacket over a pink shirt, gesturing with his hands while speaking. He is positioned in front of a dark wooden bookshelf filled with books. A subtitle at the bottom left identifies him as 'Bruce Schneier' and 'Resilient Systems'. Below that, a caption reads 'now because it has something that nothing else has.' The video player interface at the bottom includes a subtitle scale indicator ('Subtitle scale: 0.5'), a progress bar showing '00:00:19 / 00:07:32', and a page number '65'.

## Summary

### **The Adversarial Mindset: Magnitude 1,000,000**

Computer security is unique because it is the only field with an **adversary relationship**. Unlike building a bridge or a standard OS, in security, someone is actively trying to thwart your Logic-Sync.

#### 1. Thinking "Outside the Box" (The Pi Test)

Schneier shares a classic pedagogical exploit:

- **The Task:** Write down the first 1,000 digits of Pi.
- **The Logic:** It's impossible to memorize. The assignment implicitly requires you to **cheat**.
- **The Catch:** If you get caught, you fail.
- **The Lesson:** Security is a mindset. You don't look at how to build a system; you look at how to make it **fail precisely** to do the damage you want.

## 2. Security as a Social & Technical Discipline

Security isn't just code; it involves:

- **Psychology & Law:** Understanding human behavior and the legal consequences of "hacking."
- **Trust:** Using security tools as a way to enable trust in a society of "Liars and Outliers."
- **Domains:** Whether it's your future IoT Rover, a voting booth, or a self-driving car, the principles of embedded code interacting with society remain the same.

### 3. Career & Calling

- **Demand/Supply:** The demand for security experts is Magnitude 1,000,000, greatly outstripping the supply.
- **Open-Ended Exploration:** Schneier encourages following your passion—be it forensics, cryptography, or SCADA systems. Once you learn "how to think," the technical specifics (like VPNs) are easily mastered.

#### 4. Recommended Audit List (Schneier's Books)

Book

Focus

**Secrets and Lies**

Basic concepts of network security thinking.

**Cryptography Engineering**

How to build (and break) cryptosystems.

**Liars and Outliers**

Security as a social enabler for trust.

**Data and Goliath**

The reality of mass surveillance and privacy.