Fundamentals of Cryptography: Session 2

NALINI ELKINS

INDUSTRY NETWORK TECHNOLOGY COUNCIL

PRESIDENT@INDUSTRYNETCOUNCIL.ORG

A few words about me

- President: Industry Network Technology Council
- Founder & CEO: Inside Products, Inc.
- Advisory Board: India Internet Engineering Society
- RFCs: RFC8250 (Embedded performance and diagnostics for IPv6) and others
- Product developer (OEMed by IBM and others)
- Working with IPv6 for over 20 years
- Working with network management, diagnostic, cryptography, performance issues at large brick-and-mortar enterprises for over 30 years

Fundamentals of Cryptography

Details:

- DES
- 3DES
- Asymmetric encryption / symmetric encryption
- Elliptic curve cryptography
- Certificate authority
- Diffie-Hellman key exchange
- Diffie-Hellman groups
- Hashed message authentication code (HMAC)
- HMAC MD5
- HMAC_SHA
- Message authentication code (MAC)
- Message digest algorithm 5 (MD5)
- Rivest Shamir Adleman (RSA)
- Secure hash algorithm 1 (SHA1)
- X.500 distinguished name
- X.509 digital certificate

Concepts

- Block cipher
- Encryption
- Hash
- Keys
- Public / private keys
- Tags

Issues

- Key sizes
- Choice of protocol
- End-to-end security

Forward Secrecy

- Protects past sessions against future compromises of keys or passwords.
- Generates a unique session key for every session a user initiates (ephemeral key)
- Compromise of a single key will not affect any data other than that exchanged in the specific session protected by that particular key.
- Goal: session keys cannot be determined even by an attacker who obtains the long-term keys some time after the conversation or an attacker who is in possession of the long-term keys but remains passive during the conversation. (https://www.rfc-editor.org/rfc/rfc7525)

Post-Compromise Secrecy

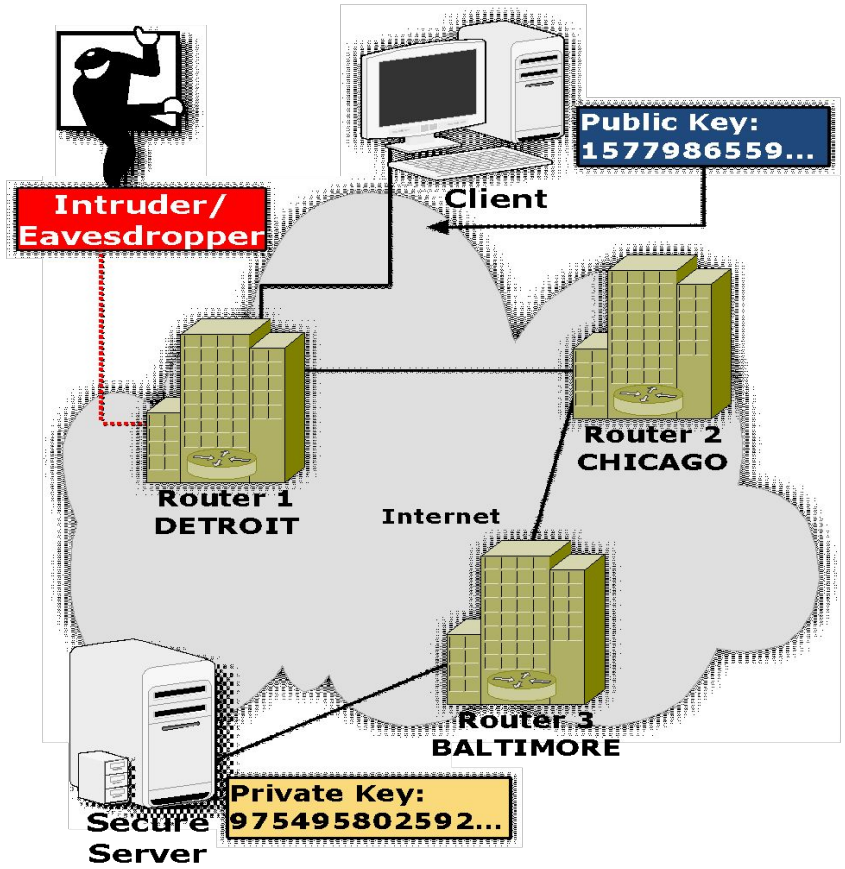
- Healing will occur such that if a key is compromised, then key updates are done.
- From MLS Architecture:

"PCS means that if a group member's state is compromised at some time t1 but the group member subsequently performs an update at some time t2, then all MLS guarantees apply to messages sent by the member after time t2, and by other members after they have processed the update. For example, if an attacker learns all secrets known to Alice at time t1, including both Alice's long-term secret keys and all shared group keys, but Alice performs a key update at time t2, then the attacker is unable to violate any of the MLS security properties after the updates have been processed."

https://datatracker.ietf.org/doc/draft-ietf-mls-architecture/

Pros and Cons of Public Key Systems

- There is no need to agree on a common key for both the sender and the receiver.
- If someone wants to receive an encrypted message, the sender only needs to know the receiver's public key.
- As long as the receiver keeps the private key secret, no one but the receiver will be able to decrypt the messages.
- Unlike symmetric algorithms, public-key systems can guarantee integrity and authentication, not only privacy.
- The main disadvantage of using public-key systems is that they are not as fast as symmetric algorithms



https://www.iiesoc.in/

https://industrynetcouncil.org/

Elliptic Curve Cryptology

- One of the advances in cryptography is called Elliptic Curve Cryptography.
- In theory, elliptic-curve cryptography can use smaller keys and is more efficient than other public-key methods.
- With more and more devices requiring secure communications, speed and size matter.

https://www.iiesoc.in/

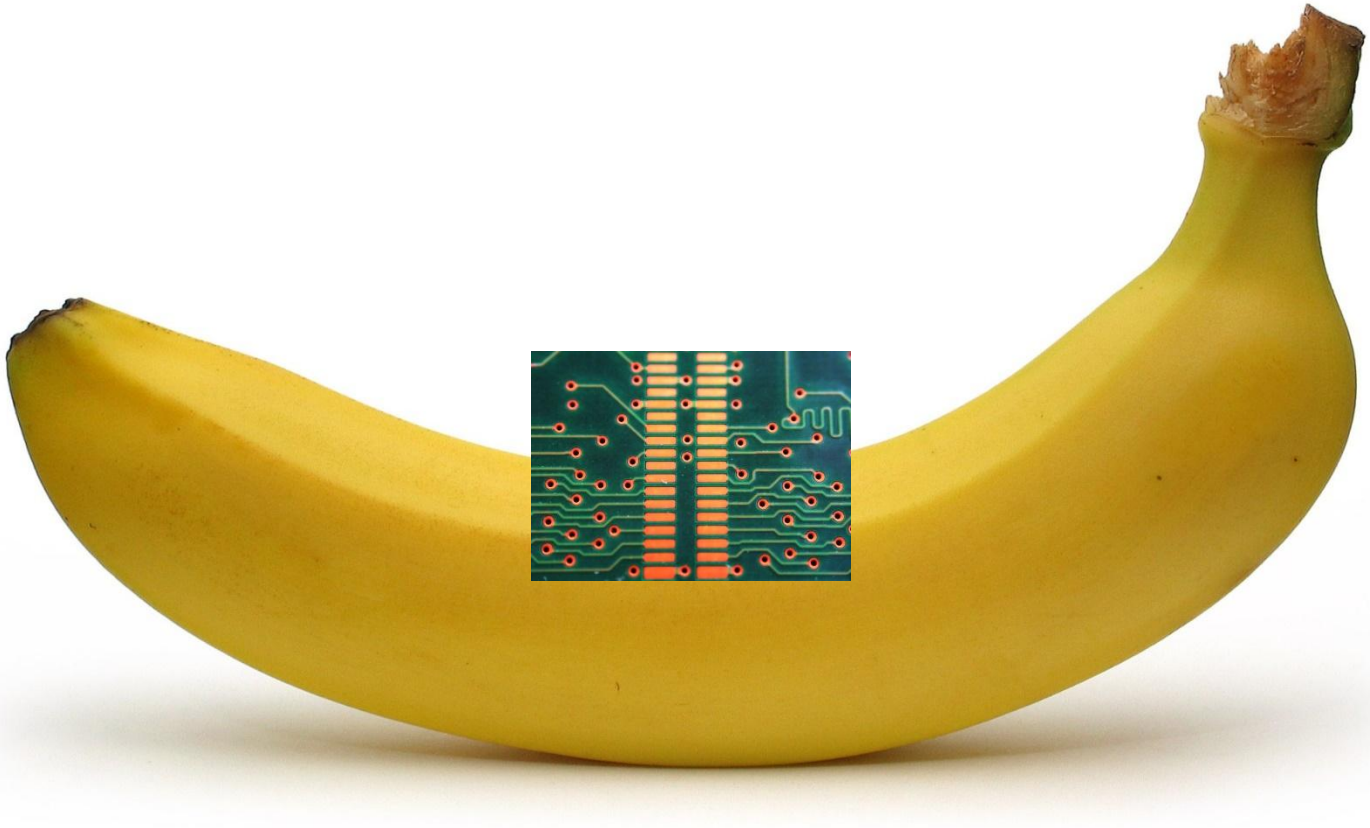
https://industrynetcouncil.org/

Embedded Systems

- Small key sizes and efficient signatures make elliptic curve cryptography the method of choice for securing many embedded systems.

- As more embedded devices become networked or connected, ECC offers many advantages when security is required.

RFID Sensor Networks



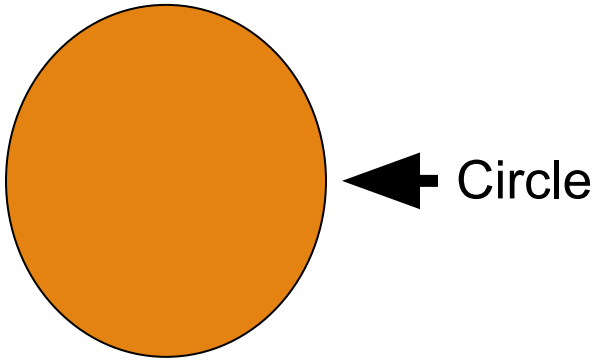
https://www.iiesoc.in/

https://industrynetcouncil.org/

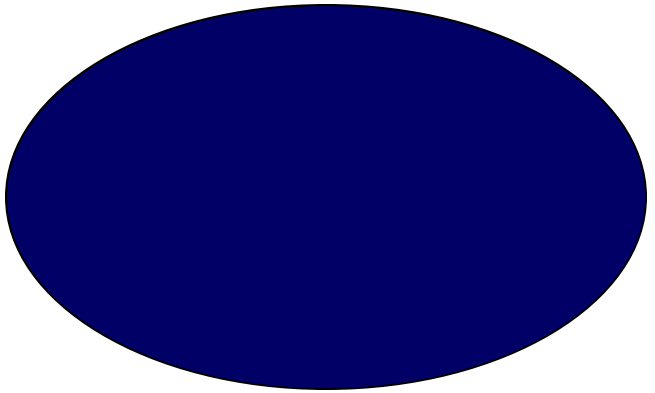
Getting Started

Integers are the numbers

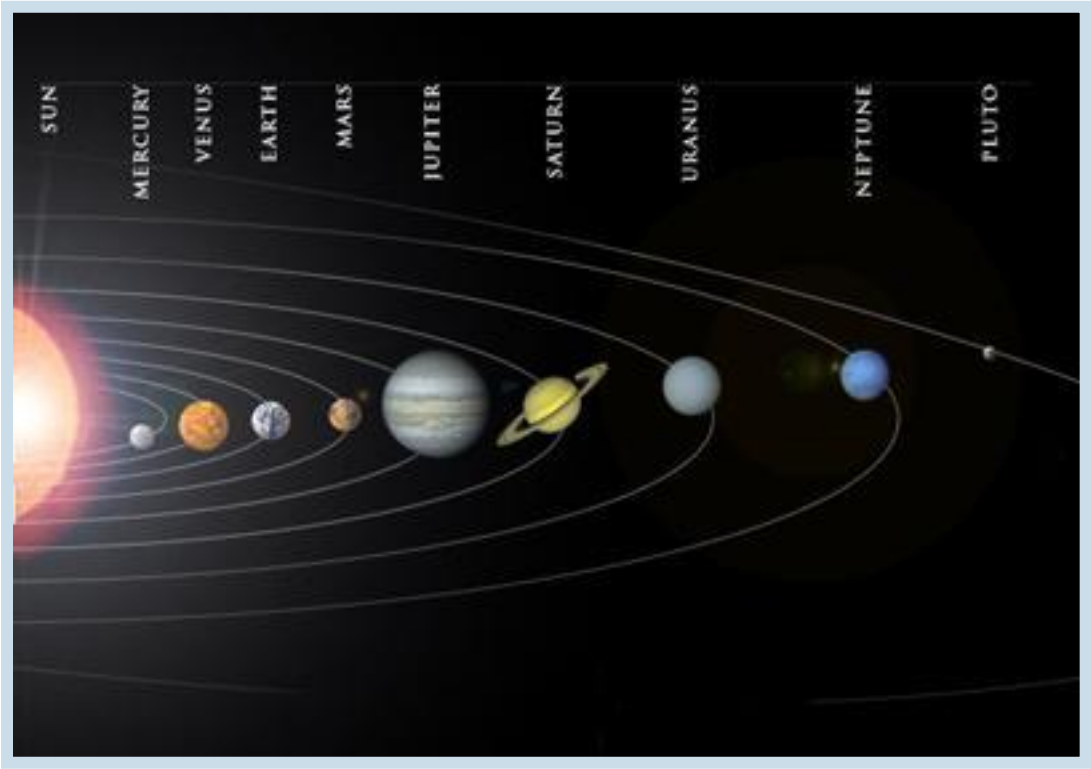
…, -4, -3, -1, 0, 1, 2, 3, 4,…



An ellipse is an oval-shaped curve.



Why?



Thanks to Johannes Kepler, we know that the planets move in an elliptic orbit with the sun at one focus.

https://www.iiesoc.in/

https://industrynetcouncil.org/

Basic Elliptical Curve Equation

An elliptic curve is a curve with equation

$$y2 = x3 + ax2 + bx + c,$$

where a, b, c are integers.

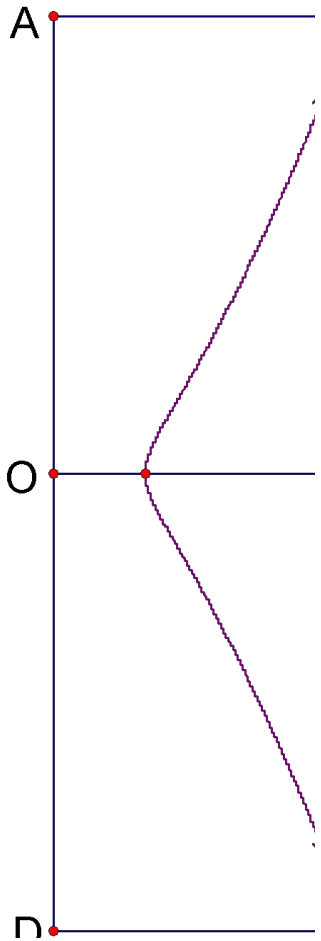
The name “elliptic” got attached to these curves, because in the past they were used to measure the lengths of the perimeters of ellipses and the lengths of planetary orbits which, as we noted, were ellipses.

Sample Elliptical Curve

- As we see from the following sketches of some elliptic curves for various values of a, b, c, these curves are not elliptic-shaped at all.
- In this Figure:

  a = -1, b = 0, c = 0.
- Remember the formula:

  $y2 = x3 + ax2 + bx + c$
- O is the origin (0, 0). The points A, B, C, D have coordinates (0, 5), (4, 5), (4, -5), (0, -5) respectively.



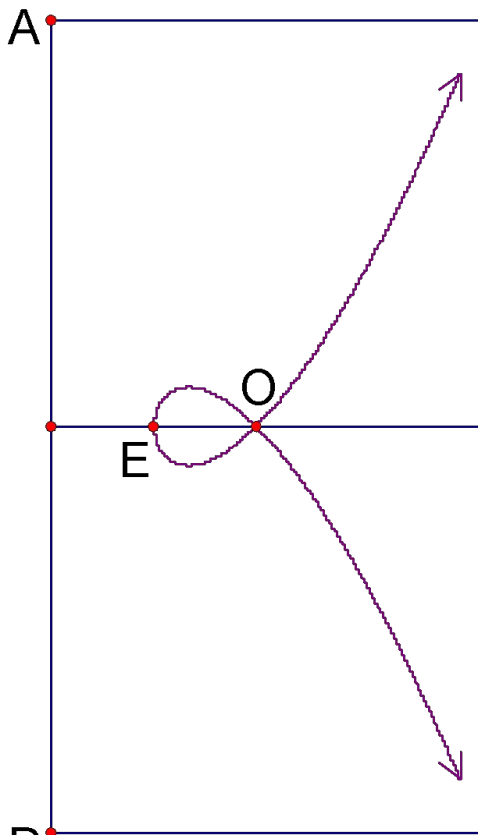
https://www.iiesoc.in/

https://industrynetcouncil.org/

Another Elliptical Curve

- In this figure, a = 1, b = 0, c = 0.

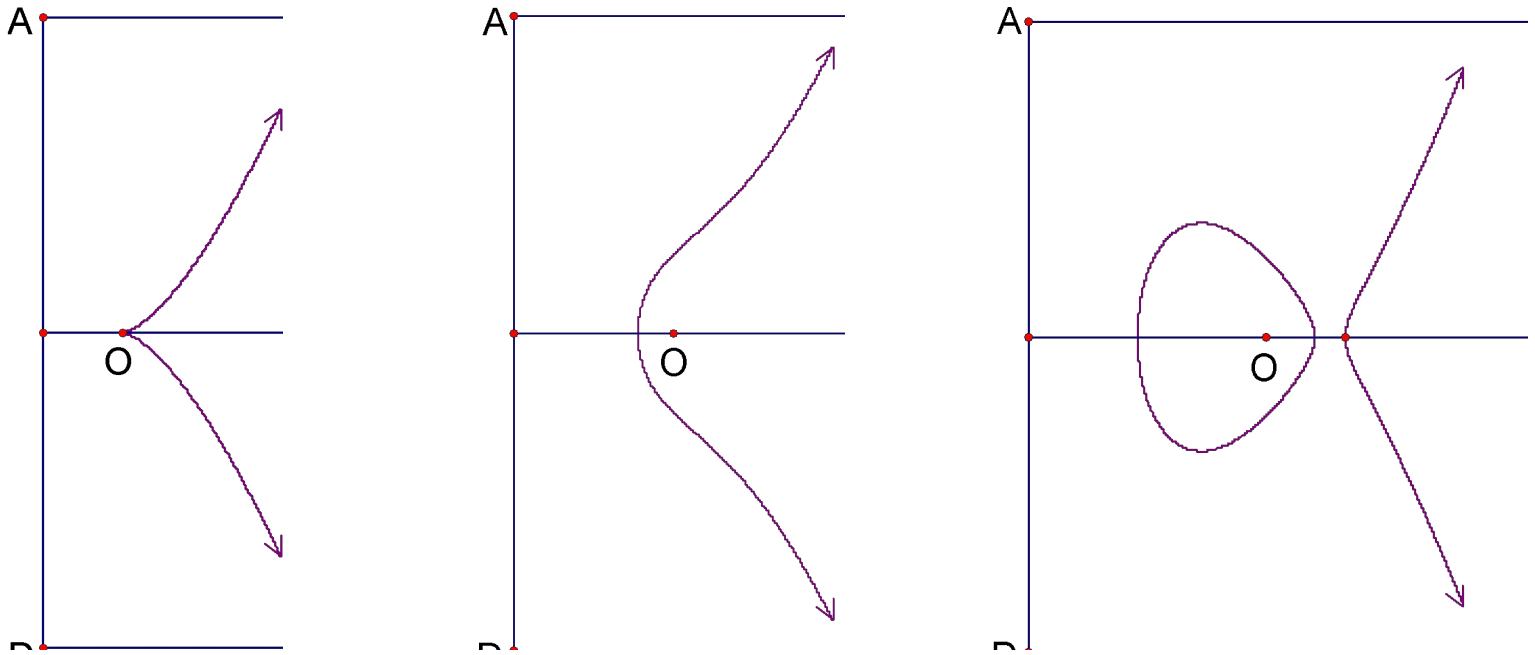
- Here, O is (0, 0). A, B, C, D, E are respectively (-2, 4), (3, 4), (3, -4), (-2, -4), (-1, 0).



https://www.iiesoc.in/

https://industrynetcouncil.org/

And Some More…..



https://www.iiesoc.in/

https://industrynetcouncil.org/

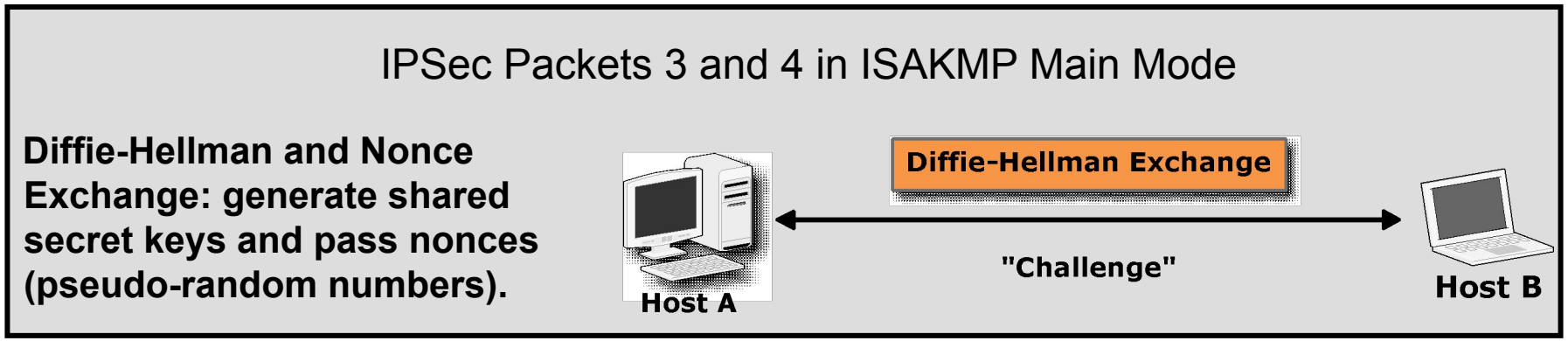
Why is ECC Secure?

- The main operation involved in elliptic curve cryptography is point multiplication, i.e., multiplication of any point on the curve by a scalar (i.e., a constant) to get another point on the curve. This is accomplished by two basic elliptic curve operations - point addition and doubling.

- It is claimed that the security of ECC depends on the ability to compute a point multiplication and the inability to compute the multiplicand given the original and product points.

Diffie-Hellman (-Merkle) Key Exhange

- Today, it is typical practice to use a symmetric system to encrypt the data and an asymmetric system to encrypt the symmetric keys.
- That is precisely what Diffie-Hellman is capable of doing – and does do when used for key exchange.
- Diffie-Hellman is not an encryption mechanism as we normally think of them, in that we do not typically use it to encrypt data -- it is a method to securely exchange the keys that encrypt data.
- Diffie-Hellman accomplishes this secure exchange by creating a “shared secret” (sometimes called a “key encryption key”) between two devices. The shared secret then encrypts the symmetric key (or “data encryption key” i.e. DES, Triple DES, CAST, IDEA, Blowfish, etc.) for secure transmittal.



Diffie-Hellman Ephemeral

- The Diffie-Hellman algorithm is often used to provide forward secrecy
- Key is never passed over the wire
- Both ends calculate the needed key
- The “key” is ephemeral or short-lived

How does Diffie Hellman work?

From Wikipedia:

1. Alice and Bob agree to use a modulus $p = 23$ and base $g = 5$ (which is a primitive root modulo 23). (Nalini's note: let's not get into what a primitive root modulo is. It is a special relationship between the two numbers.)

2. Alice chooses a secret integer $a = 6$, then sends Bob $A = g^a \bmod p$
   - $A = 5^6 \bmod 23 = 8$

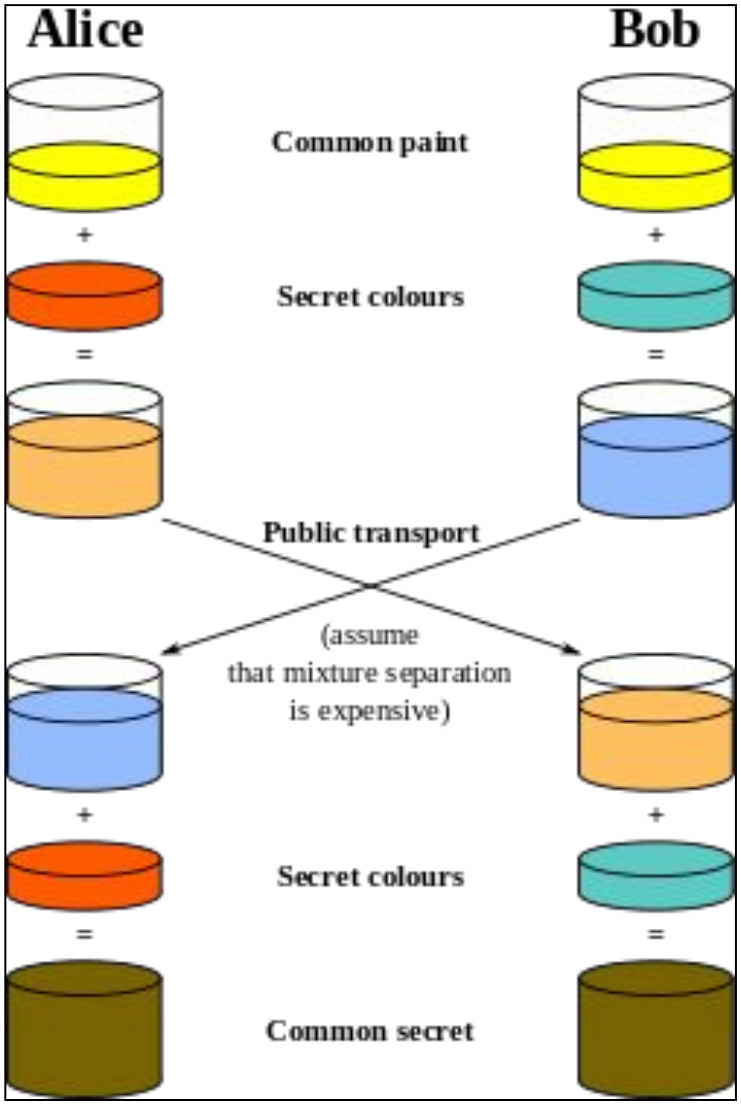
3. Bob chooses a secret integer $b = 15$, then sends Alice $B = g^b \bmod p$
   - $B = 5^{15} \bmod 23 = 19$

4. Alice computes $s = B^a \bmod p$
   - $s = 19^6 \bmod 23 = 2$

5. Bob computes $s = A^b \bmod p$
   - $s = 8^{15} \bmod 23 = 2$

6. Alice and Bob now share a secret (the number 2).

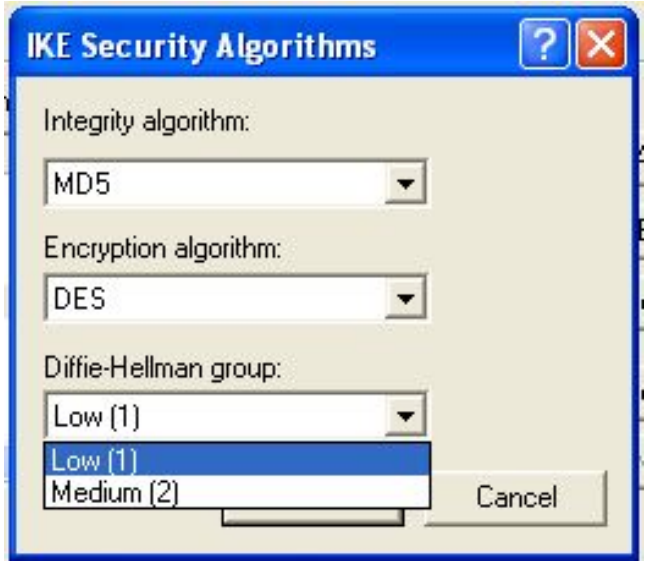
Diffie Hellman in color



From Wikipedia

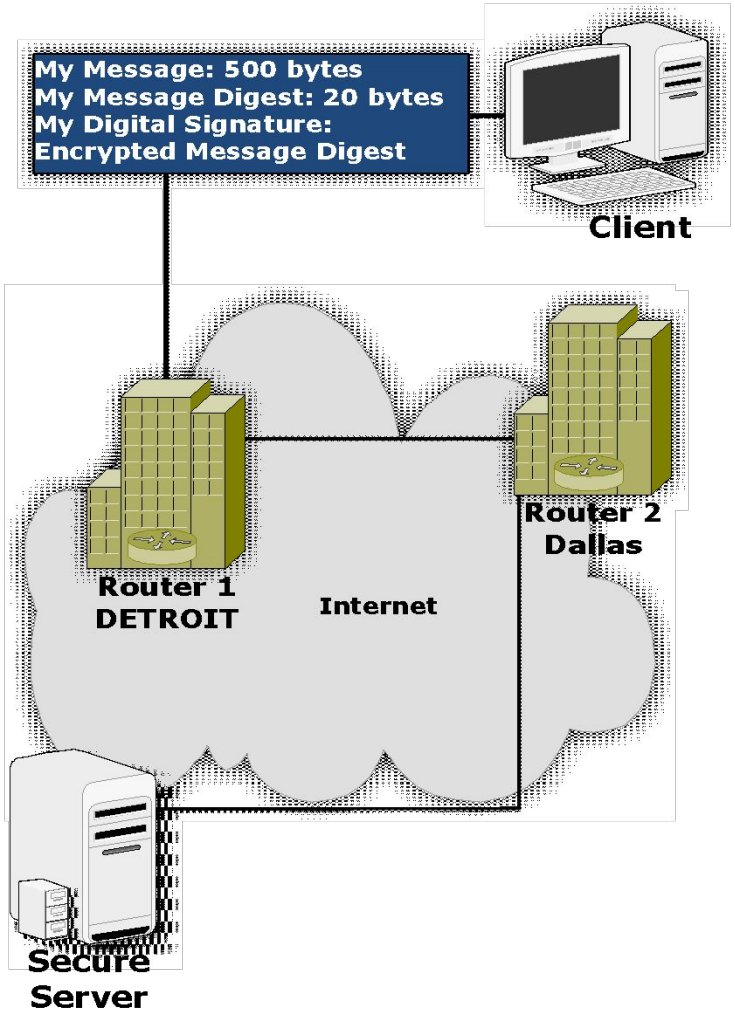
Diffie-Hellman Groups

- For example, in this example, using IPSec that you can specify which Diffie-Hellman groups to use.
- Diffie-Hellman groups are used to determine the length of the base prime numbers used during the key exchange process. The cryptographic strength of any key derived depends, in part, on the strength of the Diffie-Hellman group upon which the prime numbers are based.
- Group 2048 (high) is stronger (more secure) than Group 2 (medium), which is stronger than Group 1 (low). Group 1 provides 768 bits of keying strength, Group 2 provides 1024 bits, and Group 2048 provides 2048 bits. If mismatched groups are specified on each peer, negotiation fails. The group cannot be switched during the negotiation.
- The Diffie-Hellman group is configured as part of the IPSec ISAKMP main mode key exchange settings.



Digital Signatures

- Integrity is guaranteed in public-key systems by using *digital signatures*.
- A digital signature is a piece of data which is attached to a message and which can be used to find out if the message was tampered with during the conversation (e.g. through the intervention of a malicious user)
- The digital signature for a message is generated in two steps:
  - A *message digest* is generated.
  - The message digest is encrypted using the sender's *private* key. The resulting encrypted message digest is the *digital signature*.



https://www.iiesoc.in/

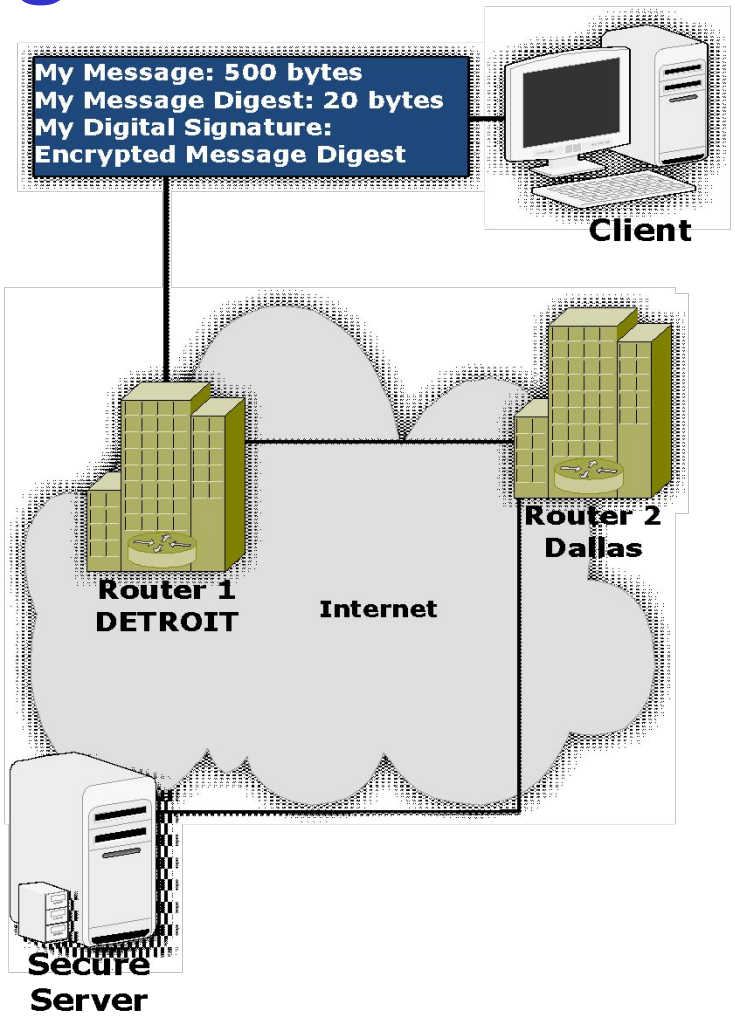
https://industrynetcouncil.org/

Message Digest

- The first step of the digital signature is to create a *message digest* or *one-way hash function.*
- A hash function is a computation that takes a variable-size input and returns a fixed-size string, which is called the hash value. If the hash function is one-way (that means hard to invert), it is also called a message-digest function.
- It is a digital fingerprint or summary of the larger document.
- A message digest has two important properties:

  (1) It is always smaller than the message itself and

  (2) Even the slightest change in the message produces a different digest.

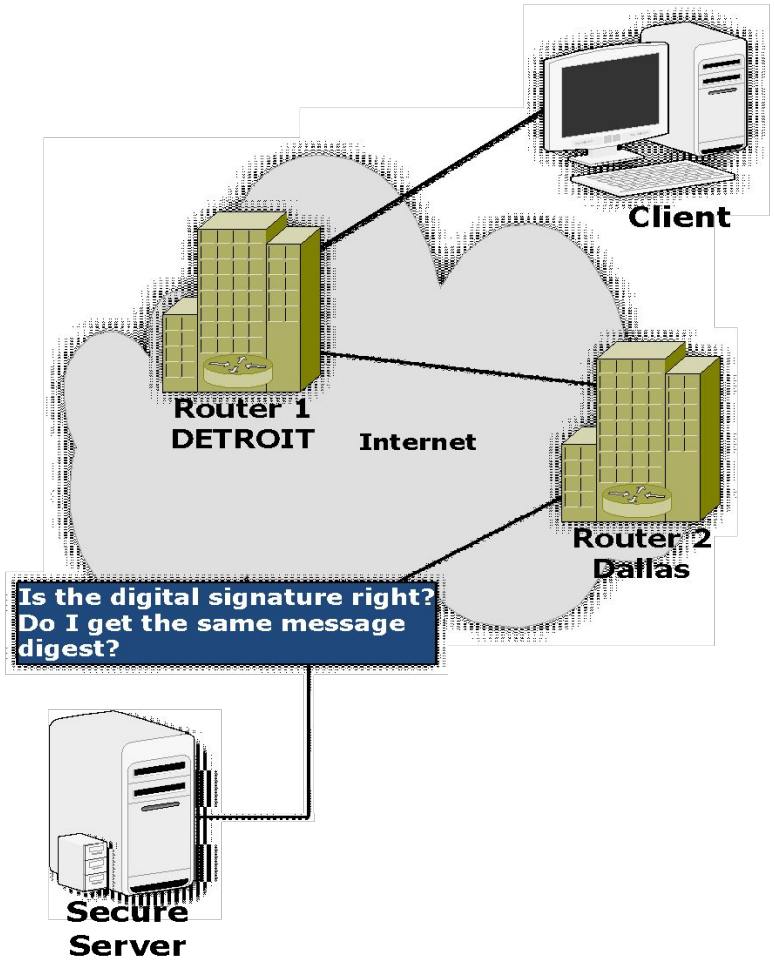


https://www.iiesoc.in/

https://industrynetcouncil.org/

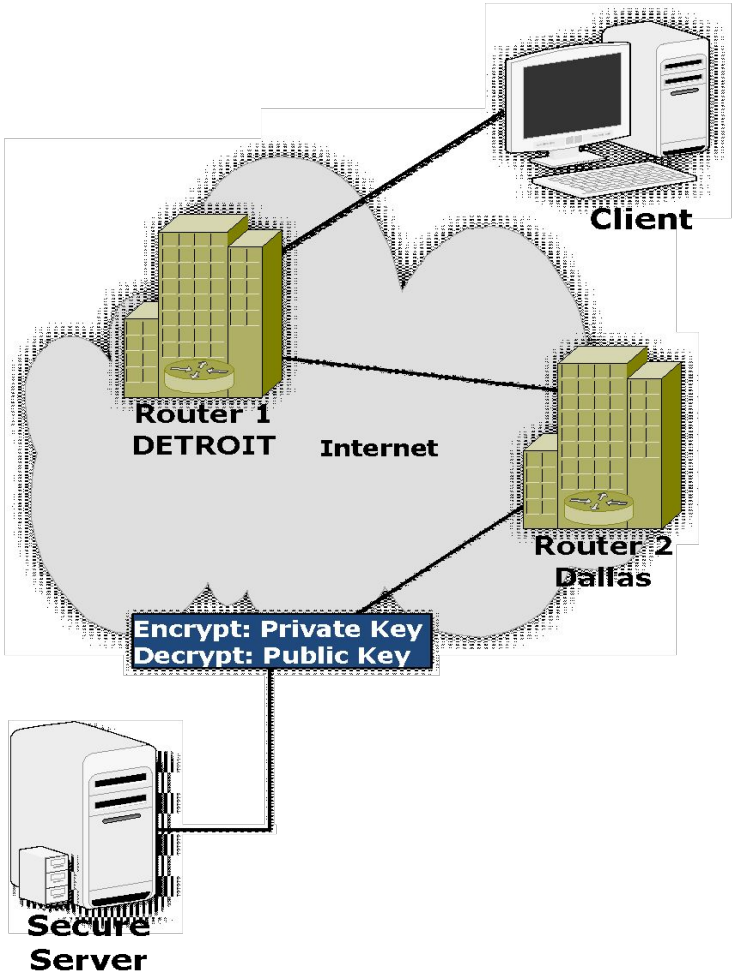
Receiver of Digital Signature

- The digital signature is attached to the message, and sent to the receiver.
- The receiver then does the following:
  - Using the sender's public key, decrypts the digital signature to obtain the message digest generated by the sender.
  - Uses the same message digest algorithm used by the sender to generate a message digest of the received message.
  - Compares both message digests (the one sent by the sender as a digital signature, and the one generated by the receiver).



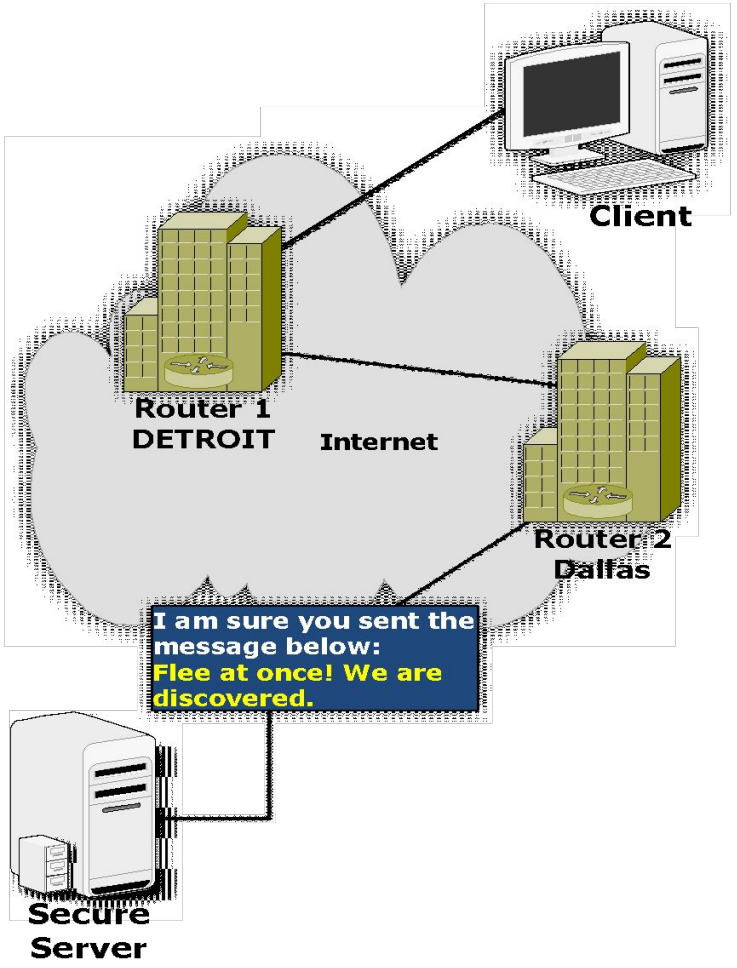
Message Digest Comparison

- If the message digest comparison shows that they are not exactly the same, the message has been tampered with by a third party.
- We can be sure that the digital signature was sent by the sender (and not by a malicious user) because only the sender's public key can decrypt the digital signature (which was encrypted by the sender's private key.
- Remember that what one key encrypts, the other one decrypts, and vice versa).
- If decrypting using the public key renders a faulty message digest, this means that either the message or the message digest are not exactly what the sender sent.



Message Digest Ensures Integrity

- Using public-key cryptography in this manner ensures integrity, because we have a way of knowing if the message we received is exactly what was sent by the sender.
- However, notice how the above example guarantees only integrity.
- The message itself may be sent in clear text.
- This is not necessarily a bad thing: in some cases we might not be interested in keeping the data private, we simply want to make sure it isn't tampered with.
- To add privacy to this conversation, we would need to encrypt the message.



Message Digest Algorithms

- MD2, MD4, and MD5 are message-digest algorithms developed by Rivest.
- All three algorithms take a message of arbitrary length and produce a 128-bit message digest.
- MD2 is for 8-bit machines, MD4 and MD5 are aimed at 32-bit machines. All are Internet RFCs.
- MD2 was developed by Rivest in 1989. It is considered generally secure – immune to cryptanalysis.
- MD4 was developed by Rivest in 1990 and is now considered broken.
- MD5 was developed by Rivest in 1991. It is basically MD4 with "safety-belts" and while it is slightly slower than MD4, it is more secure. The algorithm consists of four distinct rounds and has a slightly different design from that of MD4.
- MD5 is also considered broken. SHA-2 family is recommended instead.

Message Digest Algorithms:

- How to create a message digest
- 128-bit message digest
- MD2: 8-bit
- MD4: Broken
- MD5: Broken

Secure Hash Algorithm

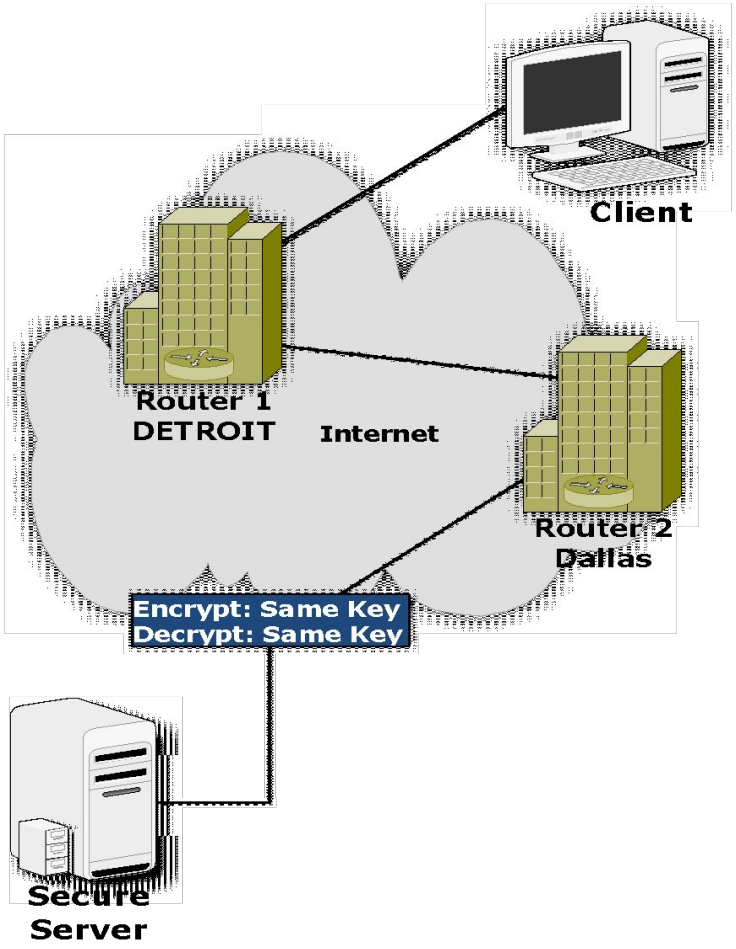
- The Secure Hash Algorithm (SHA), the algorithm specified in the Secure Hash Standard (SHS, FIPS 180), was developed by NIST.
- SHA-1 is a revision to SHA that was published in 1994; the revision corrected an unpublished flaw in SHA. Its design is very similar to the MD4 family of hash functions developed by Rivest.
- The algorithm takes a message of less than 264 bits in length and produces a 160-bit message digest. The algorithm is slightly slower than MD5 but the larger message digest makes it more secure against brute-force collision and inversion attacks.
- SHA-1 is often used to verify digital signatures but it should not be used.
- SHA-2 family consists of six hash functions with digests (hash values) that are 224, 256, 384 or 512 bits: SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256. SHA-2 family should be used.

Secure Hash Algorithms:

- Another way to create a message digest
- 160-bit message digest

Message Authentication Code (MAC)

- An alternative to a digital signature is a message authentication code (MAC).
- A MAC is an authentication tag (also called a checksum) derived by applying an authentication scheme, together with a secret key, to a message.
- Unlike digital signatures, MACs are computed and verified with the same key, so that they can only be verified by the intended recipient.
- There are four types of MACs: (1) unconditionally secure, (2) hash function-based, (3) stream cipher-based, or (4) block cipher-based.

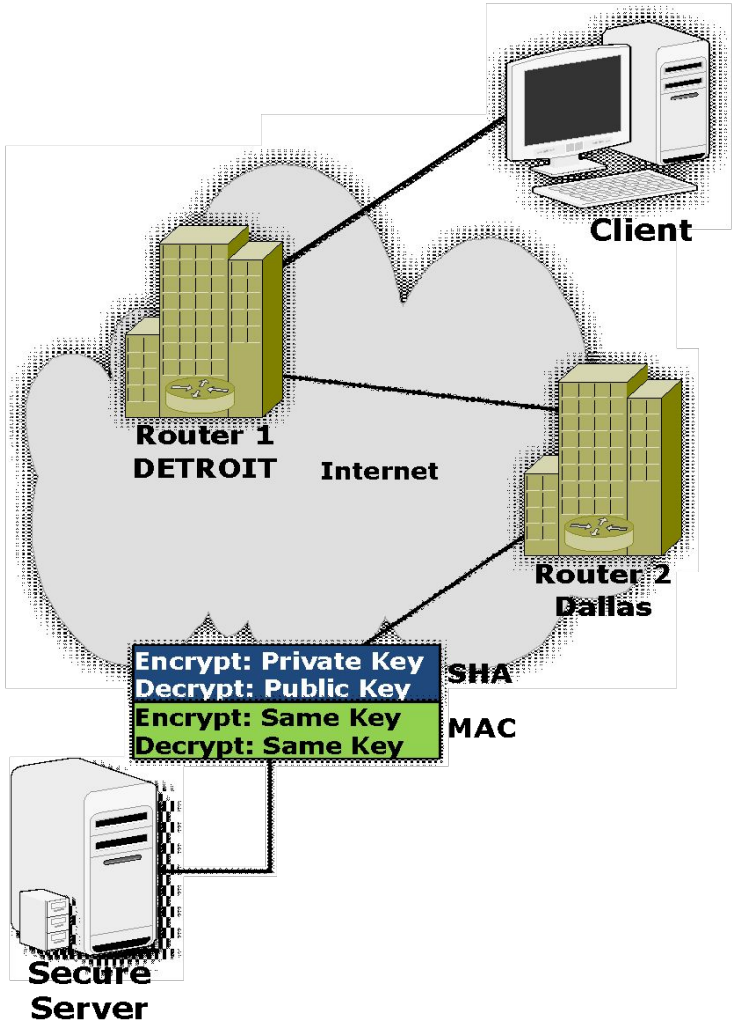


https://www.iiesoc.in/

https://industrynetcouncil.org/

HMAC

- Hash function-based MACs (often called HMACs) use a key or keys in conjunction with a hash function to produce a checksum that is appended to the message.
- Examples are the keyed-MD5 method of message authentication called HMAC-MD5 and HMAC-SHA1.
- HMAC (RFC-2104) is often used in conjunction with a SHA-n algorithm.

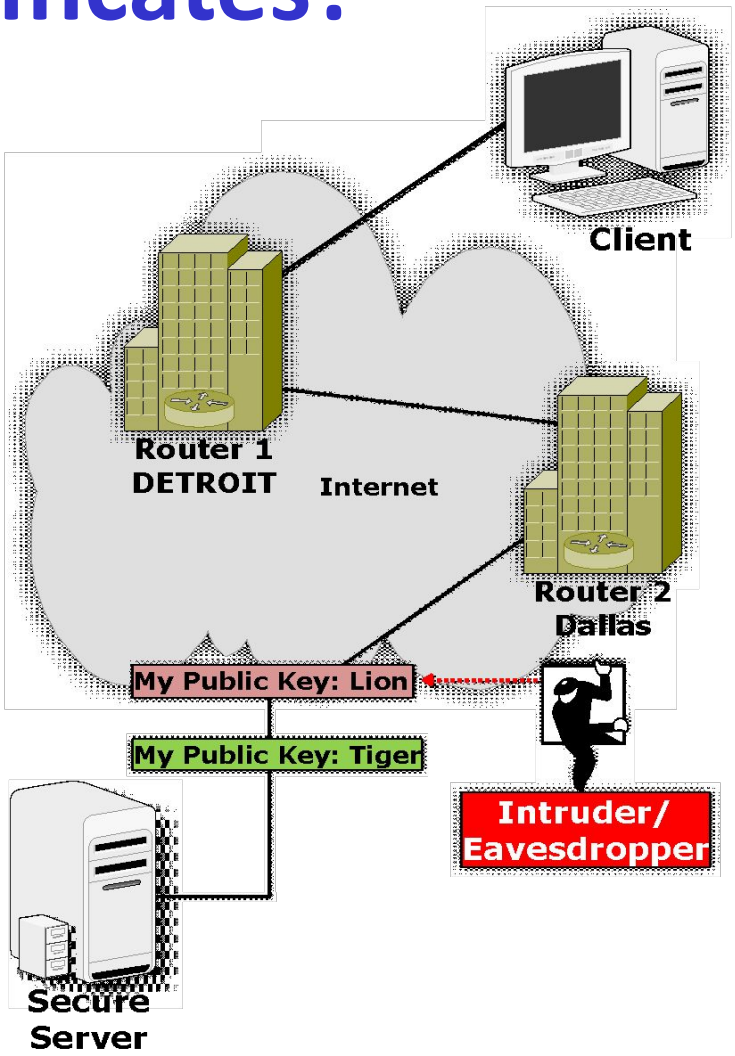


https://www.iiesoc.in/

https://industrynetcouncil.org/

Why Digital Certificates?

- Everything we have been talking about does guarantee, to a certain extent, the authenticity of the sender since only the sender's public key can decrypt the digital signature (encrypted with the sender's private key).
- However, the only thing this guarantees is that whoever sent the message has the private key corresponding to the public key we used to decrypt the digital signature.
- Although this public key might have been advertised as belonging to the sender, how can we be absolutely certain?
- Maybe the sender isn't really who he claims to be, but just someone impersonating the sender.
- When you want to require that there is absolutely no doubt about a user's identity, you use a digital certificate.



Digital Certificates

- A digital certificate is a digital document that certifies that a certain public key is owned by a particular user. This document is signed by a third party called the certificate authority (or CA).
- Of course, the certificate is encoded in a digital format.
- The important thing to remember is that the certificate is signed by a third party (the certificate authority) which does not itself take place in the secure conversation.

- The signature is actually a digital signature generated with the CA's private key.
- Therefore, we can verify the integrity of the certificate using the CA's public key.

https://www.iiesoc.in/

https://industrynetcouncil.org/

Certificate Authorities

- Now, how can you trust the certificate? To be more exact, how can you trust the CA that signs the certificate.
- Believe it or not, there are no fancy algorithms to decide when a CA is trustworthy... you must decide by yourself whether you trust or don't trust a given CA.
- This means that the public-key system you use will generally have a list of 'trusted CAs', which includes the digital certificates of those CAs you will trust (each of these certificates, in turn, include the CA's public key, so you can verify digital signatures).
- You have to decide.

Well-known Certificate Authorities

- Some CAs are so well known that they are included by default in many public-key systems (for example, web browsers usually include VeriSign and GlobalSign certificates, because many websites use certificates issued by those companies to authenticate themselves to web browsers).
- Of course, you can add other CAs to the 'trusted list'.
- For example, if your department sets up a CA, and you trust that the department's CA will only issue certificates to trustworthy people, then you could add it to the list.

https://www.iiesoc.in/

https://industrynetcouncil.org/

X.509 Certificate Format

- The format in which digital certificates are encoded is called the X.509 certificate format.
- An X.509 certificate is a plain text file which includes a lot of information in a very specific syntax (which is beyond the scope of this class!)
- The four most important things we can find in an X.509 certificate:
  - **Subject**: This is the 'name' of the user. It is encoded as a *distinguished name* (the format for distinguished names will be explained next)
  - **Subject's public key**: This includes not only the key itself, but information such as the algorithm used to generate the public key.
  - **Issuer's Subject**: CA's distinguished name.
  - **Digital signature**: The certificate includes a digital signature of all the information in the certificate. This digital signature is generated using the CA's private key. To verify the digital signature, we need the CA's public key (which can be found in the CA's certificate).

X.509 Certificate Sample

- An X.509 certificate contains the user name, CA's name, public key, and CA's signature.
- Of course, the certificate, does not include the private key, which must be kept separate from the public key.
- Remember that the certificate is a public document we want to be able to distribute to other users so they can verify our identity, so we don't want to include the private key (which must be known only by the owner of the certificate).
- When we have both a certificate and its associated private key, these two items are generally referred to as the user's credentials.

X.509 Distinguished Names

- Names in X.509 certificates are not encoded simply as 'common names', such as “Glen Varanasi", “Megan Browning", "Certificate Authority XYZ", or "Systems Administrator".
- They are encoded as *distinguished names*, which are a comma-separated list of name-value pairs.
- For example, the following could be our distinguished names:
  - **O=Inside Products, OU=Sales, CN=Harry Meetsquota**
  - **O=Big University, OU=Computer Science, CN=Jane Verysmart**
- So what do "O", "OU", and "CN" mean? A distinguished name can have several different attributes, and the most common are the following:
  - **O** : Organization
  - **OU** : Organizational Unit
  - **CN** : Common Name (generally, the user's name)
  - **C** : Country

Certificate Revocation List (CRL)

- The CRL generally lists revoked certificates, along with the reason(s) for revocation.
- The CRLs are checked by relying parties to verify that a user's certificate has not been revoked.
- The fields in a CRL identify the issuer, the date the current CRL was generated, the date by which the next CRL will be generated, and the revoked users' certificates.
- The problem with CRLs, of course, is that they have to be maintained.
- A newer protocol, Online Certificate Status Protocol (OCSP), is described in RFC2560. It does a similar function in real time but this has to be supported by your platform.
- CAs use CRLs to publicize the revocation of a subject's certificate. If a CA issues a large number of certificates, then the use of distribution points is encouraged in order to ensure that CRLs do not become too large.

Changes for Post Quantum

- Need cipher suites
- Need TLS changes (key exchange)
- Need certificate / signing changes
- Need implementation in crypto libraries (OpenSSL, etc.)
- Need changes to compilers (Java, C++, etc.)
- Need changes to web servers (Apache), data base servers, etc.
- Need to change application programs

Upcoming Sessions

Fundamentals of Cryptography: session #3 May 18:

11 am Eastern, 8:30 pm India

Very next session! IPv6!

IPv6 webinar (EH-PDM results): May 4:

11 am Eastern 8:30 pm India

Questions?

Contact:

info@iiesoc.in

president@industrynetcouncil.org