Unit #4

MIS5214

Case Study 1 and Cryptography

Agenda

- Case Study 1
- Cryptography terminology
- Symmetric Key Cryptography
  - Symmetric stream cryptography
  - Symmetric block cryptography
- Key sharing problem
- Public Key Cryptography
  - Diffie-Hellman algorithm: symmetric key generation through asynchronous cryptography
  - RSA algorithm
- Hybrid-Cryptography
  - Perfect Forward Secrecy
- Where do cryptographic controls go in the FedRAMP System Security Plan
- Brief review: Hashing & Digital Signatures

Case 1 Exercise: Draw a logical network diagram for the Titan System - adding in graphical elements that illustrate the system boundary, interconnections and data flow

You can use: https://app.diagrams.net/ , another graphic drawing program, or PowerPoint

Use your drawing to help you answer the following questions:

1. What was the specific attack vector(s) used by attacker?

2. What failings existed in the following areas?
   - IT Governance
   - End users
   - Information technology services
   - Information security
   - Incident response

What should Margrete Raaum do next?

	Preventive Controls	Detective Controls	Corrective/Responsive Controls
IT Governance			
Incident Response			
Information Security			

Draw a logical network diagram with graphical elements illustrating the system boundary, interconnections and data flow

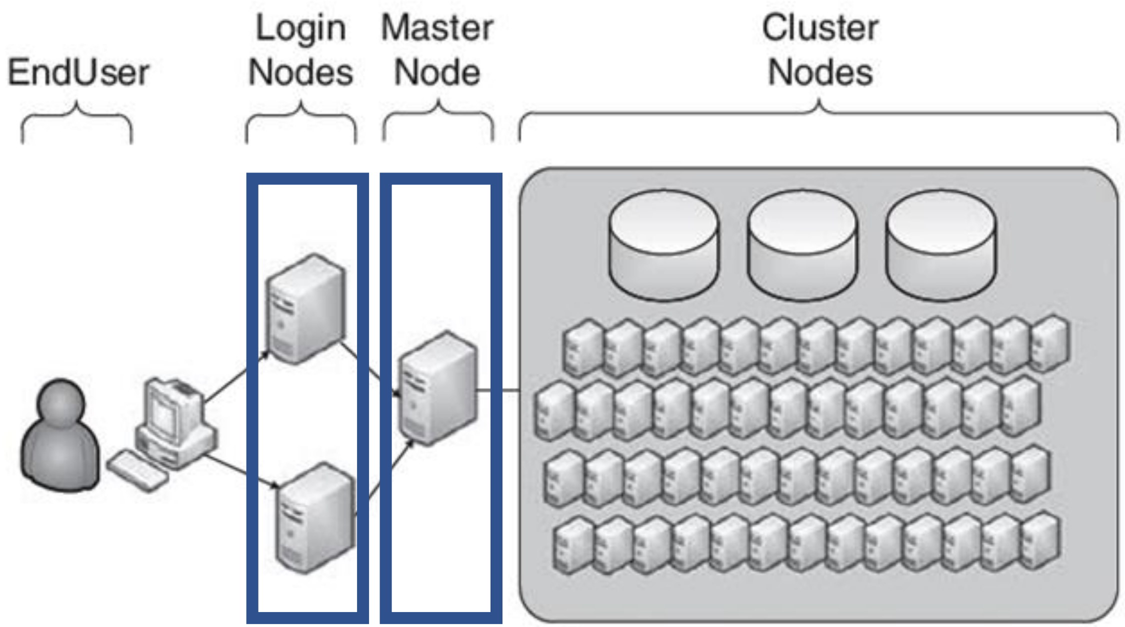


Figure 1 A typical grid architecture.

A High Performance Computing Cluster Under Attack: The Titan Incident

<u>What was the specific attack vector(s) used by attacker:</u>

1. ?
2. ?
3. ?
4. ?
5. ?
6. ?

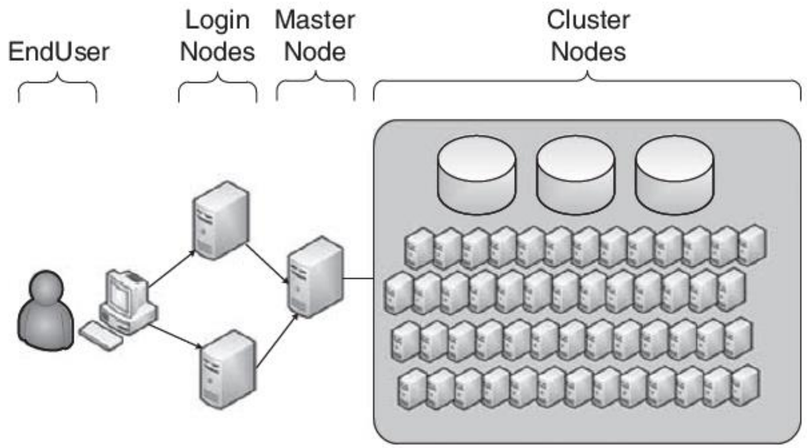


Figure 1 A typical grid architecture.

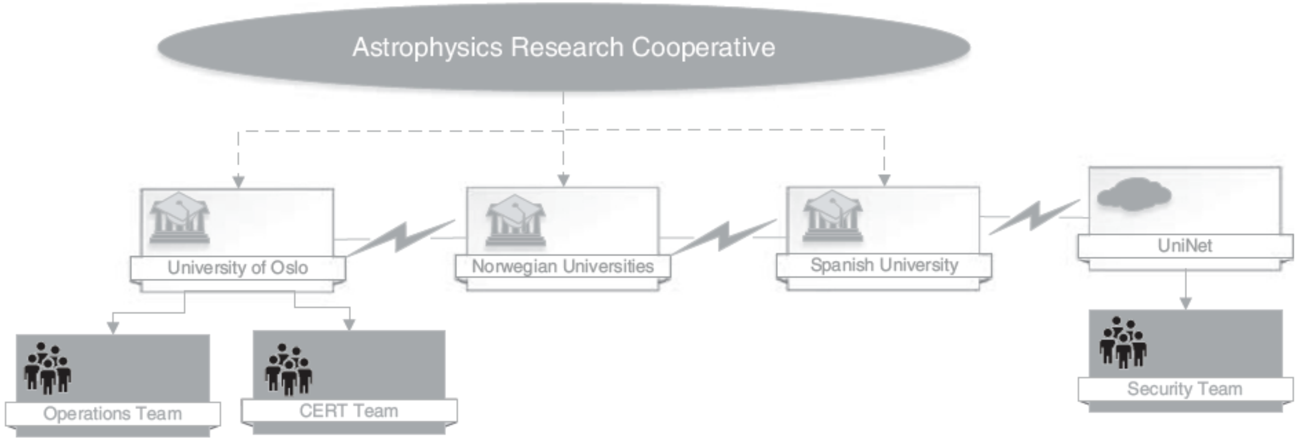


Figure 2 Geographic dispersion of Nordic DataGrid facility Tier-1 clusters.

A High Performance Computing Cluster Under Attack: The Titan Incident

Specific attack vector used by attacker:

1. Attacker obtained valid usernames and password combinations from a system in Spain that had a research agreement with University of Oslo (UiO)
2. Attacker accessed the Titan cluster as a research user using the valid credentials that were "harvested" from the previously compromised system in Spain
3. The attacker used a local Linux system exploit (CVE-2010-3847 GNU C library dynamic linker – '$ORIGIN' Expansion) to gain root administrative privileges on the Titan system (https://cwe.mitre.org/data/definitions/59)
4. Once administrative privileges were obtained, the attacker modified the SSH system files to collect the usernames and passwords of other end-users as they accessed the grid
5. The attacker created at least one "backdoor", or method of accessing the system without relying on the compromised accounts
6. The newly stolen credentials were used to gain unauthorized access to other systems, and they may also have been sold on the black market

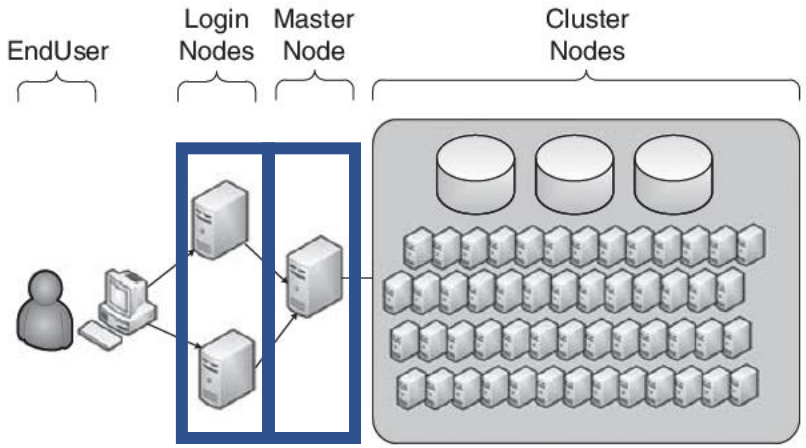


Figure 1 A typical grid architecture.

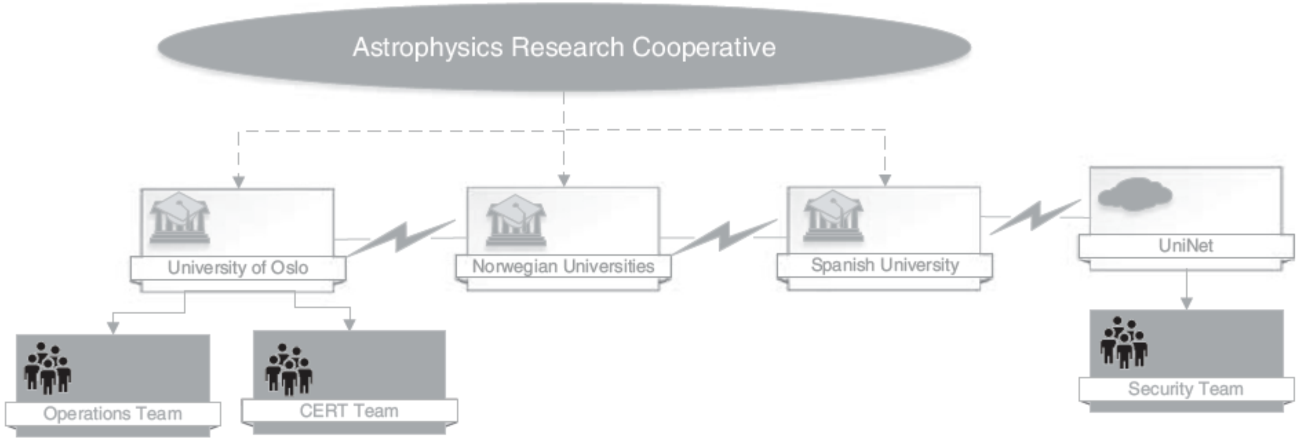


Figure 2 Geographic dispersion of Nordic DataGrid facility Tier-1 clusters.

...the attacker modified SSH system files to collect the usernames and passwords of other end-users as they accessed the grid

Secure Shell (SSH) is
a cryptographic network protocol for operating network services securely over an unsecured network

Common applications include remote command-line login and remote command execution, but any network service can be secured with SSH.

The most visible application of the protocol is for access to shell accounts on Unix-like operating systems, but is in limited use on Windows as well

What should Margrete Raaum do next?

	Preventive Controls	Detective Controls	Corrective/Responsive Controls
IT Governance			
Incident Response			
Information Security			

What should Margrete Raaum do next?

	Preventive Controls	Detective Controls	Corrective/Responsive Controls
IT Governance	• Analyze stakeholder needs and risks associated with not meeting those needs • Educate staff and end users in security policies	• IT security audit of high-risk security processes • Security risk awareness	• Stakeholder communication plan • Response plan

What should Margrete Raaum do next?

	Preventive Controls	Detective Controls	Corrective/Responsive Controls
Incident Response	• Policy and procedure creation • Evaluate incident management capability	• Proactive monitoring and detection • Reactive detection and follow-up	• Investigate and respond (technical, managerial, and legal) • Incident reporting guidelines
IT Governance	• Analyze stakeholder needs and risks associated with not meeting those needs • Educate staff and end users in security policies	• IT security audit of high-risk security processes • Security risk awareness	• Stakeholder communication plan • Response plan

What should Margrete Raaum do next?

	Preventive Controls	Detective Controls	Corrective/Rsponsive Controls
Information Security	• Vulnerability scanning and Operating System (OS) Patching • Principle of least privilege • Multifactor-factor Authentication • Penetration tests	• File system integrity checks • Network IDS/IPS	• Review logs • Forensic analysis • Malware analysis
Incident Response	• Policy and procedure creation • Evaluate incident management capability	• Proactive monitoring and detection • Reactive detection and follow-up	• Investigate and respond (technical, managerial, and legal) • Incident reporting guidelines
IT Governance	• Analyze stakeholder needs and risks associated with not meeting those needs • Educate staff and end users in security policies	• IT security audit of high-risk security processes • Security risk awareness	• Stakeholder communication plan • Response plan

Agenda

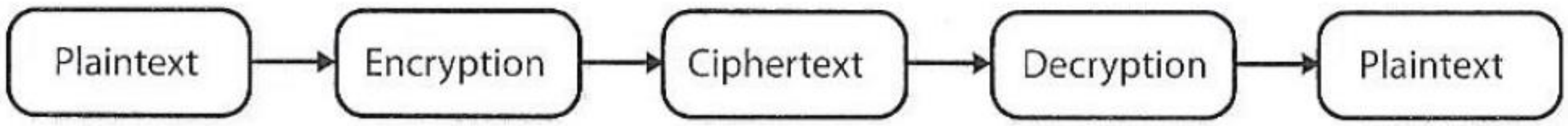
✓Team Project – It is not too early to get started…

✓Case Study 1

- Cryptography terminology
- Symmetric Key Cryptography
  - Symmetric stream cryptography
  - Symmetric block cryptography
- Key sharing problem
- Public Key Cryptography
  - Diffie-Hellman algorithm: symmetric key generation through asynchronous cryptography
  - RSA algorithm
- Hybrid-Cryptography
  - Perfect Forward Secrecy
- Where do cryptographic controls go in the FedRAMP System Security Plan
- Brief review: Hashing & Digital Signatures

Terminology

- **Plaintext** – is the readable version of a message
- **Ciphertext** – is the unreadable results after an encryption process is applied to the plaintext
- **Cryptosystem** – includes all the necessary components for encryption and decryption
  - Algorithms
  - Keys
  - Software
  - Protocols



Harris, S. and Maymi, F. (2016) All-In-One CISSP Exam Guide, McGraw Hill Education

Remember: Services of cryptosystems

- **Confidentiality** – Renders information unintelligible except by authorized entities
- **Authentication** – Verifies the identity of the user or system that created, requested or provided the information
- **Nonrepudiation** – Ensure the sender cannot deny sending the information
- **Integrity** – Data has not been altered in an unauthorized manner since it was created, transmitted, or stored

Cipher = encryption algorithm

2 main attributes combined in a cypher

1. **Confusion:** usually carried out through substitution

2. **Diffusion:** Usually carried out through transposition

Harris, S. and Maymi, F. (2016) All-In-One CISSP Exam Guide, McGraw Hill Education

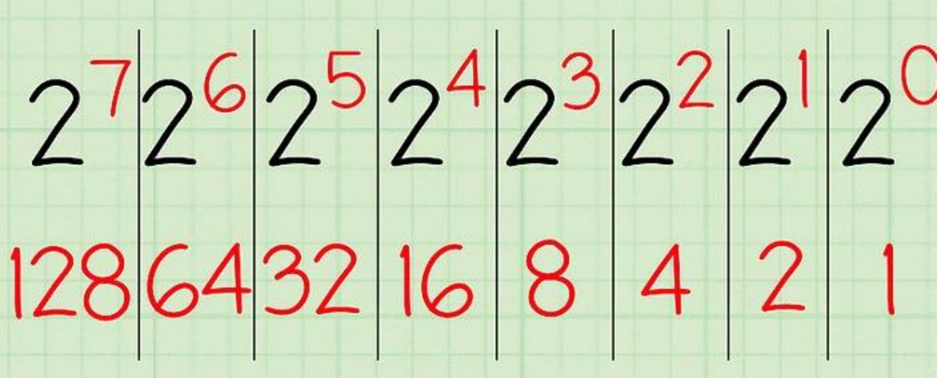
The translation of what we type into ASCII, and then into binary is what is sent in data packets across the network to other computers…

Binary – Decimal

0 0 0 0 0 0 0 0 = 0

1 1 1 1 1 1 1 1 = 255

8 bits supports 256 numbers



ASCII - Decimal

Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e

ASCII Character Table

Name	Hex	Dec
. (period)	2E	046
0	30	048
1	31	049
2	32	050
3	33	051
4	34	052
5	35	053
6	36	054
7	37	055
8	38	056
9	39	057

Name	Hex	Dec
A	41	065
B	42	066
C	43	067
D	44	068
E	45	069
F	46	070
G	47	071
H	48	072
I	49	073
J	4A	074
K	4B	075

Name	Hex	Dec
L	4C	076
M	4D	077
N	4E	078
O	4F	079
P	50	080
Q	51	081
R	52	802
S	53	083
T	54	084
U	55	085
V	56	086

Name	Hex	Dec
W	57	087
X	58	088
Y	59	089
Z	5A	090

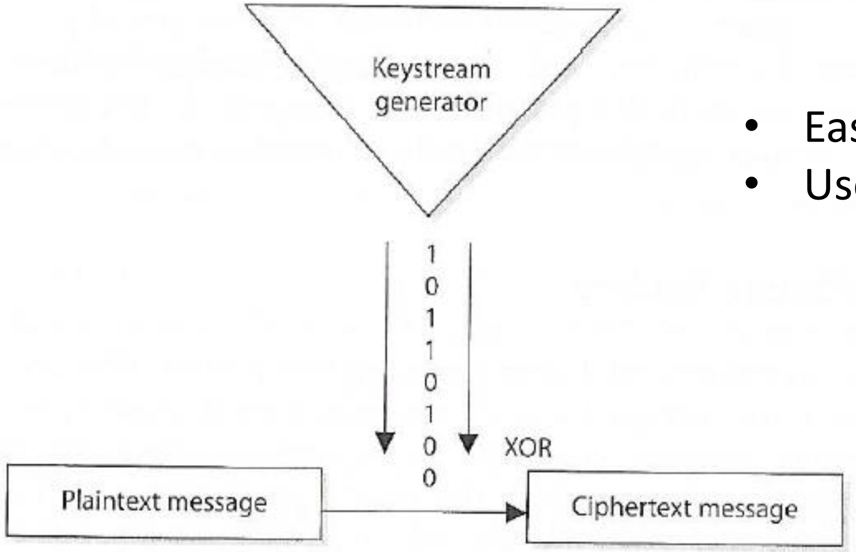
XOR – Exclusive OR

Creating "confusion" through substitution with a binary mathematical function called "exclusive OR", abbreviated as XOR

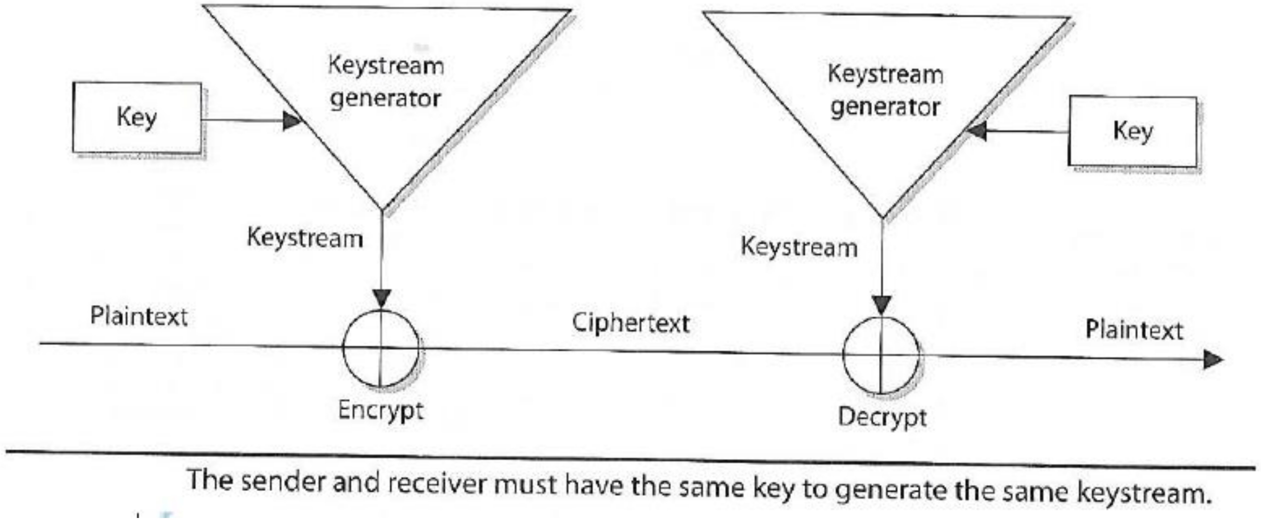
Message stream:	1001010111
Keystream:	0011101010
Ciphertext stream:	1010111101

Harris, S. and Maymi, F. (2016) <u>All-In-One CISSP Exam Guide</u>, McGraw Hill Education

Symmetric Stream Ciphers



- Easy to implement in hardware
- Used in cell phones and Voice Over Internet Protocol



Harris, S. and Maymi, F. (2016) All-In-One CISSP Exam Guide, McGraw Hill Education

Modern Block Ciphers

- Use block sizes of 128-bits or greater
  - Examples of Block Ciphers that can be used are:
    - AES (NIST's 2001 Advanced Encryption Standard – originally known as Rijndael)
      - 128 bit block size, but 3 different key lengths: 128, 192, and 256 bits
    - Blowfish
    - Twofish
    - Serpent
- Do not use these examples of block ciphers which have a 56 bit key length, which is too small to provide secure encryption:
  - DES (Data Encryption Standard)
  - 3DES

Block Ciphers with an S-box

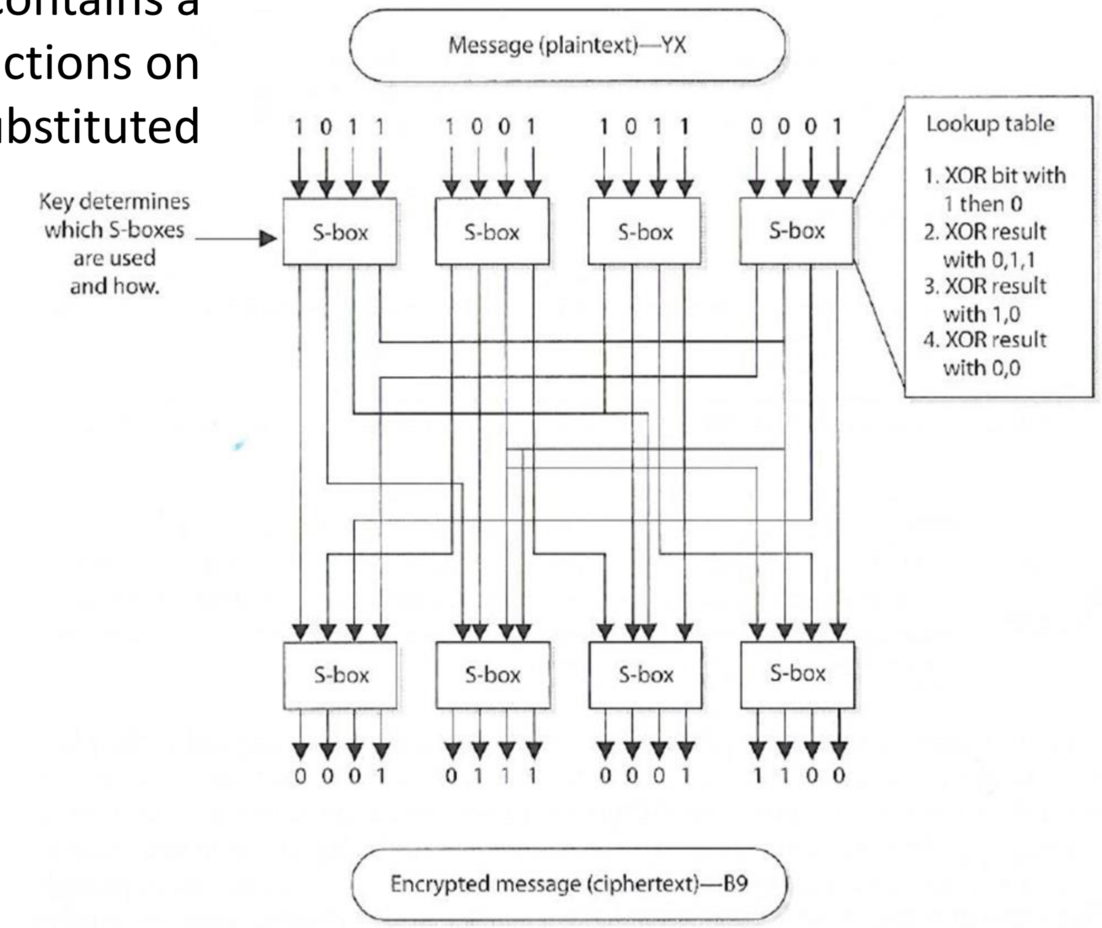
In this example, each Substitution Box (S-box) contains a lookup table used by the algorithm as instructions on how the bits are substituted

Plaintext	Ciphertext
0000	1110
0001	0100
0010	1101
0011	0001
0100	0010
0101	1111
0110	1011
0111	1000
1000	0011
1001	1010
1010	0110
1011	1100
1100	0101
1101	1001
1110	0000
1111	0111

Encryption table

Ciphertext	Plaintext
0000	1110
0001	0011
0010	0100
0011	1000
0100	0001
0101	1100
0110	1010
0111	1111
1000	0111
1001	1101
1010	1001
1011	0110
1100	1011
1101	0010
1110	0000
1111	0101

Decryption table



…followed by transposition…

Block cipher's "mode of operation"

5 modes of operation are used to tailor them for use in different applications:

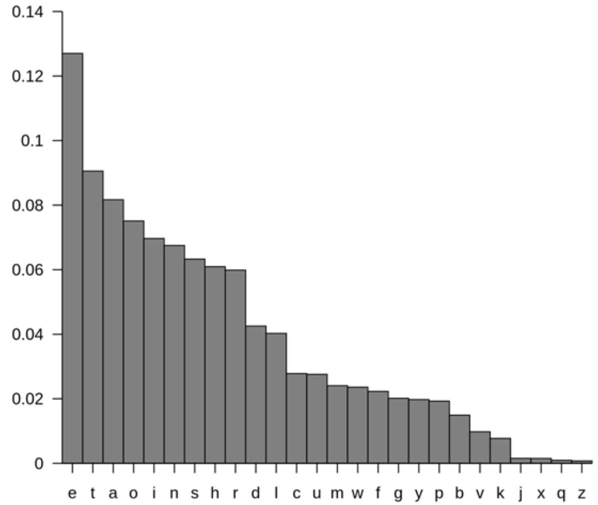
1. ECB – Electronic Code Book mode
2. CBC – Cipher Block Chaining mode
3. CFB – Cipher FeedBack mode
4. OFB – Output FeedBack mode
5. CTR – CounTeR mode

ECB – Electronic Code Book mode

- A data block of a certain size (e.g. 64 bits or 128 bits or…) is entered into the algorithm with the key, and a block of cipher text is produced

$C_i = Encrypt(Key, P_i)$
for $i = 1, ..., k$

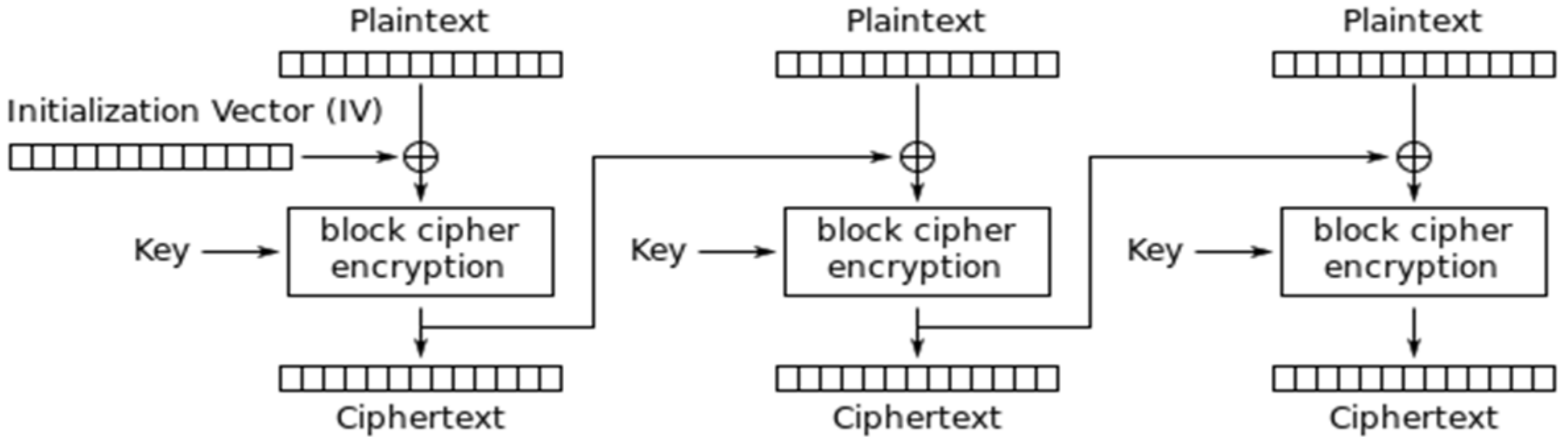
Where:
- *Ci is block i of ciphertext*
- *P, is a block of plaintext*



- Encrypts every block <u>the same way every time for a given key</u>
- Why is this a problem?
  - ➢ This is a problem because **frequency analysis** of the encrypted text can reveal a lot of information
  - ➢ Not enough randomness

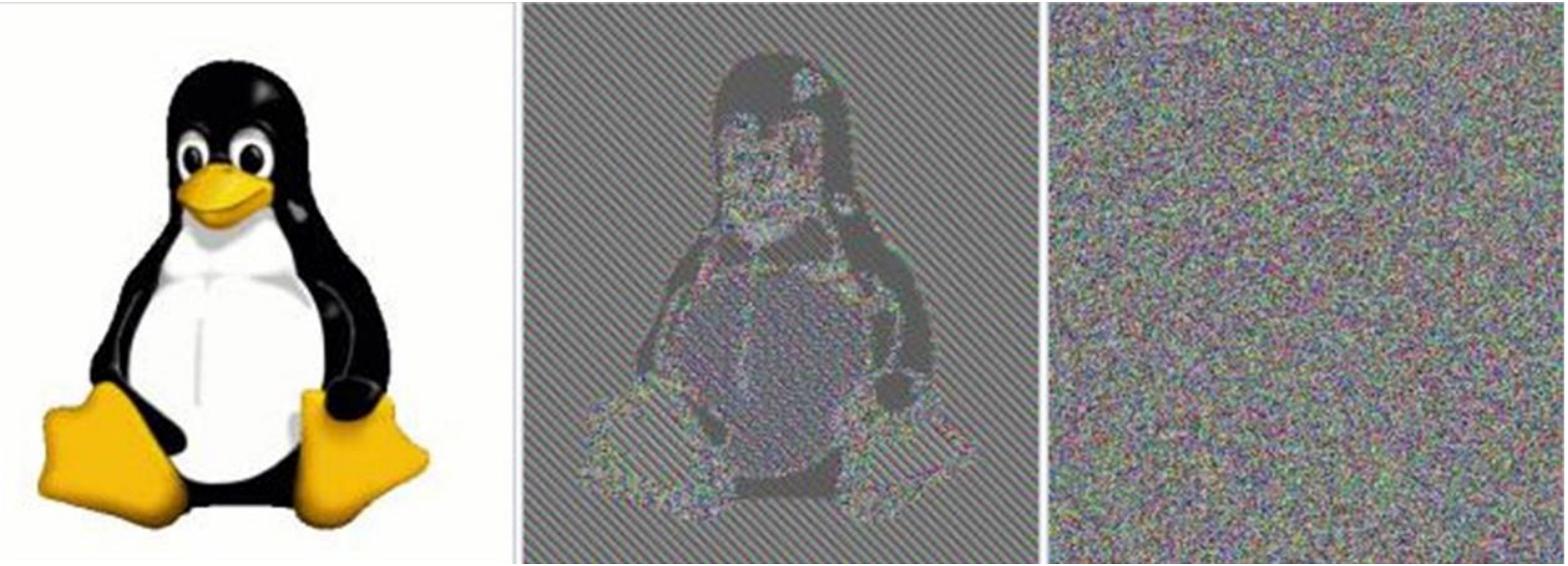
CBC – Cipher Block Chaining mode

- Is much more secure
- Does not reveal a pattern of encryption for frequency analysis
- Each block of text, the key, and the value based on the previous block are processed in the algorithm and applied to the next block of text



- XORs a plaintext with the **last** encrypted block before encrypting it. This ensures that the same plaintext is encrypted differently every time.
- Requires an initialization vector (or IV) to get started, since the first block doesn't have a previous encrypted block to XOR against.

Visa



Original Image

Block cipher with ECB (Electronic Code Book) encryption

Not good!

Block cipher with CBC (Cipher Block Chaining) or any of the other modes of encryption

These are good!

Cryptanalysis Attacks

- Brute force
  - Trying all key values in the keyspace
- Frequency Analysis
  - Guess values based on frequency of occurrence
- Dictionary Attack
  - Find plaintext based on common words
- Known Plaintext
  - Format or content of plaintext available
- Chosen Plaintext
  - Attack can encrypt chosen plaintext
- Chosen Ciphertext
  - Decrypt known ciphertext to discover key
- Random Number Generator (RNG) Attack
  - Predict initialization vector used by an algorithm
- Social Engineering
  - Humans are the weakest link

Modern Block Ciphers

Use block sizes of 128-bits or greater

- Examples of Block Ciphers that can be used are:
  - AES
  - Blowfish
  - Twofish
  - Serpent

Do not use these examples of block ciphers which use 64-bit blocks, which are too small to be secure include:

- DES
- 3DES

Agenda

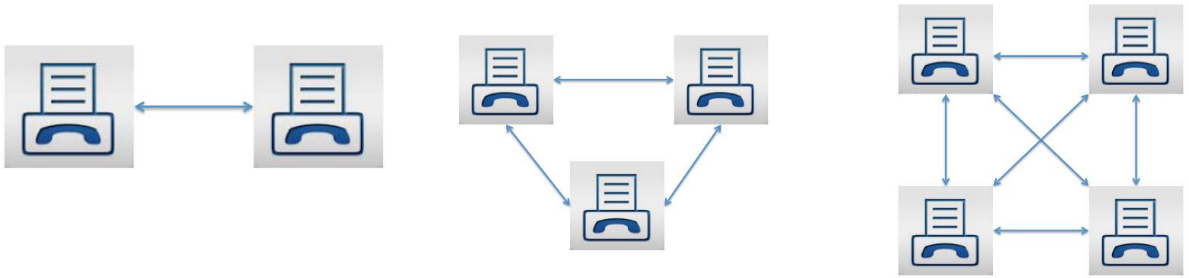
✓Team Project – It is not too early to get started…

✓Case Study 1

✓Cryptography terminology

- Review Symmetric Key Cryptography
  - Symmetric stream cryptography
  - Symmetric block cryptography
- Key sharing problem
- Public Key Cryptography
  - Diffie-Hellman algorithm: symmetric key generation through asynchronous cryptography
  - RSA algorithm
- Hybrid-Cryptography
  - Perfect Forward Secrecy
- Where do cryptographic controls go in the FedRAMP System Security Plan
- Brief review: Hashing & Digital Signatures

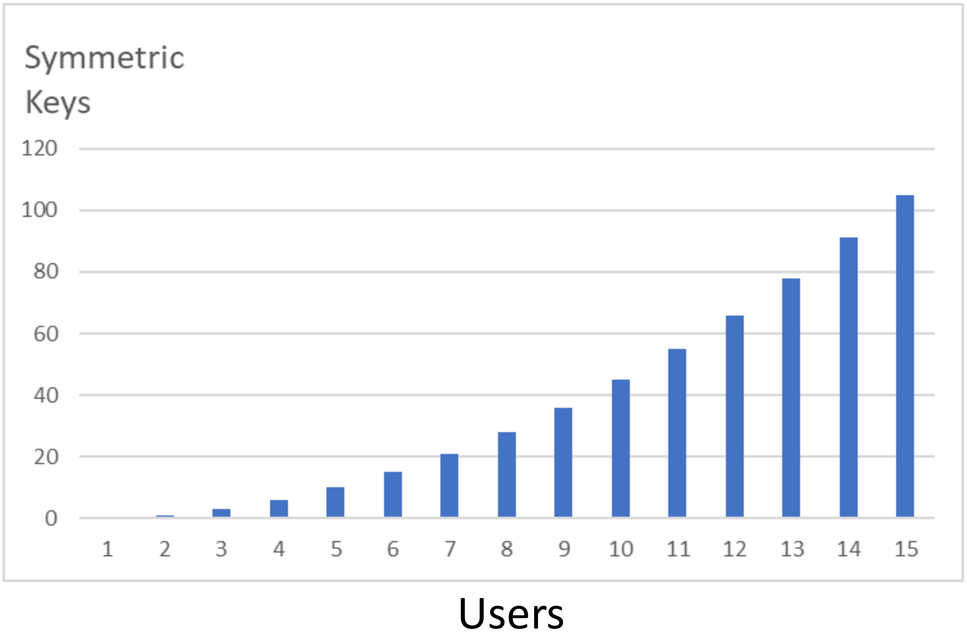
Key sharing problem



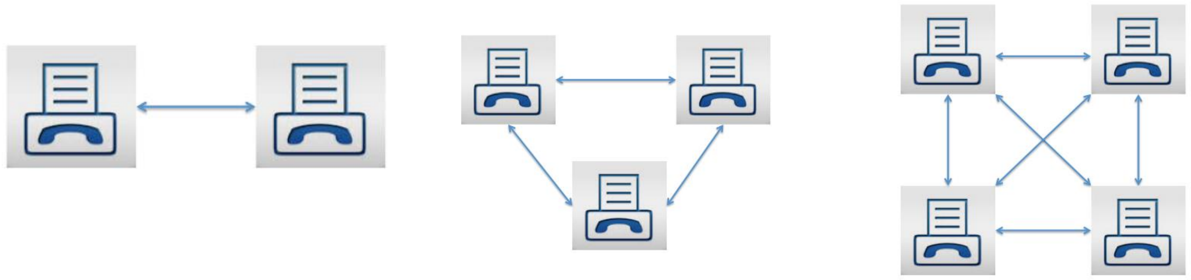
Sharing cryptographic keys has been a problem throughout history

- The number of pairs of keys (“secure network connections”) grows at a near exponential rate (i.e. geometric rate) as the number of users increases

Users	Symmetric Keys
1	0
2	1
3	3
4	6
5	10
6	15
7	21
8	28
9	36
10	45
11	55
12	66
13	78
14	91
15	105
…	…



Key sharing problem



- The number of pairs of keys needed for “n” users is determined by an equation known as Metcalf’s Law
- Number of key pairs needed for n users = (n*(n-1))/2
  - *The reason for the n-1 is that you do not need to communicate with yourself*
- For MIS 4596 with 22 students how many keys would we need:

$$(22 * 21)/2 = 231 \text{ keys}$$

Diffie-Hellman Algorithm: *Secret symmetric key derivation through public key sharing*

Assumptions:

A prime number is a positive whole number whose only factors (i.e. integer divisors) are 1 and the number itself (e.g. 2, 3, 5, 7, 11, 13, 17, 19, 23, …). Bob & Alice want to compute a shared secret key to protect confidentiality of their conversation. Eve eavesdrops…

Algorithm:

1. Bob & Alice publicly agree on "**p**" called *prime modulus* (e.g. **p = 23**) & "**g**" called *generator* (e.g. **g = 5**), Eve overhears
2. Bob & Alice each choose their own secret key:
   - Bob's secret key is referred to as "**x_bob**" which is a number between 1 and p-1 (e.g. **x_bob = 12**)
   - Alice's secret key is referred to as "**x_alice**" which also is a number between 1 and p-1 (e.g**. x_alice = 7**)
3. Bob & Alice each computes their own public key, which they share with each other and Eve intercepts…
   - Bob computes: **y_bob =** g^{x_bob} **mod p** which is: **y_bob =** 5^{12} **mod 23 = <u>18</u>** which he shares with Alice (and Eve)
   - Alice computes: **y_alice =** g^{x_alice} **mod p** which is: **y_alice =** 5^{7} **mod 23 = <u>17</u>** which she shares with Bob (and Eve)
4. Bob & Alice each compute their shared secret symmetric key
   - Bob computes: $y\_alice^{x\_bob}$ **mod p** which is: 17^{12} **mod 23 = 6**
   - Alice computes: $y\_bob^{x\_alice}$ **mod p** which is: 18^{7} **mod 23 = 6**
5. Bob & Alice now have a **shared secret ("symmetric") key = 6**
6. Eve has Bob & Alice's public keys: y_bob=18 & y_alice=17, prime modulus: p=23 and generator: g=5, but not their secret keys x_bob = 12 & x_alice = 7
   - ***Eve cannot calculate Bob& Alice's shared symmetric secret key from their public keys, p and g alone – even though she knows they are using the Diffie-Hellman algorithm!***

let's use WolframAlpha online calculator

In practice, *p* must be much larger prime number… this is a 4096-bit *p*

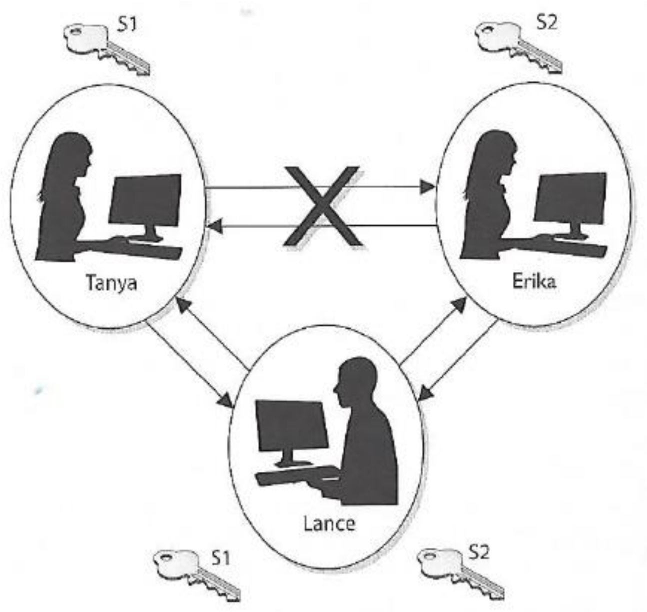
857,756,147,438,808,767,721,482,523,862,479,196,091,217,066,271,200,126,894,701,702,329,327,8
72,802,487,425,224,246,373,206,756,773,954,180,315,945,664,685,564,049,690,107,228,861,210,05
3,005,306,168,041,237,244,792,245,832,497,260,206,801,417,396,745,674,574,281,768,112,711,519,
809,332,223,737,878,554,093,201,446,763,995,425,025,965,323,912,149,043,161,823,975,594,943,9
15,411,109,637,902,372,642,611,214,196,649,667,036,726,005,577,041,694,781,738,635,943,018,156
,362,403,714,091,905,448,620,990,965,500,814,912,289,738,636,687,051,381,358,564,729,963,735,7
82,176,280,511,819,070,673,927,579,180,484,836,950,910,945,840,410,470,935,832,100,360,510,117
,962,261,152,920,101,946,255,789,679,435,711,472,267,368,823,730,863,971,596,718,223,674,224,1
06,003,985,209,174,353,308,077,140,794,884,546,003,360,030,727,697,326,025,663,819,442,780,10
5,880,604,943,197,516,223,343,068,846,392,924,237,875,653,640,416,933,764,628,191,065,601,980,
281,442,005,263,033,849,543,723,716,743,986,123,624,356,871,152,793,177,027,462,801,070,011,5
26,783,269,474,338,816,734,553,122,757,257,382,121,230,562,181,721,318,331,271,107,036,972,78
8,062,816,322,387,506,944,045,038,739,178,684,349,474,317,534,892,731,313,651,324,179,101,369,
222,316,429,969,662,605,450,068,078,088,031,941,042,867,503,697,721,512,539,949,128,099,005,1
60,179,345,242,776,041,458,121,259,813,719,561,319,392,760,414,249,584,984,440,063,314,771,03
9,261,920,249,005,444,014,069,555,961,131,639,966,539,872,980,057,279,636,609,441,274,119,014,
567,294,590,620,498,019,375,631,405,622,479,332,810,401,520,856,695,524,524,855,468,645,479,0
42,909,834,183,316,487,318,824,544,358,235,183,243,643

Diffie-Hellman

- Uses asymmetric public and private keys to exchange a symmetric key
- Does not use asymmetric keys for confidentiality (i.e. to encrypt or decrypt any messages)
- Users/systems need to negotiate a new key for every new person
- No authentication, no non-repudiation

Diffie-Hellman is vulnerable to man-in-the-middle attack, because no authentication occurs before public keys are exchanged

1. Tanya sends her public key to Erika, but Lance grabs the key during transmission so it never makes it to Erika
2. Lance spoofs Tanya's identity and sends over his public key to Erika. Erika now thinks she has Tanya's public key
3. Erika sends her public key to Tanya, but Lance grabs the key during transmission so it never makes it to Tanya
4. Lance spoofs Erika's identity and sends over his public key to Tanya. Tanya now thinks she has Erika's public key
5. Tanya combines her private key and Lance's public key and creates a symmetric key S1
6. Lance combines his private key and Tanya's public key and creates symmetric key S1
7. Erika combines her private key and Lance's public key and creates symmetric key S2
8. Lance combines his private key and Erika's public key and creates symmetric key S2
9. Now Tanya and Lance share a symmetric key (S1) and Eriak and Lance share a different symmetric key (S2). Tanya and Erika think they are sharing a key between themselves and od not realize Lance is involved
10. Tanya writes a message to Erika, and uses her symmetric key (S1) to encrypt the message, and sends it
11. Lance grabs the message and decrypts it with symmetric key S1, reads or modifies the message and re-encrypts it with symmetric key S2, and then sends it to Erika
12. Erika take symmetric key S2 and uses it to decrypt and read the message….



Harris, S. and Maymi, F. (2019) <u>All-In-One CISSP Exam Guide</u>

Symmetric versus asymmetric algorithms

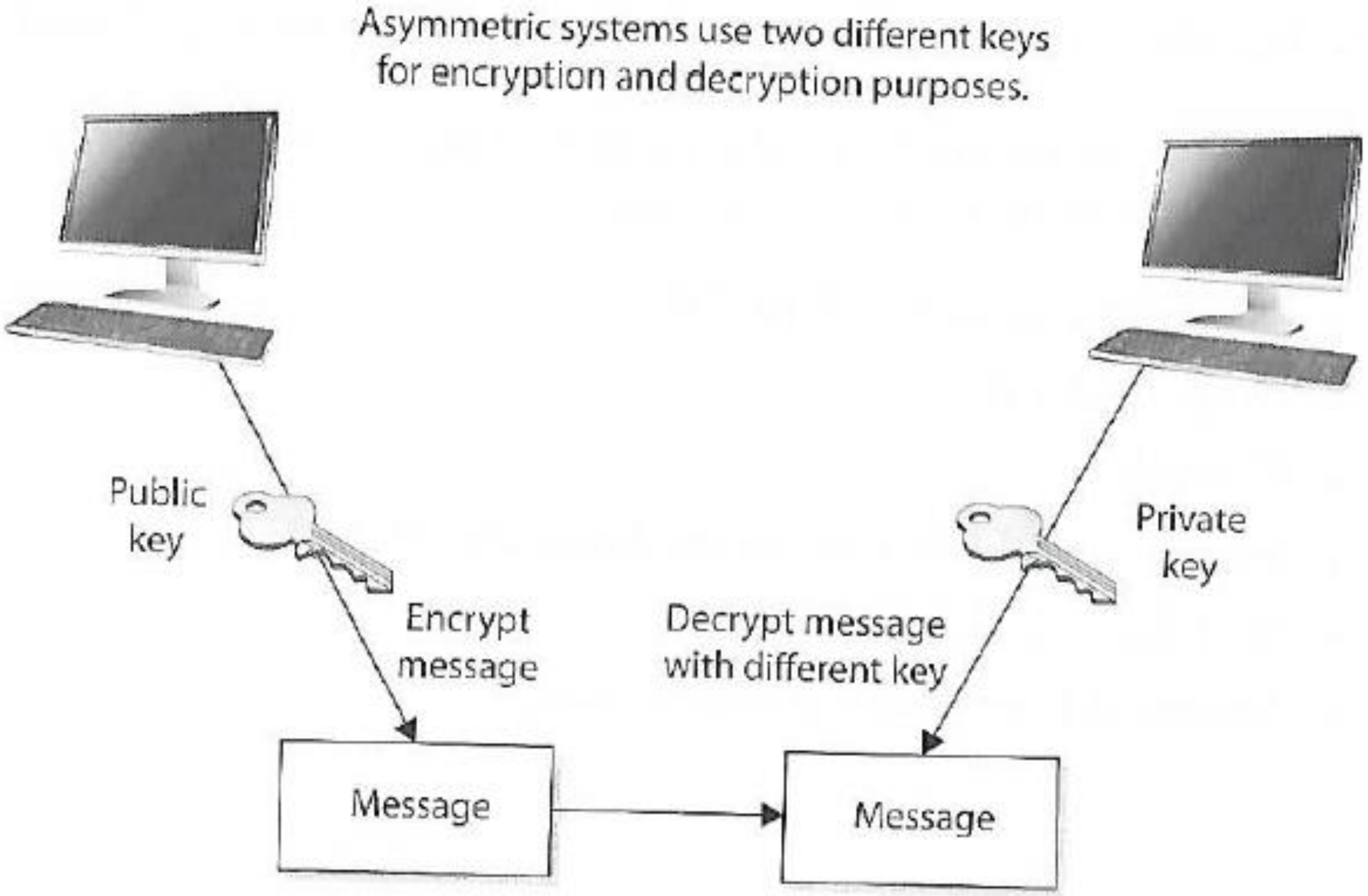
- Symmetric cryptography
  - Use a copied pair of symmetric (identical) secret keys
  - The sender and the receive use the same key for encryption and decryption functions
  - Confidentiality, but no integrity, authentication nor non-repudiation

- Asymmetric cryptography
  - Also know as "public key cryptography"
  - Use different ("asymmetric") keys for encryption and decryption
  - One is called the "private key" and the other is the "public key"
  - Confidentiality, <u>but also want authenticity and non-repudiation</u>

Asymmetric cryptography

- **Public and Private** keys are mathematically related
  - Public keys are generated from private key
  - Private keys cannot be derived from the associated public key (if it falls into the wrong hands)

- **Public key** can be known by everyone
- **Private key** must be known and used only by the owner



Asymmetric cryptography is computationally intensive and much slower (1,000 times slower) than symmetric cryptography

Harris, S. and Maymi, F. (2016) All-In-One CISSP Exam Guide

Quick review

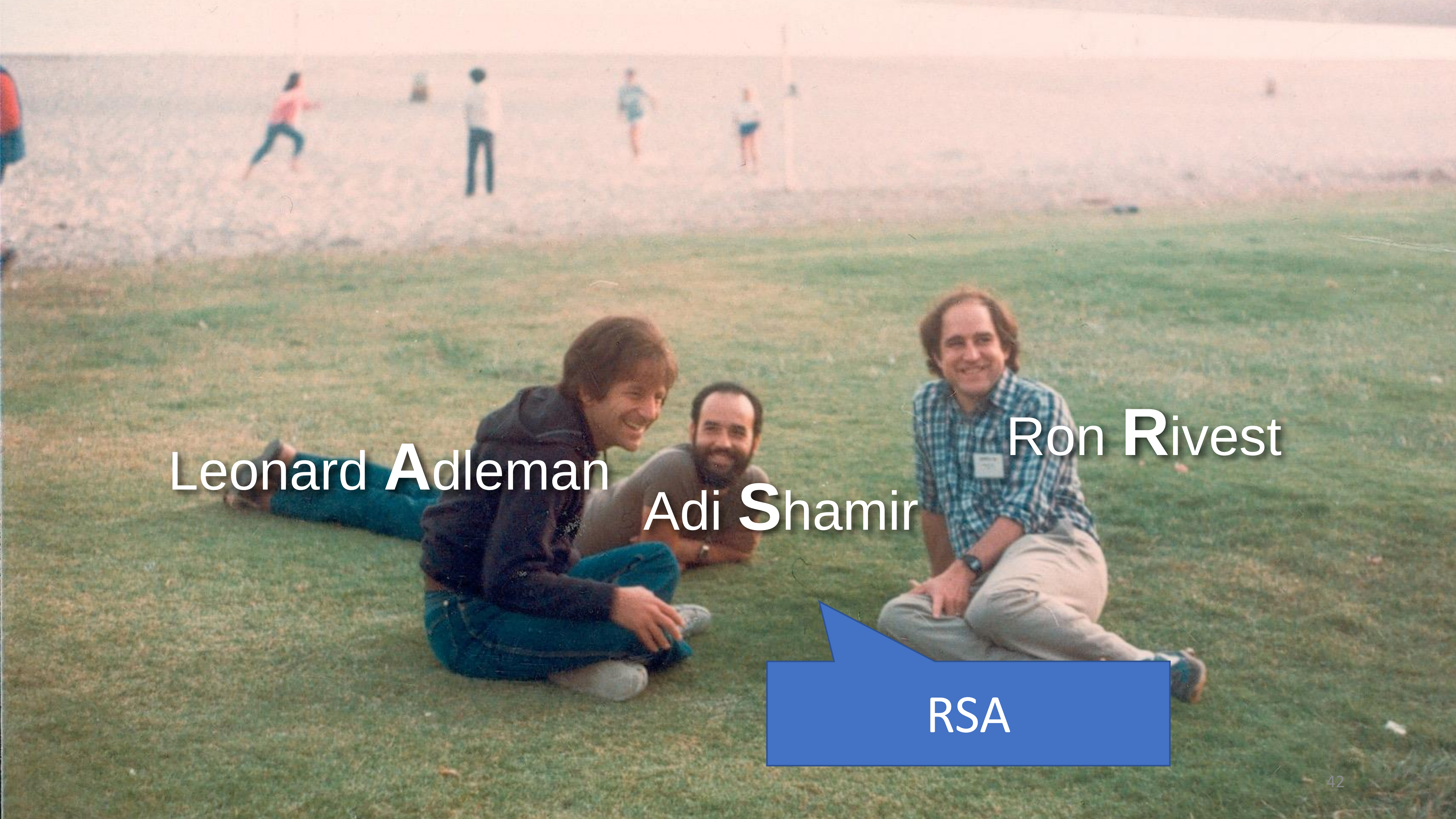
1. If a symmetric key is encrypted with a receiver's public key, what security service is provided?
    - **Confidentiality**: only the receiver's private key can be used to decrypt the symmetric key, and only the receiver should have access to this private key

Quick review

2. If data is encrypted with the sender's private key, what security services are provided?
   - **Authenticity** of the sender and nonrepudiation. If the receiver can decrypt the encrypted data with the sender's public key, then receiver knows the data was encrypted with the sender's private key

Quick review

3. Why do we encrypt the message with the symmetric key rather than the asymmetric key?
   - **Because the asymmetric key algorithm is too slow**


Leonard **A**dleman
Adi **S**hamir
Ron **R**ivest
RSA

RSA Public Key Algorithm

- Most popular worldwide standard, that can be used for:
  - Asymmetric encryption/decryption
  - Key exchange (i.e. used to encrypt AES symmetric key)
  - Digital signatures
- RSA provides:
  - Confidentiality through encryption
  - Authentication and non-repudiation through signature verification

RSA Public Key Algorithm

- Based on factoring large numbers into their prime numbers
  - A prime number is a positive whole number whose only factors (i.e. integer divisors) are 1 and the number itself
    - E.g. 2, 3, 5, 7, 11, 13, 17, 19, 23, …
  - Prime number factoring is
    - Easy when you know the result and one of the factors
      - 6,700,283 = 1889 * X
      - 6,700,283/1889 = X
      - X = 3547
      - 6,700,283 = 1889 * 3547
    - Difficult when you do not know the factors, and the result is large
      - 6,700,283 = prime1 * prime2

912,000,833,142,392,234,931,095,438,312,170,357,695,712,756,726,097,734,441,072,301,836,839,393,353,139,295,831,007,333,431,845,325,988,055,078,535,723,070,121,899,982,515,821,096,513,935,693,429,159,810,068,629,730,360,987,721,191,239,128,388,101,705,884,309,757,897,995,146,963,367,920,258,875,045,283,800,013,428,503,089,286,243,910,365,443,336,583,304,589,741,301,149,906,707,508,832,951,802,034,609,255,816,376,427,847,745,175,505,389,216,575,446,117,214,435,309,308,014,792,888,796,704,735,885,959,753,047,089,134,349,280,135,328,216,026,587,690,550,563,014,619,967,646,165,581,934,916,994,388,164,807,475,497,618,817,178,492,168,759,798,526,076,195,659,132,696,724,374,189,538,701,725,588,364,053,265,311,713,122,599,620,063,110,587,984,125,160,066,509,094,636,495,654,197,043,440,384,099,590,663,387,607,347,763,569,889,588,046,648,769,380,051,353,352,323,215,616,700,132,767,221,738,255,618,066,992,935,073,985,886,089,858,691,117,257,124,338,259,178,666,315,503,726,679,904,506,880,795,225,928,179,249,708,512,521,519,802,379,088,471,059,576,692,488,554,724,378,606,462,675,913,887,571,281,558,908,666,408,509,112,360,978,089,673,490,666,194,566,892,424,767,464,525,985,354,883,620,245,066,389,972,670,528,760,628,056,151,340,458,770,638,783,170,937,336,003,358,144,954,416,252,316,459,167,693,365,704,770,051,596,394,325,584,518,899,185,083,613,743,340,976,318,518,122,032,762,826,960,167,883,646,888,151,502,959,194,155,684,395,680,807,784,172,903,618,731,005,977,092,813,955,195,470,328,083,428,604,222,138,565,171,106,482,154,997,950,843,259,717,191,116,046,110,961,976,117,683,744,708,282,531,877,426,978,230,302,213,288,137,147

$prime_1 * prime_2 =$

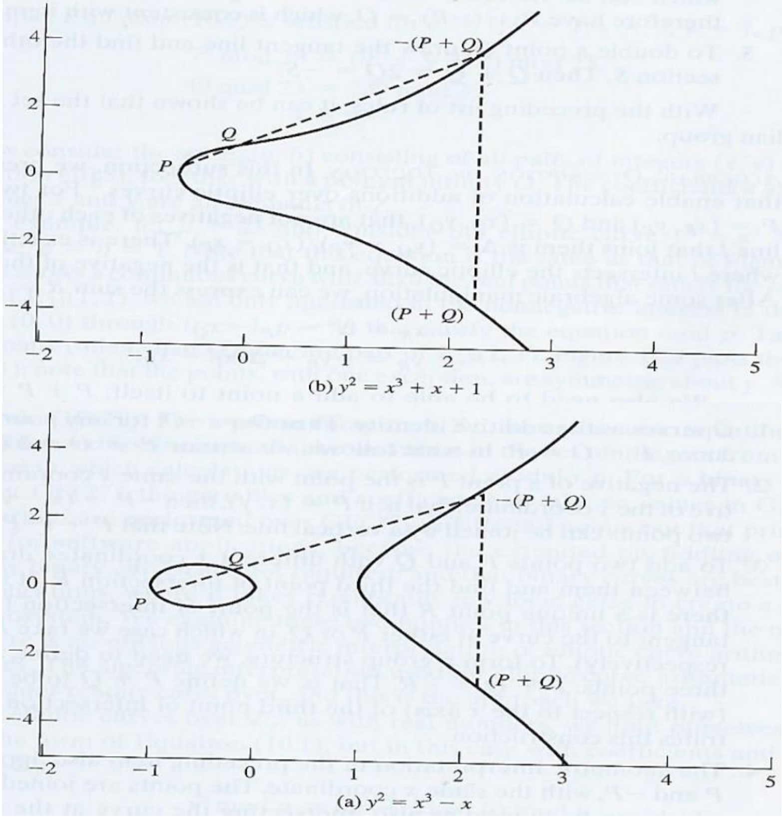
912,000,833,142,392,234,931,095,438,312,
39,393,353,139,295,831,007,333,431,845,3
6,513,935,693,429,159,810,068,629,730,36
995,146,963,367,920,258,875,045,283,800,
89,741,301,149,906,707,508,832,951,802,0
5,446,117,214,435,309,308,014,792,888,79
216,026,587,690,550,563,014,619,967,646,
78,492,168,759,798,526,076,195,659,132,6
3,122,599,620,063,110,587,984,125,160,06
387,607,347,763,569,889,588,046,648,769,
55,618,066,992,935,073,985,886,089,858,6
4,506,880,795,225,928,179,249,708,512,52
606,462,675,913,887,571,281,558,908,666,
24,767,464,525,985,354,883,620,245,066,3
3,170,937,336,003,358,144,954,416,252,31
899,185,083,613,743,340,976,318,518,122,
55,684,395,680,807,784,172,903,618,731,0
8,565,171,106,482,154,997,950,843,259,71
877,426,978,230,302,213,288,137,147

Public Key algorithms

- Fundamental security elements in cryptosystems, applications and protocols
- Assure confidentiality, authenticity and non-repudiation of electronic communications and data storage
- Provide:
  - Key distribution and secrecy (e.g. Diffie–Hellman key exchange)
  - Digital signatures (e.g. Digital Signature Algorithm)
  - Both: key distribution and secrecy and digital signatures (e.g., RSA, ECC)

Elliptic-curve cryptography (ECC)

- Alternate approach to public-key cryptography based on algebraic structure of elliptic curves (based on Galois fields)
- Provides much the same security functionality as Diffie-Hellman and RSA:
  - Encryption/decryption (confidentiality)
  - Secure key distribution (authenticity, confidentiality)
  - Digital signatures (authenticity, non-repudiation)

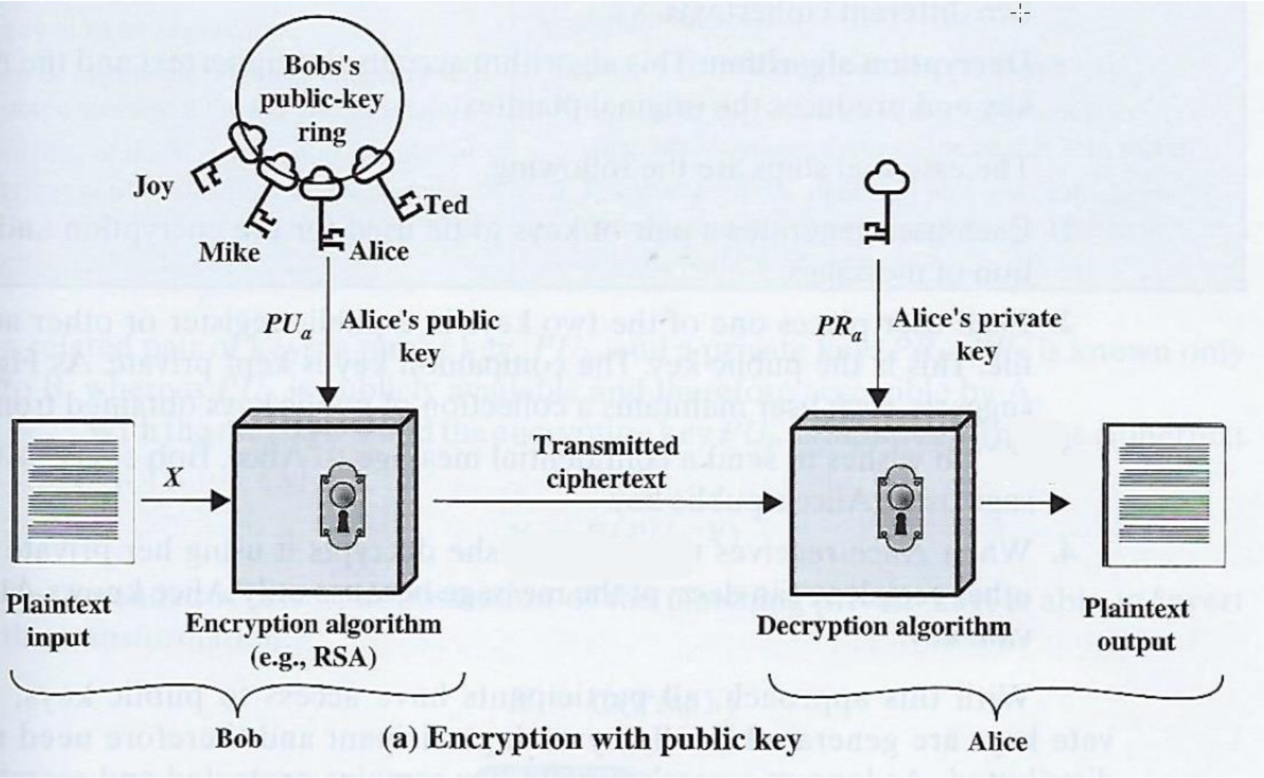


Examples of elliptic curves

- ECC’s is much more efficient than RSA and the other asymmetric algorithms
  - Requires less bits and smaller keys than RSA for achieving the same level of security in its calculations and other algorithms
  - ECC’s efficiency makes it very good for wireless devices and cellular phones with limited processing capacity, storage, power supply and bandwidth

Harris, S. and Maymi F. (2019) All-In-One CISSP Exam Guide

Public Key Management



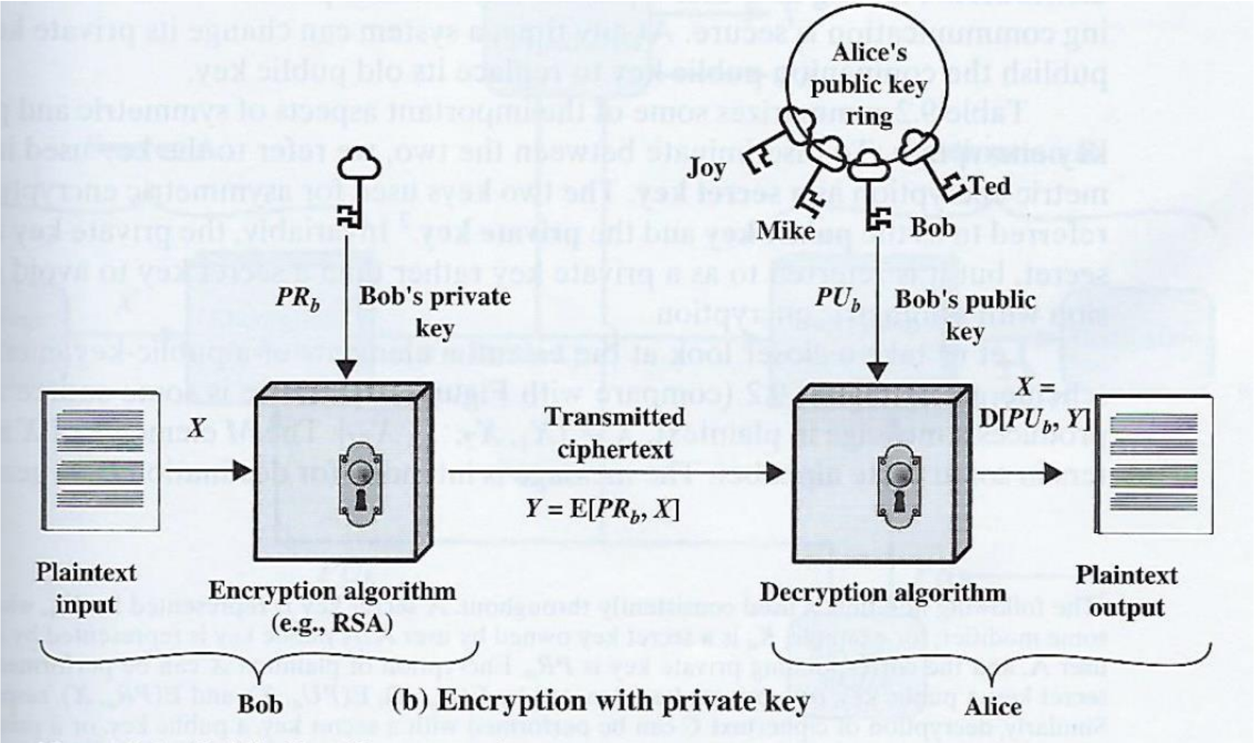


Figure 9.1 Public-Key Cryptography

Stallings, W. (2014) Cryptography and Network Security

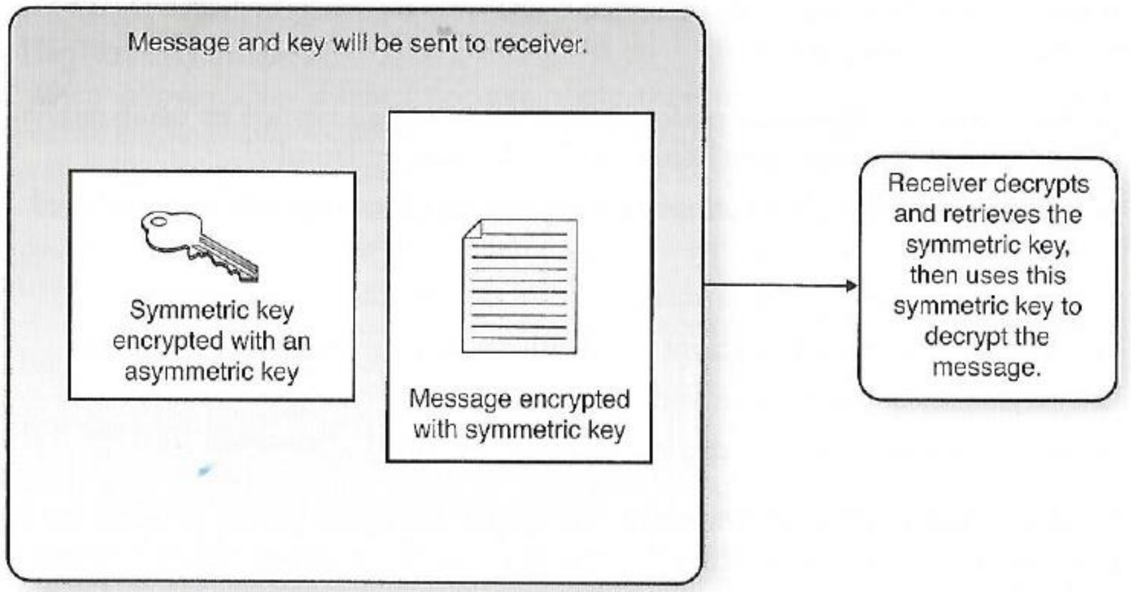
Agenda

- ✓Team Project – It is not too early to get started…
- ✓Case Study 1
- ✓Cryptography terminology
- ✓Symmetric Key Cryptography
  - ✓Symmetric stream cryptography
  - ✓Symmetric block cryptography
- ✓Key sharing problem
- ✓Public Key Cryptography
  - ✓Diffie-Hellman algorithm: symmetric key generation through asynchronous cryptography
  - ✓RSA algorithm
- Hybrid-Cryptography
  - Perfect Forward Secrecy
- Where do cryptographic controls go in the FedRAMP System Security Plan
- Brief review: Hashing & Digital Signatures

Hybrid Encryption (a.k.a. "digital envelope")

Symmetric and asymmetric and algorithms are often used together

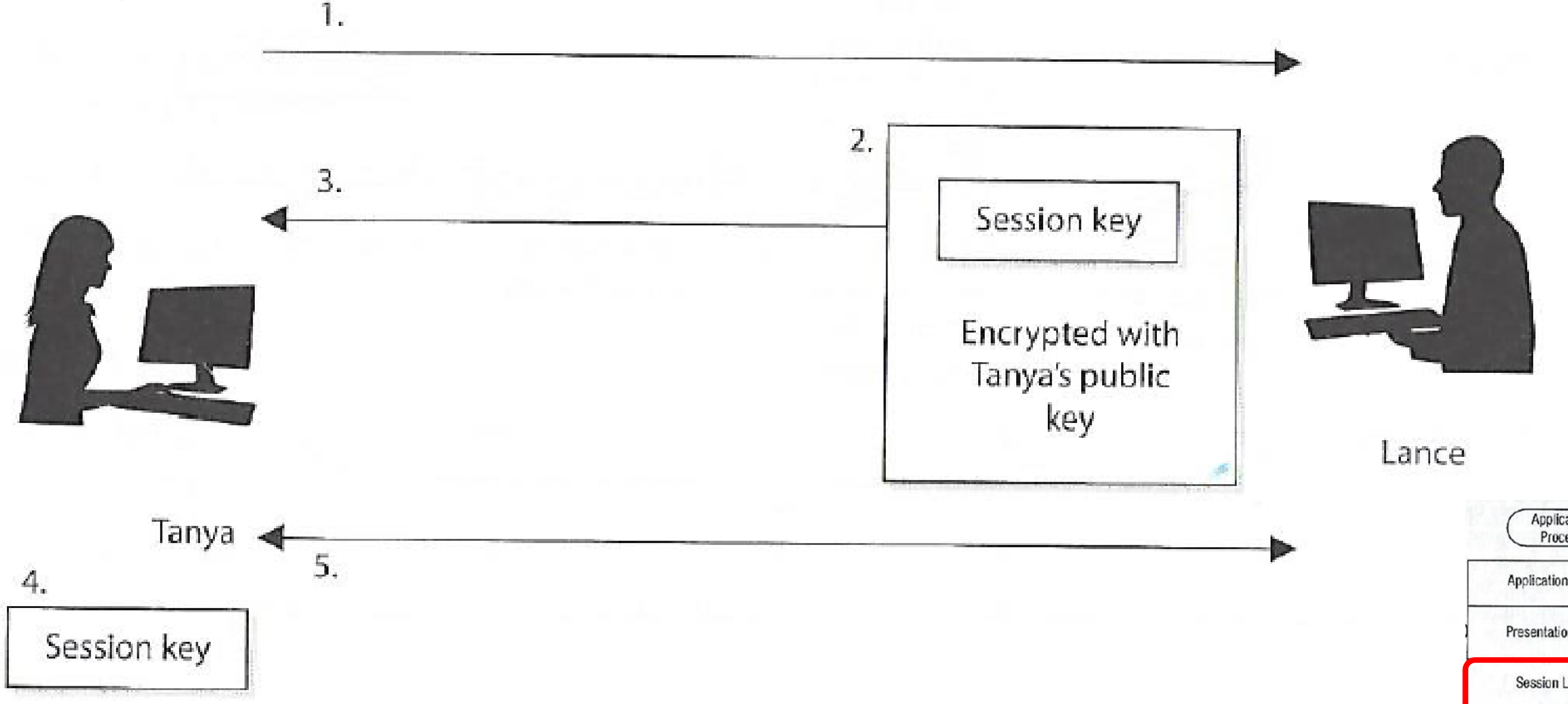
- Public key cryptography's asymmetric algorithm is used to create public and private keys for authentication and confidentiality to help achieve secure automated key distribution
- Symmetric algorithm is used to create secret keys for rapid encryption/decryption of bulk data



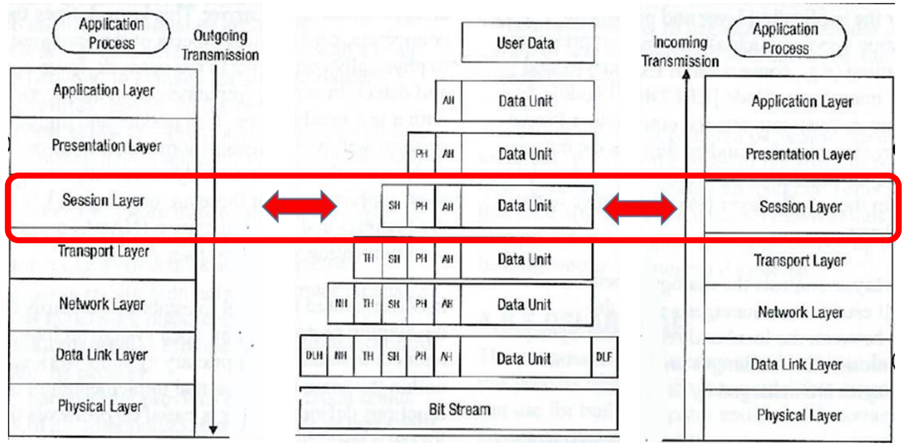
Harris, S. and Maymi, F. (2016) <u>All-In-One CISSP Exam Guide</u>, McGraw Hill Education

Session keys

<u>Single-use</u> symmetric keys used to encrypt messages between two users in an individual communication session



This is how secure web client applications communicate with server-side services



1) Tanya sends Lance her public key.
2) Lance generates a random session key and encrypts it using Tanya's public key.
3) Lance sends the session key, encrypted with Tanya's public key, to Tanya.
4) Tanya decrypts Lance's message with her private key and now has a copy of the session key.
5) Tanya and Lance use this session key to encrypt and decrypt messages to each other.

Harris, S. and Maymi, F. (2016) <u>All-In-One CISSP Exam Guide</u>, McGraw Hill Education

Perfect Forward Secrecy (PFS) or Forward Secrecy (FS)

Designed to prevent the compromise of a long-term secret key from affecting the confidentiality of past conversations

- Protects encrypted data recorded in past sessions against future attacks and compromises of private or secret keys

- Diffie-Hellman and RSA are used together to protect encrypted communications and sessions recorded in the past from being retrieved and decrypted in the future if long-term secret or private keys are compromised in the future

https://www.wired.com/2016/11/what-is-perfect-forward-secrecy/

Example of a simple instant messaging protocol employing forward secrecy:

1. Alice and Bob each generate a pair of long-term, asymmetric public and private keys, verification establishes confidence that the claimed owner of a public key is the actual owner
2. Alice and Bob use a key exchange algorithm such as Diffie–Hellman, to securely agree on a short-term symmetric session key
   - *They use the asymmetric keys from step 1 only to authenticate one another during this process*
3. Alice sends Bob a message, encrypting it with a symmetric cipher using the session key negotiated in step 2
4. Bob decrypts Alice's message using the key negotiated in step 2
5. The symmetric session key exchange process repeats for each new message sent, starting from step 2 (switching Alice and Bob's roles as sender/receiver as appropriate)
   - *Step 1 is never repeated*

- Forward secrecy is achieved by generating new session keys for each message
  - It ensures that past communications cannot be decrypted if one of the keys generated in an iteration of step 2 is compromised, since such a key is only used to encrypt a single message
  - It also ensures that past communications cannot be decrypted if the long-term private keys from step 1 are compromised
- However, masquerading as Alice or Bob would be possible going forward if this occurred, possibly compromising all future messages

https://en.wikipedia.org/wiki/Forward_secrecy

Perfect Forward Secrecy

- Forward secrecy is present in several major protocol implementations:
  - SSH
  - IPsec (RFC 2412) as an optional feature
  - Transport Layer Security (TLS)
  - Cipher suites based on Diffie–Hellman key exchange (DHE-RSA, DHE-DSA)
  - Elliptic curve Diffie–Hellman key exchange (ECDHE-RSA, ECDHE-ECDSA)
  - OpenSSL supports forward secrecy using elliptic curve Diffie–Hellman since V1.0
  - Off-the-Record Messaging, a cryptography protocol and library for many instant messaging clients

Perfect Forward Security in use…

Google Security Blog

The latest news and insights from Google on security and safety on the Internet

Protecting data for the long term with forward secrecy

November 22, 2011

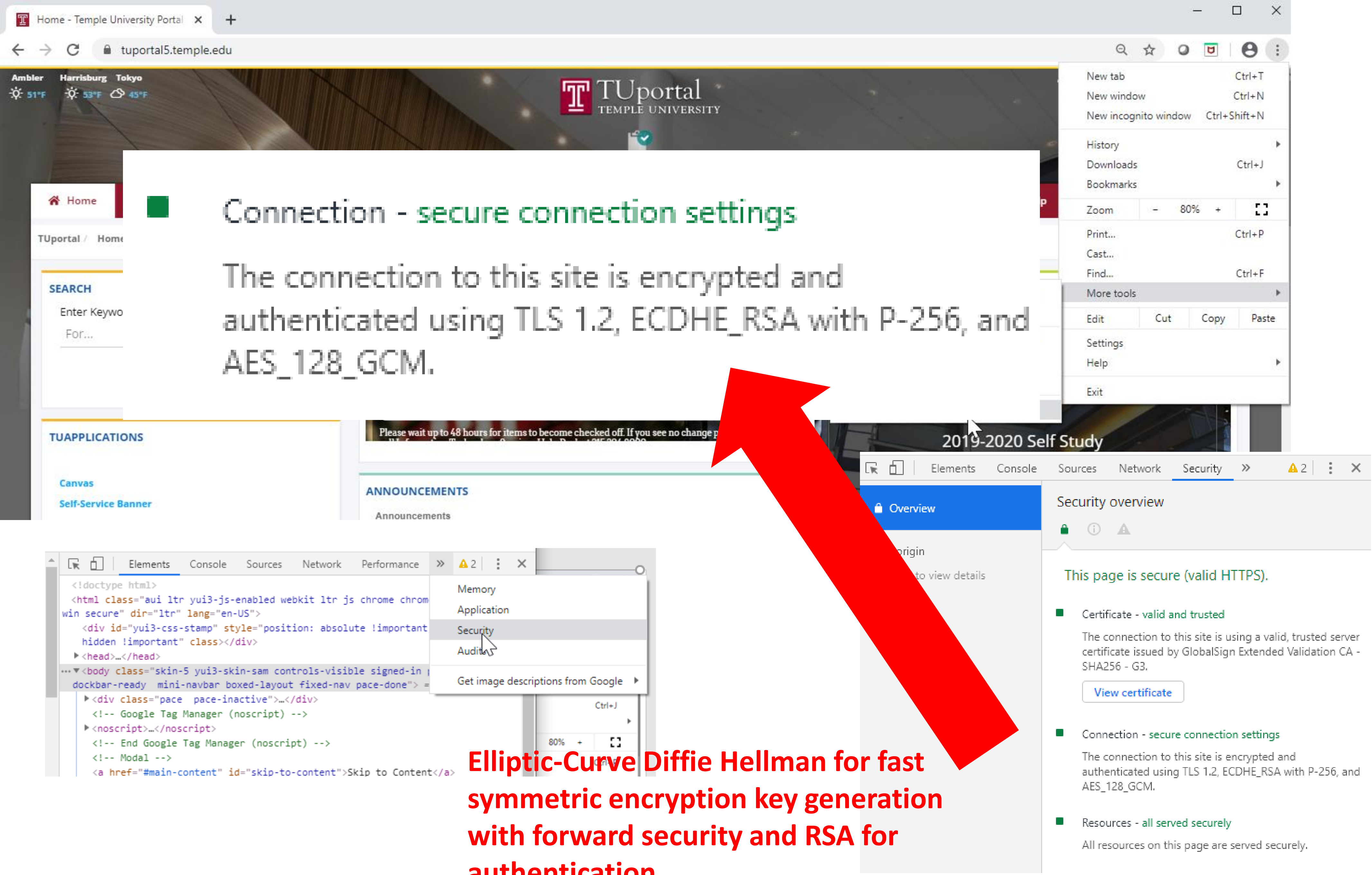
Posted by Adam Langley, Security Team

Last year we introduced HTTPS by default for Gmail and encrypted search. We're pleased to see that other major communications sites are following suit and deploying HTTPS in one form or another. We are now pushing forward by enabling forward secrecy by default.

Most major sites supporting HTTPS operate in a non-forward secret fashion, which runs the risk of retrospective decryption. In other words, an encrypted, unreadable email could be recorded while being delivered to your computer today. In ten years time, when computers are much faster, an adversary could break the server private key and

The new wireless encryption standard WPA3 uses Perfect Forward Security for all wireless clients

https://security.googleblog.com/2011/11/protecting-data-for-long-term-with.html



Elliptic-Curve Diffie Hellman for fast symmetric encryption key generation with forward security and RSA for authentication

Services of cryptosystems

- ✓**Confidentiality** – Renders information unintelligible except by authorized entities
- ✓**Authentication** – Verifies the identity of the user or system that created, requested or provided the information
- ✓**Nonrepudiation** – Ensure the sender cannot deny sending the information
- **Integrity** – Data has not been altered in an unauthorized manner since it was created, transmitted, or stored

Agenda

- ✓Team Project – It is not too early to get started…
- ✓Case Study 1
- ✓Cryptography terminology
- ✓Symmetric Key Cryptography
  - ✓Symmetric stream cryptography
  - ✓Symmetric block cryptography
- ✓Key sharing problem
- ✓Public Key Cryptography
  - ✓Diffie-Hellman algorithm: symmetric key generation through asynchronous cryptography
  - ✓RSA algorithm
- ✓Hybrid-Cryptography
  - ✓Perfect Forward Secrecy
- Where do cryptographic controls go in the FedRAMP System Security Plan
- Brief review: Hashing & Digital Signatures

Where in the FedRAMP System Security Plan would you look for information to help you assess the security of the Titan Information System?

FedRAMP

FedRAMP® (High, Moderate, Low, LI-SaaS) Baseline System Security Plan (SSP)

for <Insert CSP Name>

<Insert CSO Name>

<Insert Version X.X>

<Insert MM/DD/YYYY>

GSA Controlled Unclassified Information

info@fedramp.gov
fedramp.gov

TABLE OF CONTENTS

FedRAMP® (High, Moderate, Low, LI-SaaS) Baseline System Security Plan (SSP)
<Insert CSP Name> | <Insert CSO Name> | <Insert Version X.X> | <Insert MM/DD/YYYY>

FR

TABLE OF CONTENTS

FR

FedRAMP® (High, Moderate, Low, LI-SaaS) Baseline System Security Plan (SSP)
<Insert CSP Name> | <Insert CSO Name> | <Insert Version X.X> | <Insert MM/DD/YYYY>

Where do you look for encryption related controls that could help Titan?

NIST Special Publication 800-53
Revision 5

Security and Privacy Controls for Information Systems and Organizations

JOINT TASK FORCE

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.800-53r5

September 2020
INCLUDES UPDATES AS OF 12-10-2020; SEE PAGE XVII

U.S. Department of Commerce
Wilbur L. Ross, Jr., Secretary

National Institute of Standards and Technology
Walter Copan, NIST Director and Under Secretary of Commerce for Standards and Technology

CLASS	FAMILY	IDENTIFIER
Management	Risk Assessment	RA
Management	Planning	PL
Management	System and Services Acquisition	SA
Management	Certification, Accreditation, and Security Assessments	CA
Operational	Personnel Security	PS
Operational	Physical and Environmental Protection	PE
Operational	Contingency Planning	CP
Operational	Configuration Management	CM
Operational	Maintenance	MA
Operational	System and Information Integrity	SI
Operational	Media Protection	MP
Operational	Incident Response	IR
Operational	Awareness and Training	AT
Technical	Identification and Authentication	IA
Technical	Access Control	AC
Technical	Audit and Accountability	AU
Technical	System and Communications Protection	SC

CNTL NO.	CONTROL NAME	PRIORITY	INITIAL CONTROL BASELINES		
			LOW	MOD	HIGH
System and Communications Protection					
SC-1	System and Communications Protection Policy and Procedures	P1	SC-1	SC-1	SC-1
SC-2	Application Partitioning	P1	Not Selected	SC-2	SC-2
SC-3	Security Function Isolation	P1	Not Selected	Not Selected	SC-3
SC-4	Information in Shared Resources	P1	Not Selected	SC-4	SC-4
SC-5	Denial of Service Protection	P1	SC-5	SC-5	SC-5
SC-6	Resource Availability	P0	Not Selected	Not Selected	Not Selected
SC-7	Boundary Protection	P1	SC-7	SC-7 (3) (4) (5) (7)	SC-7 (3) (4) (5) (7) (8) (18) (21)
SC-8	Transmission Confidentiality and Integrity	P1	Not Selected	SC-8 (1)	SC-8 (1)
SC-9	**Withdrawn**	---	---	---	---
SC-10	Network Disconnect	P2	Not Selected	SC-10	SC-10
SC-11	Trusted Path	P0	Not Selected	Not Selected	Not Selected
SC-12	Cryptographic Key Establishment and Management	P1	SC-12	SC-12	SC-12 (1)
SC-13	Cryptographic Protection	P1	SC-13	SC-13	SC-13
SC-14	**Withdrawn**	---	---	---	---
SC-15	Collaborative Computing Devices	P1	SC-15	SC-15	SC-15
SC-16	Transmission of Security Attributes	P0	Not Selected	Not Selected	Not Selected
SC-17	Public Key Infrastructure Certificates	P1	Not Selected	SC-17	SC-17
SC-18	Mobile Code	P2	Not Selected	SC-18	SC-18
SC-19	Voice Over Internet Protocol	P1	Not Selected	SC-19	SC-19
SC-20	Secure Name /Address Resolution Service (Authoritative Source)	P1	SC-20	SC-20	SC-20
SC-21	Secure Name /Address Resolution Service (Recursive or Caching Resolver)	P1	SC-21	SC-21	SC-21
SC-22	Architecture and Provisioning for Name/Address Resolution Service	P1	SC-22	SC-22	SC-22
SC-23	Session Authenticity	P1	Not Selected	SC-23	SC-23
SC-24	Fail in Known State	P1	Not Selected	Not Selected	SC-24
SC-28	Protection of Information at Rest	P1	Not Selected	SC-28	SC-28
SC-39	Process Isolation	P1	SC-39	SC-39	SC-39

SC-12 CRYPTOGRAPHIC KEY ESTABLISHMENT AND MANAGEMENT

Control: The organization establishes and manages cryptographic keys for required cryptography employed within the information system in accordance with [*Assignment: organization-defined requirements for key generation, distribution, storage, access, and destruction*].

Supplemental Guidance: Cryptographic key management and establishment can be performed using manual procedures or automated mechanisms with supporting manual procedures. Organizations define key management requirements in accordance with applicable federal laws, Executive Orders, directives, regulations, policies, standards, and guidance, specifying appropriate options, levels, and parameters. Organizations manage trust stores to ensure that only approved trust anchors are in such trust stores. This includes certificates with visibility external to organizational information systems and certificates related to the internal operations of systems. Related controls: SC-13, SC-17.

Control Enhancements:

(1) *CRYPTOGRAPHIC KEY ESTABLISHMENT AND MANAGEMENT | AVAILABILITY*

The organization maintains availability of information in the event of the loss of cryptographic keys by users.

Supplemental Guidance: Escrowing of encryption keys is a common practice for ensuring availability in the event of loss of keys (e.g., due to forgotten passphrase).

(2) *CRYPTOGRAPHIC KEY ESTABLISHMENT AND MANAGEMENT | SYMMETRIC KEYS*

The organization produces, controls, and distributes symmetric cryptographic keys using [*Selection: NIST FIPS-compliant; NSA-approved*] key management technology and processes.

(3) *CRYPTOGRAPHIC KEY ESTABLISHMENT AND MANAGEMENT | ASYMMETRIC KEYS*

The organization produces, controls, and distributes asymmetric cryptographic keys using [*Selection: NSA-approved key management technology and processes; approved PKI Class 3 certificates or prepositioned keying material; approved PKI Class 3 or Class 4 certificates and hardware security tokens that protect the user's private key*].

(4) *CRYPTOGRAPHIC KEY ESTABLISHMENT AND MANAGEMENT | PKI CERTIFICATES*

[Withdrawn: Incorporated into SC-12].

(5) *CRYPTOGRAPHIC KEY ESTABLISHMENT AND MANAGEMENT | PKI CERTIFICATES / HARDWARE TOKENS*

[Withdrawn: Incorporated into SC-12].

References: NIST Special Publications 800-56, 800-57.

Priority and Baseline Allocation:

P1	LOW SC-12	MOD SC-12	HIGH SC-12 (1)

CNTL NO.	CONTROL NAME	PRIORITY	INITIAL CONTROL BASELINES		
			LOW	MOD	HIGH
System and Communications Protection					
SC-1	System and Communications Protection Policy and Procedures	P1	SC-1	SC-1	SC-1
SC-2	Application Partitioning	P1	Not Selected	SC-2	SC-2
SC-3	Security Function Isolation	P1	Not Selected	Not Selected	SC-3
SC-4	Information in Shared Resources	P1	Not Selected	SC-4	SC-4
SC-5	Denial of Service Protection	P1	SC-5	SC-5	SC-5
SC-6	Resource Availability	P0	Not Selected	Not Selected	Not Selected
SC-7	Boundary Protection	P1	SC-7	SC-7 (3) (4) (5) (7)	SC-7 (3) (4) (5) (7) (8) (18) (21)
SC-8	Transmission Confidentiality and Integrity	P1	Not Selected	SC-8 (1)	SC-8 (1)
SC-9	**Withdrawn**	---	---	---	---
SC-10	Network Disconnect	P2	Not Selected	SC-10	SC-10
SC-11	Trusted Path	P0	Not Selected	Not Selected	Not Selected
SC-12	Cryptographic Key Establishment and Management	P1	SC-12	SC-12	SC-12 (1)
SC-13	Cryptographic Protection	P1	SC-13	SC-13	SC-13
SC-14	**Withdrawn**	---	---	---	---
SC-15	Collaborative Computing Devices	P1	SC-15	SC-15	SC-15
SC-16	Transmission of Security Attributes	P0	Not Selected	Not Selected	Not Selected
SC-17	Public Key Infrastructure Certificates	P1	Not Selected	SC-17	SC-17
SC-18	Mobile Code	P2	Not Selected	SC-18	SC-18
SC-19	Voice Over Internet Protocol	P1	Not Selected	SC-19	SC-19
SC-20	Secure Name /Address Resolution Service (Authoritative Source)	P1	SC-20	SC-20	SC-20
SC-21	Secure Name /Address Resolution Service (Recursive or Caching Resolver)	P1	SC-21	SC-21	SC-21
SC-22	Architecture and Provisioning for Name/Address Resolution Service	P1	SC-22	SC-22	SC-22
SC-23	Session Authenticity	P1	Not Selected	SC-23	SC-23
SC-24	Fail in Known State	P1	Not Selected	Not Selected	SC-24
SC-28	Protection of Information at Rest	P1	Not Selected	SC-28	SC-28
SC-39	Process Isolation	P1	SC-39	SC-39	SC-39

SC-13 CRYPTOGRAPHIC PROTECTION

Control: The information system implements [*Assignment: organization-defined cryptographic uses and type of cryptography required for each use*] in accordance with applicable federal laws, Executive Orders, directives, policies, regulations, and standards.

Supplemental Guidance: Cryptography can be employed to support a variety of security solutions including, for example, the protection of classified and Controlled Unclassified Information, the provision of digital signatures, and the enforcement of information separation when authorized individuals have the necessary clearances for such information but lack the necessary formal access approvals. Cryptography can also be used to support random number generation and hash generation. Generally applicable cryptographic standards include FIPS-validated cryptography and NSA-approved cryptography. This control does not impose any requirements on organizations to use cryptography. However, if cryptography is required based on the selection of other security controls, organizations define each type of cryptographic use and the type of cryptography required (e.g., protection of classified information: NSA-approved cryptography; provision of digital signatures: FIPS-validated cryptography). Related controls: AC-2, AC-3, AC-7, AC-17, AC-18, AU-9, AU-10, CM-11, CP-9, IA-3, IA-7, MA-4, MP-2, MP-4, MP-5, SA-4, SC-8, SC-12, SC-28, SI-7.

Control Enhancements: None.

(1) *CRYPTOGRAPHIC PROTECTION | FIPS-VALIDATED CRYPTOGRAPHY*
[Withdrawn: Incorporated into SC-13].

(2) *CRYPTOGRAPHIC PROTECTION | NSA-APPROVED CRYPTOGRAPHY*
[Withdrawn: Incorporated into SC-13].

(3) *CRYPTOGRAPHIC PROTECTION | INDIVIDUALS WITHOUT FORMAL ACCESS APPROVALS*
[Withdrawn: Incorporated into SC-13].

(4) *CRYPTOGRAPHIC PROTECTION | DIGITAL SIGNATURES*
[Withdrawn: Incorporated into SC-13].

References: FIPS Publication 140; Web: http://csrc.nist.gov/cryptval, http://www.cnss.gov.

Priority and Baseline Allocation:

P1	LOW SC-13	MOD SC-13	HIGH SC-13

Where do you document this information in your SSP?

FedRAMP

FedRAMP® System Security Plan (SSP) Appendix A: High FedRAMP Security Controls

for <Insert CSP Name>

<Insert CSO Name>

<Insert Version X.X>

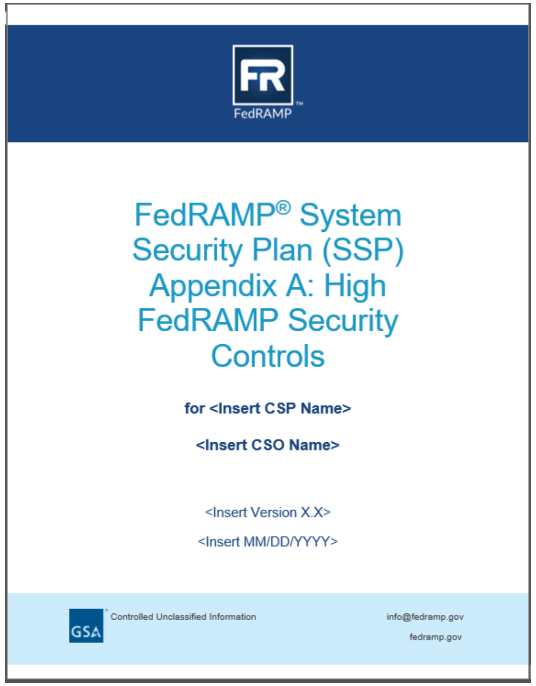
<Insert MM/DD/YYYY>

GSA Controlled Unclassified Information

info@fedramp.gov

fedramp.gov

TABLE OF CONTENTS



FedRAMP

FedRAMP® System Security Plan (SSP) Appendix A: High FedRAMP Security Controls

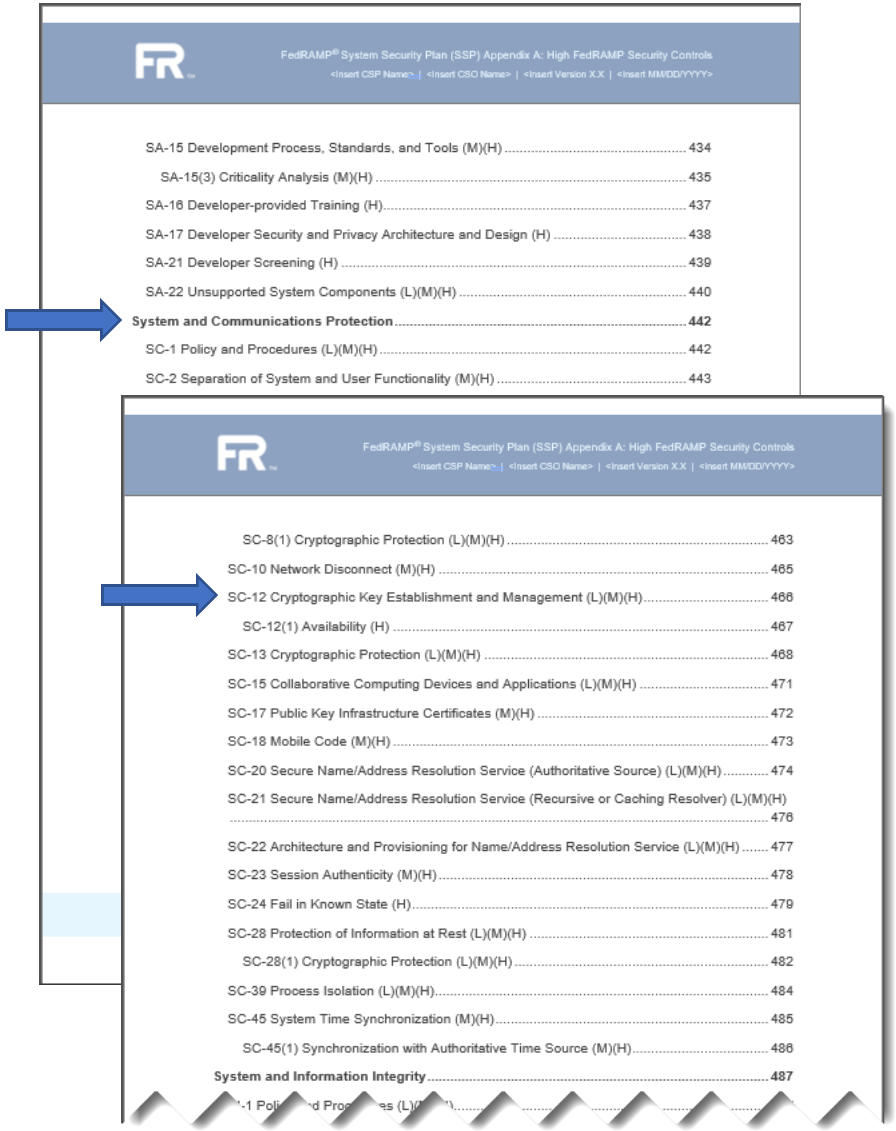
for <Insert CSP Name>

<Insert CSO Name>

<Insert Version X.X>

<Insert MM/DD/YYYY>

GSA Controlled Unclassified Information info@fedramp.gov fedramp.gov



FedRAMP® System Security Plan (SSP) Appendix A: High FedRAMP Security Controls
<Insert CSP Name> | <Insert CSO Name> | <Insert Version X.X | <Insert MM/DD/YYYY>

FedRAMP® System Security Plan (SSP) Appendix A: High FedRAMP Security Controls
<Insert CSP Name> | <Insert CSO Name> | <Insert Version X.X | <Insert MM/DD/YYYY>

SC-12 Cryptographic Key Establishment and Management (L)(M)(H)

Establish and manage cryptographic keys when cryptography is employed within the system in accordance with the following key management requirements: [FedRAMP Assignment: In accordance with Federal requirements].

SC-12 Additional FedRAMP Requirements and Guidance:

Guidance: See references in NIST 800-53 documentation.

Guidance: Must meet applicable Federal Cryptographic Requirements. See References Section of control.

Guidance: Wildcard certificates may be used internally within the system, but are not permitted for external customer access to the system.

SC-12 Control Summary Information
Responsible Role:
Parameter SC-12:
Implementation Status (check all that apply): ☐ Implemented ☐ Partially Implemented ☐ Planned ☐ Alternative implementation ☐ Not Applicable
Control Origination (check all that apply): ☐ Service Provider Corporate ☐ Service Provider System Specific ☐ Service Provider Hybrid (Corporate and System Specific) ☐ Configured by Customer (Customer System Specific) ☐ Provided by Customer (Customer System Specific) ☐ Shared (Service Provider and Customer Responsibility) ☐ Inherited from pre-existing FedRAMP Authorization for [Click here to enter text], Date of Authorization

SC-12 What is the solution and how is it implemented?

FedRAMP

FedRAMP® System Security Plan (SSP) Appendix A: High FedRAMP Security Controls

for <Insert CSP Name>

<Insert CSO Name>

<Insert Version X.X>

<Insert MM/DD/YYYY>

GSA Controlled Unclassified Information

info@fedramp.gov

fedramp.gov

FedRAMP® System Security Plan (SSP) Appendix A: High FedRAMP Security Controls
<Insert CSP Name> | <Insert CSO Name> | <Insert Version X.X | <Insert MM/DD/YYYY>

SC-12(1) Availability (H)

Maintain availability of information in the event of the loss of cryptographic keys by users.

SC-12(1) Control Summary Information
Responsible Role:
Implementation Status (check all that apply): ☐ Implemented ☐ Partially Implemented ☐ Planned ☐ Alternative implementation ☐ Not Applicable
Control Origination (check all that apply): ☐ Service Provider Corporate ☐ Service Provider System Specific ☐ Service Provider Hybrid (Corporate and System Specific) ☐ Configured by Customer (Customer System Specific) ☐ Provided by Customer (Customer System Specific) ☐ Shared (Service Provider and Customer Responsibility) ☐ Inherited from pre-existing FedRAMP Authorization for [Click here to enter text], Date of Authorization

SC-12(1) What is the solution and how is it implemented?

FedRAMP

FedRAMP® System Security Plan (SSP) Appendix A: High FedRAMP Security Controls

for <Insert CSP Name>

<Insert CSO Name>

<Insert Version X.X>

<Insert MM/DD/YYYY>

GSA Controlled Unclassified Information

info@fedramp.gov
fedramp.gov

FedRAMP® System Security Plan (SSP) Appendix A: High FedRAMP Security Controls
<Insert CSP Name> | <Insert CSO Name> | <Insert Version X.X | <Insert MM/DD/YYYY>

SC-12(1) Availability (H)

Maintain availability of information in the event of the loss of cryptographic keys by users.

SC-12(1) Control Summary Information
Responsible Role:
Implementation Status (check all that apply): ☐ Implemented ☐ Partially Implemented ☐ Planned ☐ Alternative implementation ☐ Not Applicable
Control Origination (check all that apply): ☐ Service Provider Corporate ☐ Service Provider System Specific ☐ Service Provider Hybrid (Corporate and System Specific) ☐ Configured by Customer (Customer System Specific) ☐ Provided by Customer (Customer System Specific) ☐ Shared (Service Provider and Customer Responsibility) ☐ Inherited from pre-existing FedRAMP Authorization for [Click here to enter text], Date of Authorization

SC-12(1) What is the solution and how is it implemented?

SC-12(1) Availability (H)

Maintain availability of information in the event of the loss of cryptographic keys by users.

SC-12(1) Control Summary Information
Responsible Role:
Implementation Status (check all that apply): ☐ Implemented ☐ Partially Implemented ☐ Planned ☐ Alternative implementation ☐ Not Applicable
Control Origination (check all that apply): ☐ Service Provider Corporate ☐ Service Provider System Specific ☐ Service Provider Hybrid (Corporate and System Specific) ☐ Configured by Customer (Customer System Specific) ☐ Provided by Customer (Customer System Specific) ☐ Shared (Service Provider and Customer Responsibility) ☐ Inherited from pre-existing FedRAMP Authorization for [Click here to enter text], Date of Authorization

SC-12(1) What is the solution and how is it implemented?

Learn / Azure / Security / Key Vault / Keys /

About keys

Article • 01/25/2023 • 9 contributors

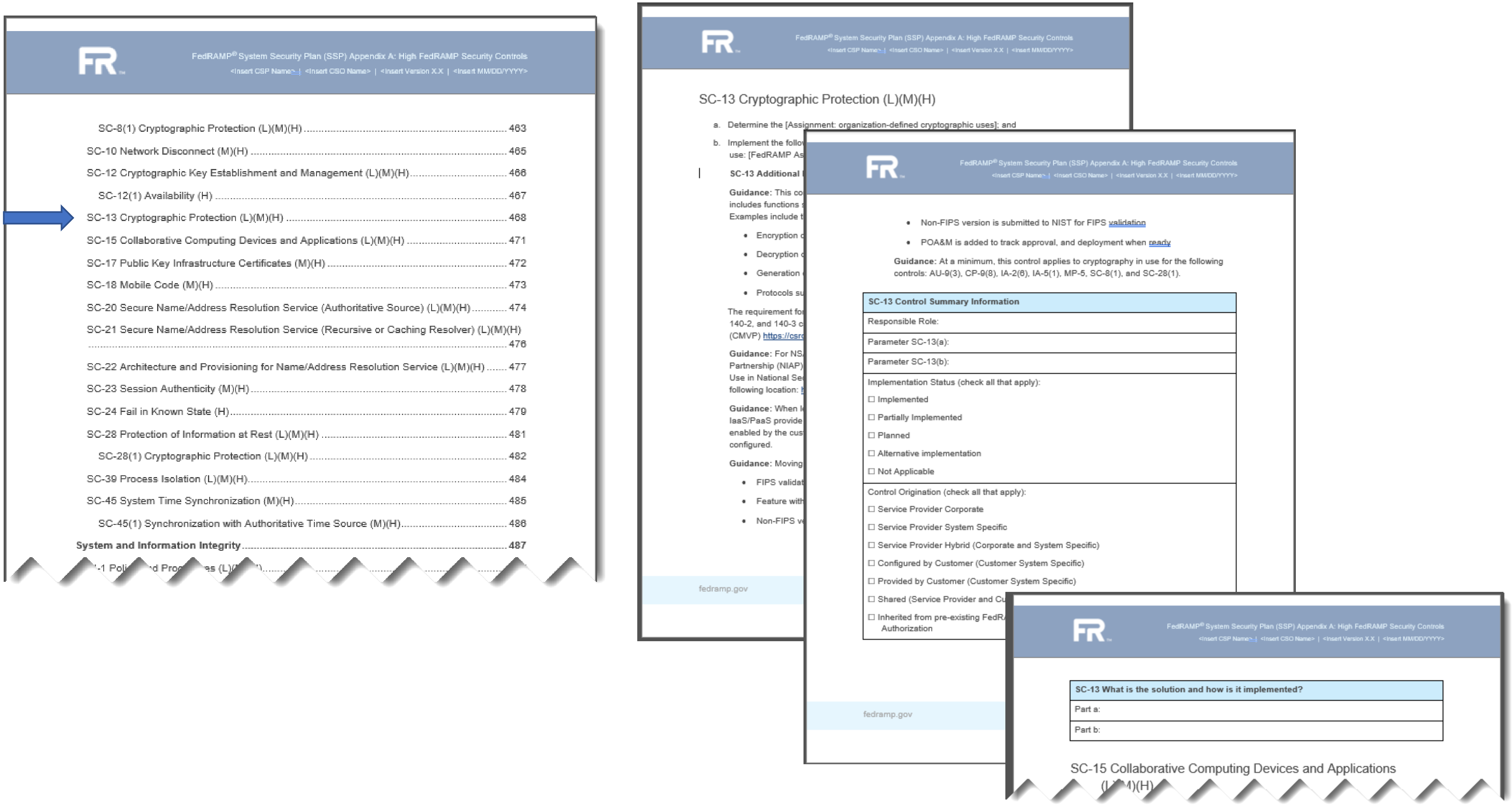
Feedback

In this article

Azure Key Vault provides two types of resources to store and manage cryptographic keys. Vaults support software-protected and HSM-protected (Hardware Security Module) keys. Managed HSMs only support HSM-protected keys.

Expand table

Resource type	Key protection methods	Data-plane endpoint base URL
Vaults	Software-protected and HSM-protected (with Premium SKU)	https://{vault-name}.vault.azure.net
Managed HSMs	HSM-protected	https://{hsm-name}.managedhsm.azure.net


FedRAMP® System Security Plan (SSP) Appendix A: High FedRAMP Security Controls
<Insert CSP Name> | <Insert CSO Name> | <Insert Version X.X | <Insert MM/DD/YYYY>
SC-8(1) Cryptographic Protection (L)(M)(H) 463
SC-10 Network Disconnect (M)(H) 465
SC-12 Cryptographic Key Establishment and Management (L)(M)(H) 466
SC-12(1) Availability (H) 467
SC-13 Cryptographic Protection (L)(M)(H) 468
SC-15 Collaborative Computing Devices and Applications (L)(M)(H) 471
SC-17 Public Key Infrastructure Certificates (M)(H) 472
SC-18 Mobile Code (M)(H) 473
SC-20 Secure Name/Address Resolution Service (Authoritative Source) (L)(M)(H) 474
SC-21 Secure Name/Address Resolution Service (Recursive or Caching Resolver) (L)(M)(H) 476
SC-22 Architecture and Provisioning for Name/Address Resolution Service (L)(M)(H) 477
SC-23 Session Authenticity (M)(H) 478
SC-24 Fail in Known State (H) 479
SC-28 Protection of Information at Rest (L)(M)(H) 481
SC-28(1) Cryptographic Protection (L)(M)(H) 482
SC-39 Process Isolation (L)(M)(H) 484
SC-45 System Time Synchronization (M)(H) 485
SC-45(1) Synchronization with Authoritative Time Source (M)(H) 486
System and Information Integrity 487
SC-13 Cryptographic Protection (L)(M)(H)
a. Determine the [Assignment: organization-defined cryptographic uses]; and
b. Implement the follo
use: [FedRAMP As
SC-13 Additional
Guidance: This co
includes functions
Examples include t
Encryption o
Decryption o
Generation
Protocols su
The requirement fo
140-2, and 140-3 c
(CMVP) https://csrc
Guidance: For NS
Partnership (NIAP)
Use in National Se
following location:
Guidance: When l
IaaS/PaaS provide
enabled by the cus
configured.
Guidance: Moving
FIPS validat
Feature with
Non-FIPS v
fedramp.gov
Non-FIPS version is submitted to NIST for FIPS validation
POA&M is added to track approval, and deployment when ready
Guidance: At a minimum, this control applies to cryptography in use for the following controls: AU-9(3), CP-9(8), IA-2(6), IA-5(1), MP-5, SC-8(1), and SC-28(1).
SC-13 Control Summary Information
Responsible Role:
Parameter SC-13(a):
Parameter SC-13(b):
Implementation Status (check all that apply):
☐ Implemented
☐ Partially Implemented
☐ Planned
☐ Alternative implementation
☐ Not Applicable
Control Origination (check all that apply):
☐ Service Provider Corporate
☐ Service Provider System Specific
☐ Service Provider Hybrid (Corporate and System Specific)
☐ Configured by Customer (Customer System Specific)
☐ Provided by Customer (Customer System Specific)
☐ Shared (Service Provider and Cu
☐ Inherited from pre-existing FedR
Authorization
fedramp.gov
SC-13 What is the solution and how is it implemented?
Part a:
Part b:
SC-15 Collaborative Computing Devices and Applications

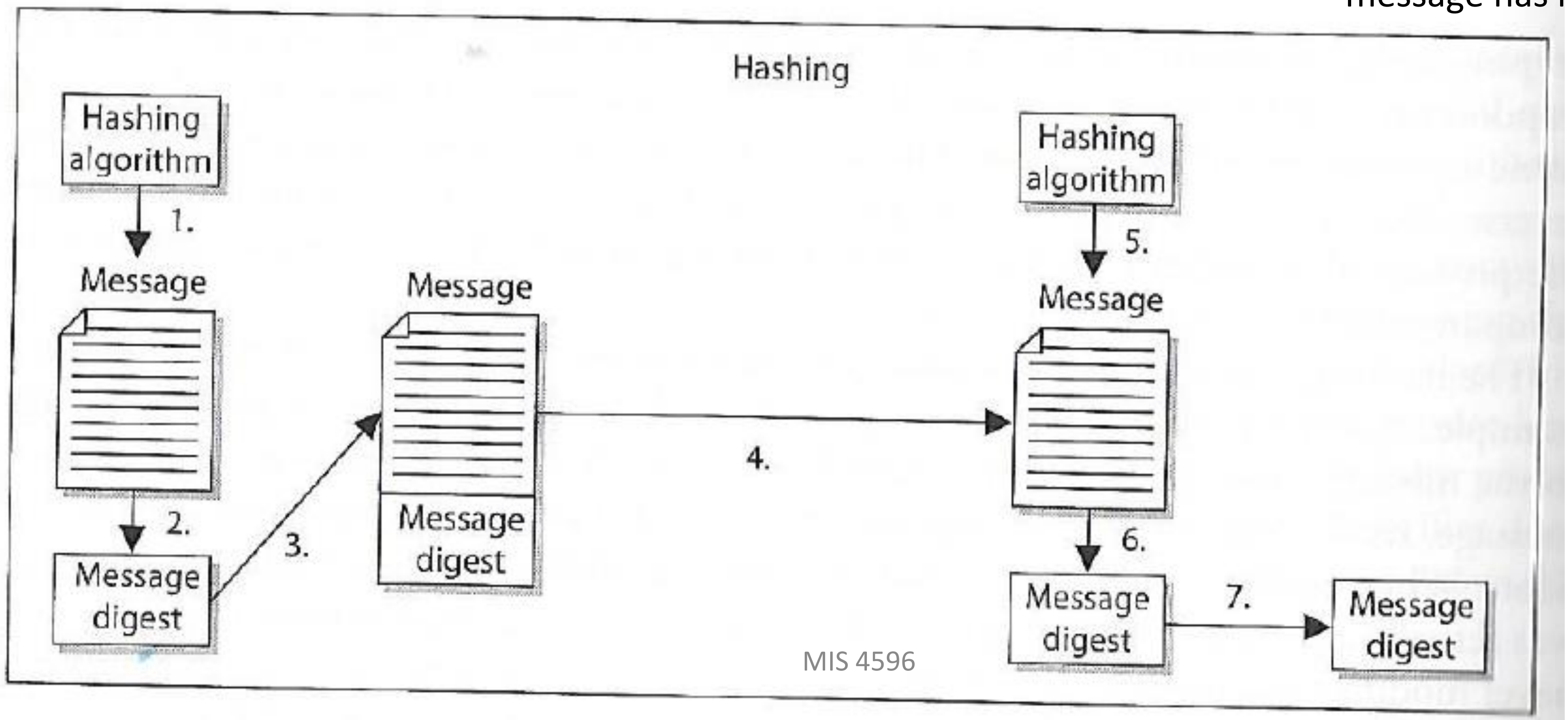
Agenda

- ✓Case Study 1
- ✓Cryptography terminology
- ✓Symmetric Key Cryptography
  - ✓Symmetric stream cryptography
  - ✓Symmetric block cryptography
- ✓Key sharing problem
- ✓Public Key Cryptography
  - ✓Diffie-Hellman algorithm: symmetric key generation through asynchronous cryptography
  - ✓RSA algorithm
- ✓Hybrid-Cryptography
  - ✓Perfect Forward Secrecy
- ✓Where do cryptographic controls go in the FedRAMP System Security Plan
- Brief review: Hashing & Digital Signatures

Quick Review: One-way Hash

- Assures message **integrity**
- A function that takes a variable-length string (i.e. message) and produces a fixed-length value called a hash value
- Does not use keys

1. Sender puts message through hashing function
2. Message digest generated
3. Message digest appended to the message
4. Sender sends message to receiver
5. Receiver puts message through hashing function
6. Receiver generates message digest value
7. Receiver compares the two message digests values. If they are the same, the message has not been altered

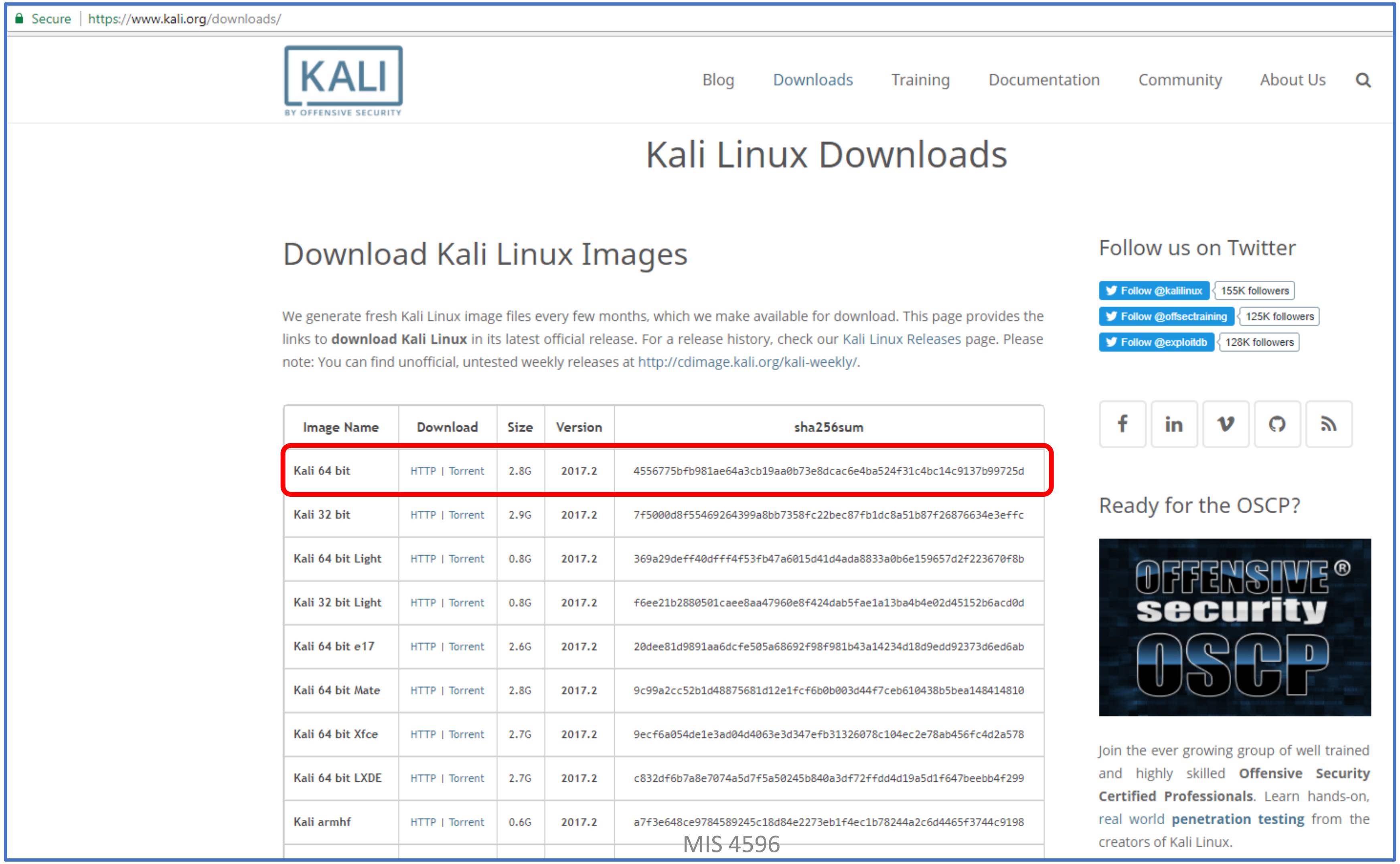


Note: Hashing results in **fixed-sized output**

- Names for the output of a hashing functions include “hash” and a *message digest* **(md)**, because a hash “digests” an input of any size down to a **fixed-sized output**
  - No matter the size of the input, the out put is the same, for example the md5 hash function’s output:
    - Letter ‘a’ in binary: 01000001 => md5 hash => 32-character string
    - Blu-ray disk digest => md5 hash => 32-character string
    - 6 TB hard drive digest => md5 hash => 32-character string

One-way hash example…

Testing the integrity of a file (e.g. program) downloaded from the internet…



Secure | https://www.kali.org/downloads/

KALI BY OFFENSIVE SECURITY

Blog Downloads Training Documentation Community About Us

Kali Linux Downloads

Download Kali Linux Images

We generate fresh Kali Linux image files every few months, which we make available for download. This page provides the links to **download Kali Linux** in its latest official release. For a release history, check our Kali Linux Releases page. Please note: You can find unofficial, untested weekly releases at http://cdimage.kali.org/kali-weekly/.

Image Name	Download	Size	Version	sha256sum
Kali 64 bit	HTTP | Torrent	2.8G	2017.2	4556775bfb981ae64a3cb19aa0b73e8dcac6e4ba524f31c4bc14c9137b99725d
Kali 32 bit	HTTP | Torrent	2.9G	2017.2	7f5000d8f55469264399a8bb7358fc22bec87fb1dc8a51b87f26876634e3effc
Kali 64 bit Light	HTTP | Torrent	0.8G	2017.2	369a29deff40dfff4f53fb47a6015d41d4ada8833a0b6e159657d2f223670f8b
Kali 32 bit Light	HTTP | Torrent	0.8G	2017.2	f6ee21b2880501caee8aa47960e8f424dab5fae1a13ba4b4e02d45152b6acd0d
Kali 64 bit e17	HTTP | Torrent	2.6G	2017.2	20dee81d9891aa6dcfe505a68692f98f981b43a14234d18d9edd92373d6ed6ab
Kali 64 bit Mate	HTTP | Torrent	2.8G	2017.2	9c99a2cc52b1d48875681d12e1fcf6b0b003d44f7ceb610438b5bea148414810
Kali 64 bit Xfce	HTTP | Torrent	2.7G	2017.2	9ecf6a054de1e3ad04d4063e3d347efb31326078c104ec2e78ab456fc4d2a578
Kali 64 bit LXDE	HTTP | Torrent	2.7G	2017.2	c832df6b7a8e7074a5d7f5a50245b840a3df72ffdd4d19a5d1f647beebb4f299
Kali armhf	HTTP | Torrent	0.6G	2017.2	a7f3e648ce9784589245c18d84e2273eb1f4ec1b78244a2c6d4465f3744c9198

Follow us on Twitter

Follow @kalilinux 155K followers
Follow @offsectraining 125K followers
Follow @exploitdb 128K followers

Ready for the OSCP?

OFFENSIVE® security OSCP

Join the ever growing group of well trained and highly skilled **Offensive Security Certified Professionals**. Learn hands-on, real world **penetration testing** from the creators of Kali Linux.

One-way hash example…

Image Name	Download	Size	Version	sha256sum
Kali 64 bit	HTTP | Torrent	2.8G	2017.2	4556775bfb981ae64a3cb19aa0b73e8dcac6e4ba524f31c4bc14c9137b99725d

Windows PowerShell

```
PS C:\Users\tue87168> cd Downloads
PS C:\Users\tue87168\Downloads> dir *.iso


    Directory: C:\Users\tue87168\Downloads


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        8/10/2017  10:55 AM      674803712 CSET_8.0 (1).iso
-a----        8/10/2017  11:03 AM      674803712 CSET_8.0 (2).iso
-a----        6/12/2017  10:29 AM      674803712 CSET_8.0.iso
-a----        9/27/2017   3:03 PM     2421987328 en_project_professional_2016_x86_x64_dvd_6962236.iso
-a----        10/3/2017   8:49 PM     2421987328 en_visio_professional_2016_x86_x64_dvd_6962139.iso
-a----       11/11/2016  11:45 AM     1469054976 Fedora-Live-Workstation-x86_64-23-10.iso
-a----        11/9/2017   2:31 PM     3020619776 kali-linux-2017.2-amd64.iso


PS C:\Users\tue87168\Downloads> Get-FileHash kali-linux-2017.2-amd64.iso | Format-List

Algorithm : SHA256
Hash      : 4556775BFB981AE64A3CB19AA0B73E8DCAC6E4BA524F31C4BC14C9137B99725D
Path      : C:\Users\tue87168\Downloads\kali-linux-2017.2-amd64.iso


PS C:\Users\tue87168\Downloads> _
```

One-way hash example…

Windows PowerShell

Notice the amount of <u>confusion</u> and <u>diffusion</u> resulting from a 1 character change!

```
PS C:\Users\tue87168\Downloads> dir *.txt


    Directory: C:\Users\tue87168\Downloads


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        11/9/2017   3:04 PM             15 MIS5206-IsGood.txt


PS C:\Users\tue87168\Downloads> type MIS5206-IsGood.txt
MIS5206 is good
PS C:\Users\tue87168\Downloads> Get-FileHash MIS5206-IsGood.txt | Format-List


Algorithm : SHA256
Hash      : E6F053ADE3857C0EDC2896B229D0B91D4752B2D9D8C9BD4B2A45A4ACCB3999DD
Path      : C:\Users\tue87168\Downloads\MIS5206-IsGood.txt



PS C:\Users\tue87168\Downloads> type MIS5206-IsGood.txt
MIS5206 is goop
PS C:\Users\tue87168\Downloads> Get-FileHash MIS5206-IsGood.txt | Format-List


Algorithm : SHA256
Hash      : 877B45EA5D40D98FF8D1ABD919E154F446FEA11387DBB13DDEE448F9932928A5
Path      : C:\Users\tue87168\Downloads\MIS5206-IsGood.txt



PS C:\Users\tue87168\Downloads>
```

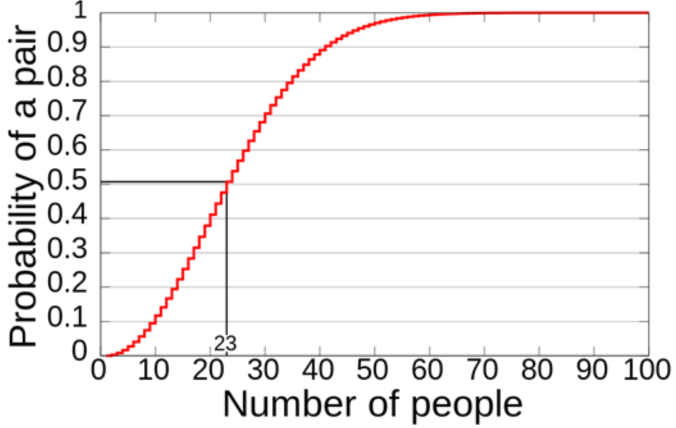
Cryptanalysis Attack

Collisions

- *Two different messages with the same hash value*
- Based on the "birthday paradox"
- Hash algorithms should be resistant to this attack

The birthday paradox, also known as the birthday problem, states that in a random group of 23 people, there is about a 50 percent chance that two people have the same birthday.

Is the Birthday Attack Real?



- There are multiple reasons why this seems like a paradox

- One is that when in a room with 22 other people, if a person compares his or her birthday with the birthdays of the other people it would make for only 22 comparisons—only 22 chances for people to share the same birthday.

When all 23 birthdays are compared against each other, it makes for much more than 22 comparisons. How much more? Well, the first person has 22 comparisons to make, but the second person was already compared to the first person, so there are only 21 comparisons to make. The third person then has 20 comparisons, the fourth person has 19 and so on. If you add up all possible comparisons (22 + 21 + 20 + 19 + … +1) the sum is 253 comparisons, or combinations. Consequently, each group of 23 people involves 253 comparisons, or 253 chances for matching birthdays.

MD5 (Message Digest 5)

- A 128-bit hash algorithm, still in common use
- Has been broken
- 128-bit hash, but only need $2^{128/2} = 2^{64}$ to find a collision
- Not strong enough for modern computers

Example of an MD5 hash collision:

Name	Date modified	Type	Size
ProgramA	1/27/2020 4:08 PM	Application	6 KB
ProgramB	1/27/2020 4:08 PM	Application	6 KB

```
Hello, world!

(press enter to quit)
```

ProgramA run

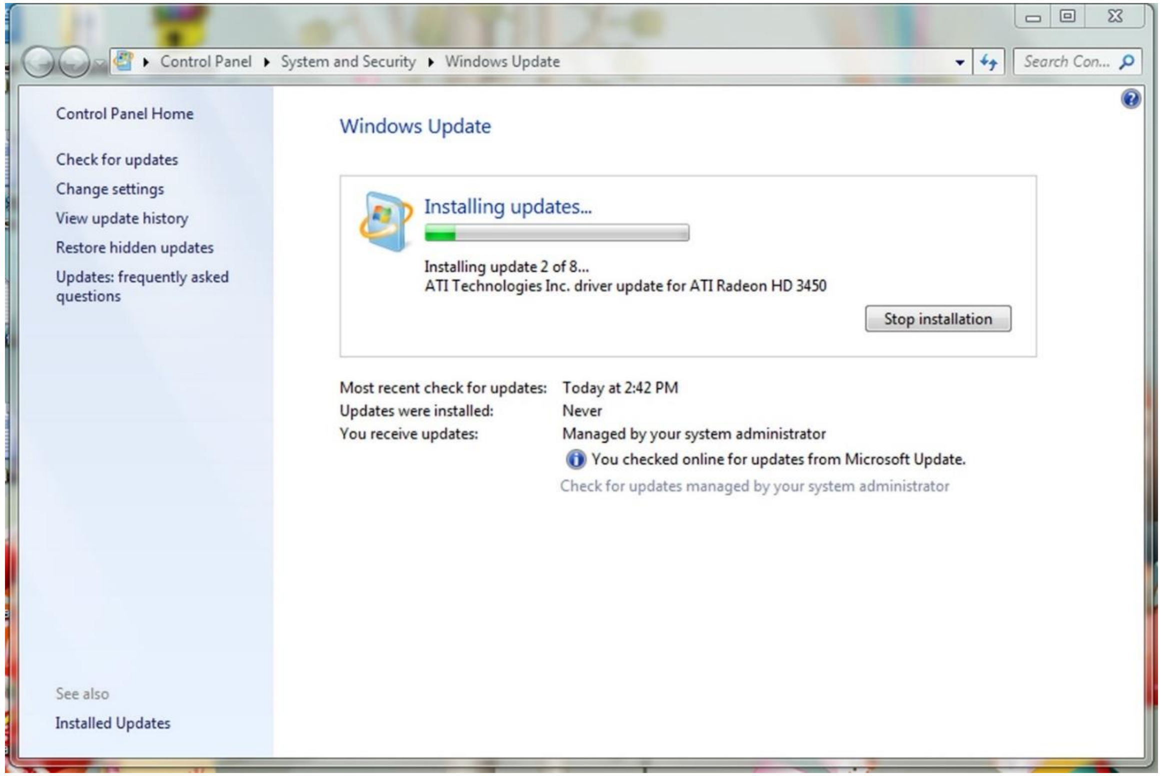
```
This program is evil!!!
Erasing hard drive...1Gb...2Gb... just kidding!
Nothing was erased.

(press enter to quit)_
```

ProgramB run

```
Windows PowerShell
PS C:\Users\Dave\Desktop\MD5-Hash-Collision-Example> get-filehash ProgramA.exe -Algorithm MD5

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
MD5             CDC47D670159EEF60916CA03A9D4A007                                       C:\Users\Dave\Desktop\MD5-Hash-Collision-Example\ProgramA.exe

PS C:\Users\Dave\Desktop\MD5-Hash-Collision-Example> get-filehash ProgramB.exe -Algorithm MD5

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
MD5             CDC47D670159EEF60916CA03A9D4A007                                       C:\Users\Dave\Desktop\MD5-Hash-Collision-Example\ProgramB.exe
```

In 2012 malware Flame used a MD5 hash collision to hijack Microsoft Windows Update and spread itself across networks

- Flame collected **audio, keystrokes, screenshots** which it sent to a malicious server
- Found a collision within a single millisecond
- Cost ~$200k computing time just for 1ms
- Attributed to advanced persistent threat group Equation Group
- Used in espionage attacks on countries



SHA -1 (Security Hash Algorithm 1)

- A 160-bit hash algorithm, still in common use
- Has been broken
- 160-bit hash, but only need $2^{160/2} = 2^{80}$ to find a collision
- No longer strong enough for modern computers

SHA-2 uses 224, **256**, 384, and 512-bit hashes

- But… it is built using the design of SHA-1, and prone to the same weaknesses
- It's believed to be a matter of time before SHA-2 is also exploited

SHA-3 was recently ratified by NIST, the U.S. National Institute of Standards and Technology

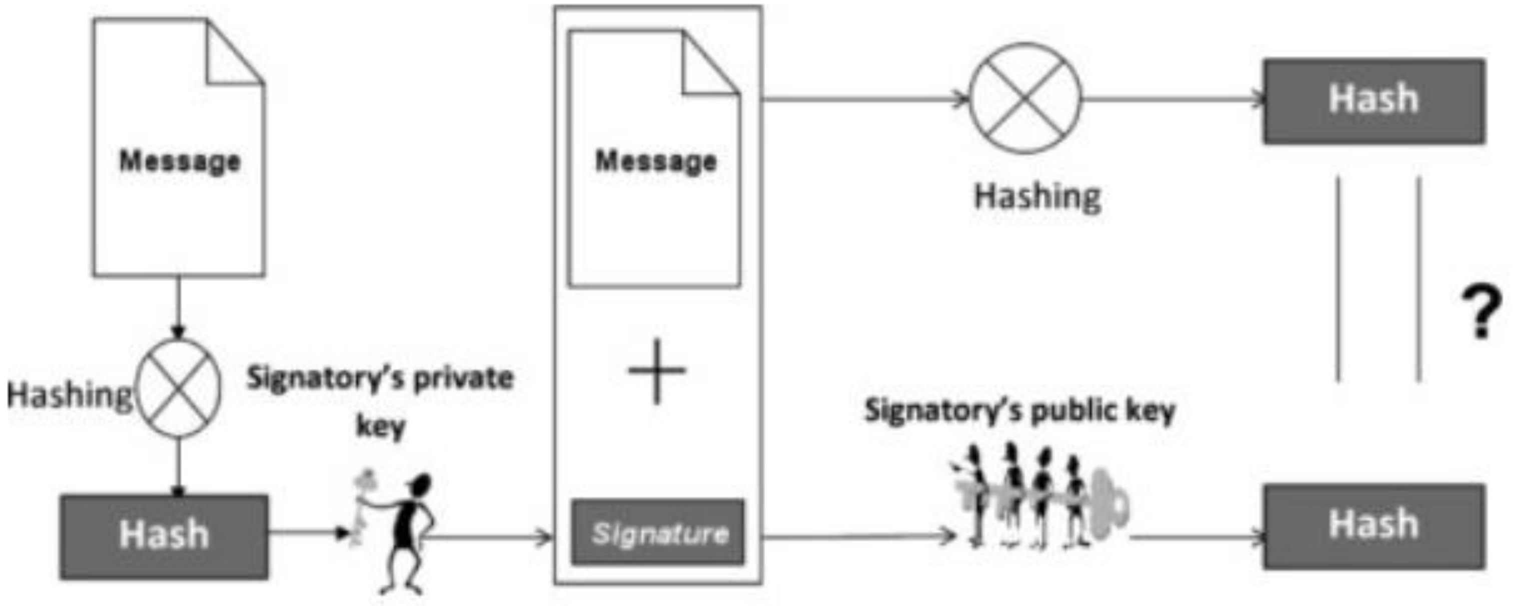
- It was the result of a six-year hashing competition. Also uses 224-, 256-, 384-, 512-bit hashes

Why does this matter for businesses?

Business needs a reliable way to prove integrity of data, files, programs, that can be trusted

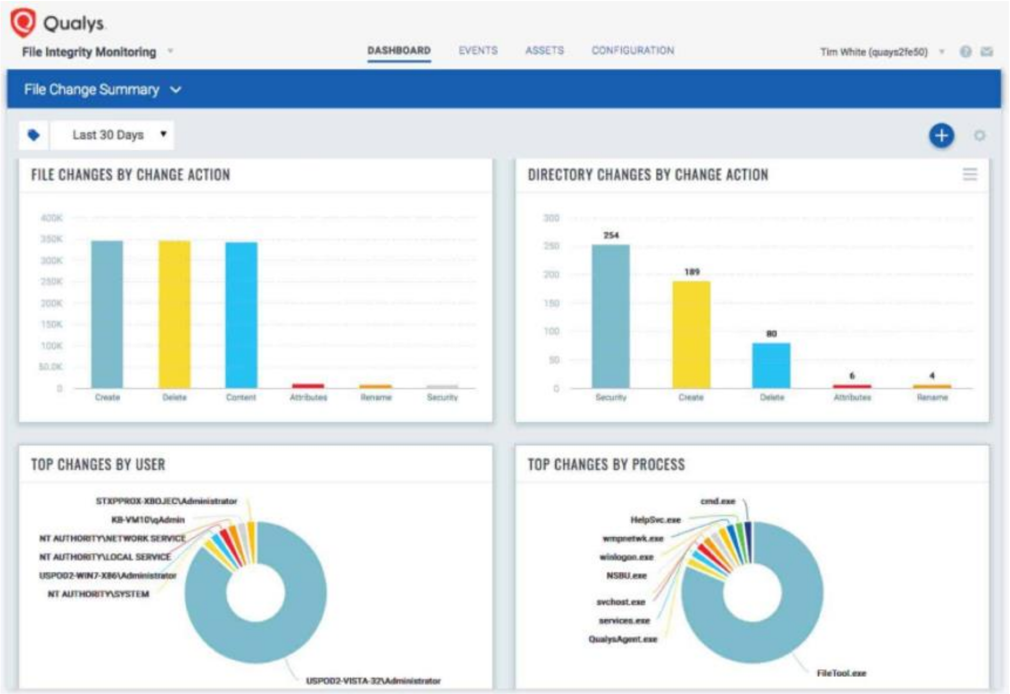
Digital Signature

The act of signing means encrypting the message's hash value with the private key



Integrity-based malware detection

- Calculates and stores a hash for each component of the system: operating system files, application files, configuration files, …
- Each new scan of the system calculates a hash for each component and compares it with the stored hash to detect differences
- Detected differences send alters and are flagged as suspect for further analysis



Services of cryptosystems

- ✓**Confidentiality** – Renders information unintelligible except by authorized entities
- ✓**Authentication** – Verifies the identity of the user or system that created, requested or provided the information
- ✓**Nonrepudiation** – Ensure the sender cannot deny sending the information
- ✓**Integrity** – Data has not been altered in an unauthorized manner since it was created, transmitted, or stored

Summary: Symmetric Algorithms

Name	Key Length (bits)	Block Size (bits)	Notes
DES	56 (56 + 8 parity)	64	Replaced by 3DES
3DES	56, 112, or 168 (+ 8, 16, 24 parity)	64	Replaced by AES
Blowfish	32 to 448	64	Replaced by Twofish Slower than AES.
TwoFish	128, 192, or 256	128	
AES (Rijandel)	128, 192, or 256	128	FIPS 197
RC4	8 to 2048-bit key (usually 40 to 256)	Stream	No longer in use
RC5	Variable (up to 2048)	32, 64, or 128	Very Strong
RC6	128, 192, and 256 bits up to 2040-bits	128	Based on RC5. (RSA)

Summary: Hashing Algorithms (Integrity)

	Hash Size (bits)	Block Size (bits)	Rounds	Strength
MD5	128	512	64	Weak – Password Files
SHA-0	160	512	80	Weak
SHA-1	160	512	80	Generally not recommended for Federal Systems – Refer to NIST SP800-131A for allowable uses.
SHA-2 (224 or 256)	224 or 256	512	64	Acceptable, 256 recommended
SHA-2 (384 or 512)	384 or 512	1024	80	All of the following are acceptable. Refer to NIST SP800-57 Part 1
SHA-512/224	224	1024	80	
SHA-512/256	256	1024	80	
SHA3-224	224	1600	1152	
SHA3-256	256	1600	1088	
SHA3-384	384	1600	832	
SHA3-512	512	1600	576	

https://csrc.nist.gov/Projects/Hash-Functions/NIST-Policy-on-Hash-Functions

Summary: Asymmetric Algorithms *(primarily used for key transport/exchange)*

- RSA – is the Public Key Cryptography Standard #1 (PKCS)
- Diffie-Hellman
- Elliptic Curve Cryptography
  - ECDH (Elliptic-Curve Diffie-Hellman) for secure key exchange for symmetric cryptography, provides forward secrecy therefore compromise of long-term key does not affect past sessions
  - ECDSA (Elliptic-Curve Digital Signature Algorithm) for digital signatures enabling authenticity and integrity of a message or certificate

Agenda

- ✓Team Project – It is not too early to get started…
- ✓Case Study 1
- ✓Cryptography terminology
- ✓Symmetric Key Cryptography
  - ✓Symmetric stream cryptography
  - ✓Symmetric block cryptography
- ✓Key sharing problem
- ✓Public Key Cryptography
  - ✓Diffie-Hellman algorithm: symmetric key generation through asynchronous cryptography
  - ✓RSA algorithm
- ✓Hybrid-Cryptography
  - ✓Perfect Forward Secrecy
- ✓Where do cryptographic controls go in the FedRAMP System Security Plan
- ✓Brief review: Hashing & Digital Signatures

Quiz

Which control is the BEST way to ensure that the data in a file have not been changed during transmission?

a) Reasonableness check
b) Parity bits
c) Hash values
d) Check digits

The PRIMARY reason for using digital signatures is to ensure data:

a) confidentiality
b) integrity
c) availability
d) Timeliness

Which of the following provides the GREATEST assurance for database password encryption?

a) Secure hash algorithm-256 (SHA-256)
b) Advanced encryption standard (AES)
c) Secure Shell (SSH)
d) Triple data encryption standard (3DES)

Email message authenticity and confidentiality is BEST achieved by signing the message using the:

a) Sender’s private key and encrypting the message using the receiver’s public key
b) Sender’s public key and encrypting the message using the receiver’s private key
c) Receiver’s private key and encrypting the message using the sender’s public key
d) Receiver’s public key and encrypting the message using the sender’s private key