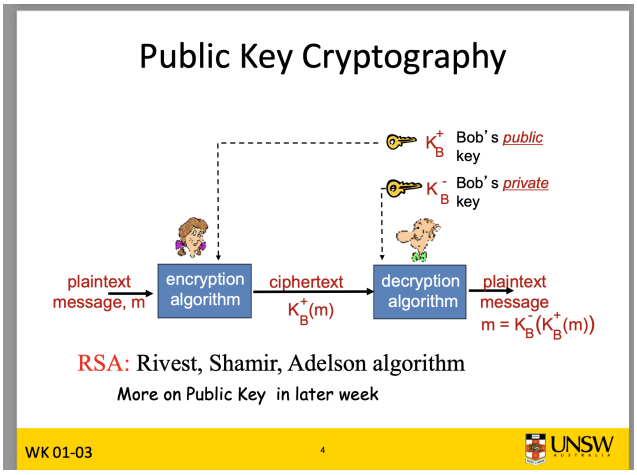
Week1:

DoS/ DDoS

Network Security Attacks

- Denial of Service (DoS) : attacker sends an avalanche of bogus packets to a server to keep the server constantly busy or clog up the access link
- Distributed DoS (DDoS): attack a large number of compromised devices (bots)
  - Mirai is an example of DDoS in 2016, compromised Linux-based IP cameras, utility meters, home routers and others
  - Done by exploiting weak authentication configurations including use of default passwords
- In IP spoofing attacks: impersonate as an authorised user by crafting a packet with forged IP address and adjusting certain other fields to make it look legitimate

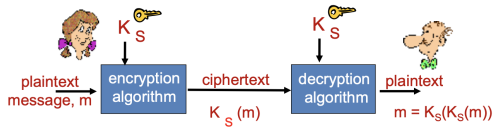
RSA is **asymmetric**



RSA is intended to provide confidentiality of the message. It can't guarantee integrity. The attacker can replace the encrypted message without knowing it.

Symmetric key cryptography: Alice and Bob need to agree on key value.

Symmetric Key Cryptography



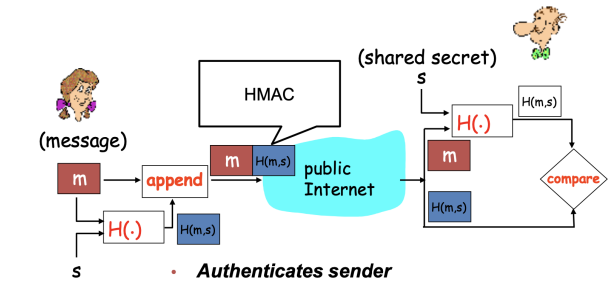
symmetric key crypto: Bob and Alice share same (symmetric) key: K_S

Q: how do Bob and Alice agree on key value?

HMAC:

HMAC takes a message and a secret key as inputs, then uses a hash function to produce a fixed length output that serves as a message authentication code. The output is then sent along with the message and can be used to verify the integrity and authenticity of the message.

HMAC: Integrity and Authentication



- ***Authenticates sender***
- ***Verifies message integrity***
- No encryption ! – Confidentiality not always needed
- Also called "keyed hash"
- Please note that HMAC take both message and key as input, not a simple concatenation.

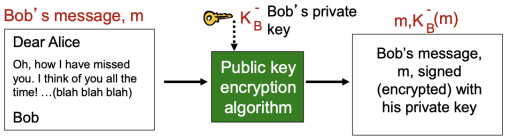
HMAC: Integrity only, not confidentially.

Digital signature -

Digital signatures

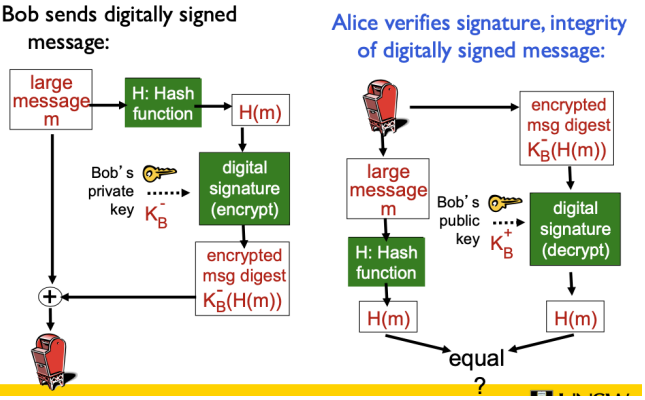
simple digital signature for message m:

- Bob signs m by encrypting with his private key K_B^-, creating "signed" message, $K_B^-(m)$



Bob can use a hash function to encrypt m before sending it. Bob also sends K(-B) of the encrypted message (i.e.: uses his private key to sign the hashed message).
Alice verifies K(+B)(K(-B)(hashed(m))) == hashed(m).
Alice decrypts the hashed message.

Digital signature = signed message digest



Two types of symmetric ciphers

- Block ciphers
  - Break plaintext message in equal-size blocks
  - Encrypt each block as a unit
  - Used in may Internet protocols (PGP-secure email, SSL (secure TCP), IPsec (secure net-transport layer)
- Stream ciphers
  - encrypt one bit at time
  - Used in secure WLAN

Block ciphers:

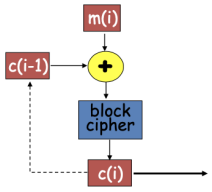
CBC: Sender

- *cipher block chaining:* XOR ith input block, $m(i)$, with previous block of cipher text, $c(i-1)$
  - $c(0)$ is an Initialisation Vector transmitted to receiver in clear
  - First block:

$c(1) = K_S(m(i) \oplus c(0))$

  - *Subsequent blocks:*

$c(i) = K_S(m(i) \oplus c(i-1))$



CBC: Receiver

- How to recover $m(i)$?
  - Decrypt with *Ks* to get $s(i) = Ks(c(i)) = m(i) \oplus c(i-1)$
  - Now the receiver knows $c(i-1)$, it can get $m(i) = s(i) \oplus c(i-1)$
  - **IV** sent only once
  - Intruder can't do much with **IV** since it doesn't have Ks
  - CBC has important consequence for designing secure network protocols

AES, DES are blocked ciphers. **Both are symmetric.**
AES: 128 bits data block, while DES: 64 bits block.
AES is 6x faster than DES.

DES uses a 56 bits key. AES uses 192 or 192 or 256 bits keys.

AES confidentiality modes: CBC, ECB, …
ECB mode: plain texts are divided into blocks of the same sizes and encrypted.

Week 2:

Stream ciphers:
Combine each byte of keystream with byte of messages to get cipher text:

m(i) = i th unit of message
ks(i) = i th unit of keystream
c(i) = i th unit of ciphertext
c(i) = ks(i) ⊕ m(i) (⊕ = exclusive or)
m(i) = ks(i) ⊕ c(i)

WEP:

Encryption:

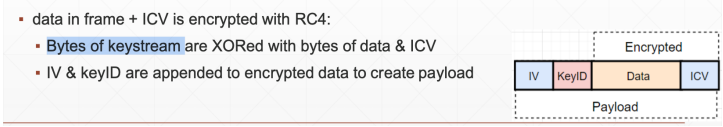
(pre-shared key for both side is the WEP password entered by the user)

For data integrity, ICV is calculated over data (CRC-32) and appended to the end.

128-bit key = 24-bit IV + 104-bit (above key)

Use **RC4** on the key to generate key-stream.

RC4 is **symmetric**, like AES and DES.



Makes IV plain text, but ICV is encrypted.

Decryption:

The receiver retrieves IV, and it knows the shared-secret-key, then it gets the key stream.

XOR the key stream with the encrypted texts.

Verify the data with ICV.

Breaking WEP:

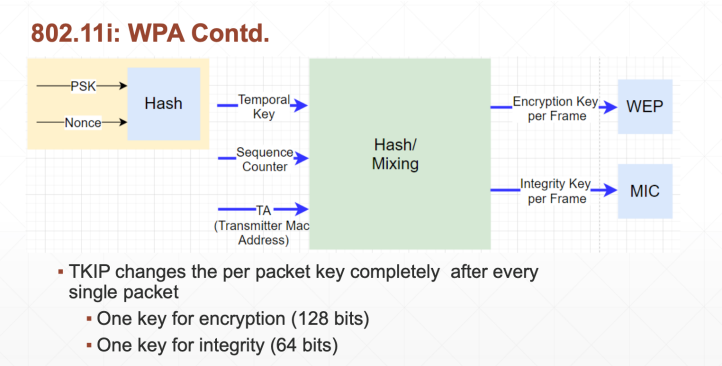
24 bits IV is limited - IV reused

WPA - solve some weakness of WEP

(uses different encryption key per frame)

WPA is much more safe than WEP as follows:

- IV = 48 bits
- IV used as a sequence counter, together with the temporal key, this also prevents replay attack
- RC4 key = f(WPA key || IV)



RSA:

K+(K-(m)) = K-(K+(m))

DH:
It's not preventing MITM,
Alice, Bob determines p (very big prime) and g (base).

Week 3:

It's essential to identify the key's owner.

RA (registration authority) verifies the identity of the entity
CA (certificate authority) issues and signs the digital certificates.
VA (validation authority) verifies the digital certificates.

RA verifies the identity of the entity.
Alice requests for a certificate:

1) Alice provides "proof of identity" to RA.
2) RA verifies the id and gives approval to CA.

CA binds a public key to a particular entity.
Alice registers her public key with CA:

1) Alice provides proofed identity and her PK.
2) CA creates certificate binding Alice to its PK.

CA uses the private key to encrypt Alice's PK.

VA (on receiver's side)
Bob verifies the certificate from Alice:

1) Sending it to VA.
2) VA verifies the signature of certificates and sends the result back to Bob.

Lifecycle of a certificate:
Enrollment, issuance, validation, revocation, renewal

Certificate revocation list

X.509 formats:
The format standard of the public key certificate.

X.509 Formats

Definition:

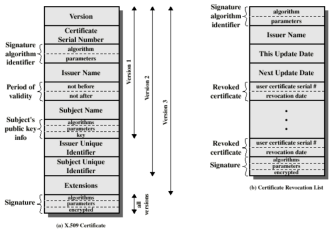
- The format standard of the public key certificate.

Usage:

- Network protocols, SSL/TLS;
- Electronic signature service.

Key ingredients:

- Public key;
- Identity information;
- Signature information;
- Certificate revocation list;
- Certificate validity verification algorithm.



Kerberos
"Never store passwords in the server"

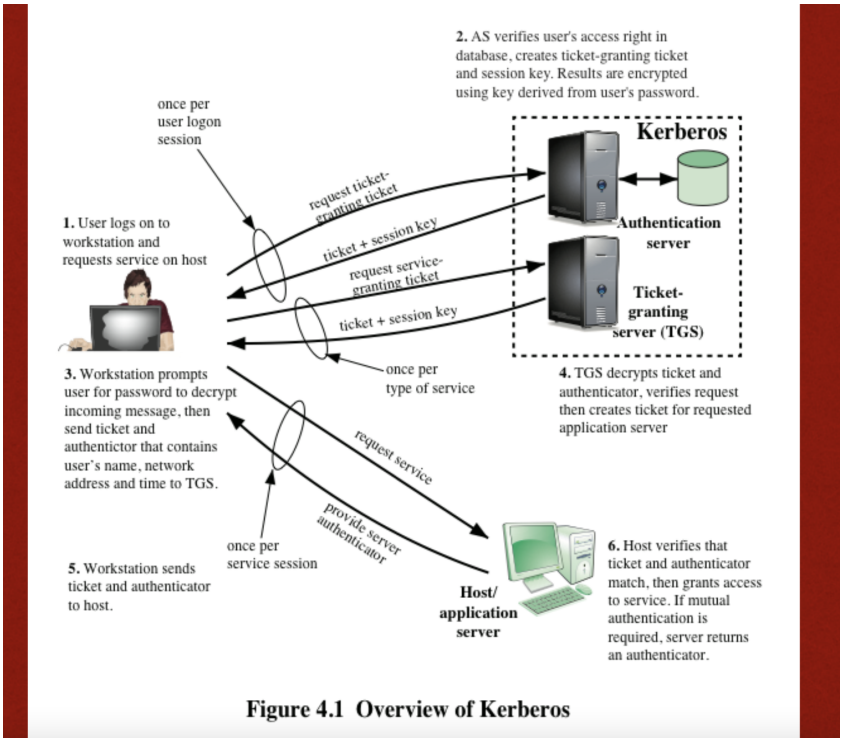


Figure 4.1 Overview of Kerberos

(1) client & authen server (AS): client says they want to get services from the server, and they get a TGT from the AS after their ID is verified. The results (session key + TGT) are encrypted with the key derived from the client's password.
(2) Client & ticket-granting server (TGS): client shows the TGT received from the AS, and TGS gives to the client the (service ticket (ST) + session key).
(3) The client's workstation sends a ticket and authenticator to the host server, who verifies the ST and authenticator match.

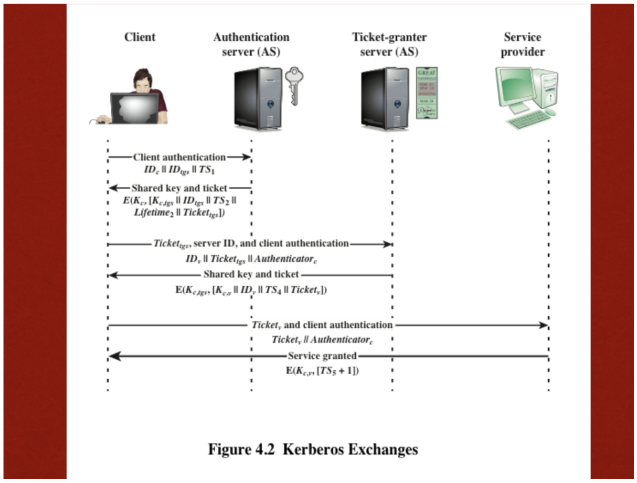


Figure 4.2 Kerberos Exchanges

(1) ID_c: tells AS client's identity, ID_tgs: tells the AS that request for TGS
(2) K_(c, tgs): copy of session key accessible to client created by AS to permit secure exchange between the client and the TGS, TS_2: timestamp of the ticket, Ticket_tgs: the ticket sends to TGS for client's access

(3) ID_v: tells TGS that I want server v
(4) K_(c, tgs):

TABLE 4.1
SUMMARY OF KERBEROS VERSION 4 MESSAGE EXCHANGES

(1) C → AS $ID_c \| ID_{tgs} \| TS_1$
(2) AS → C $E(K_c, [K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}])$
$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$

(a) Authentication Service Exchange to obtain ticket-granting ticket

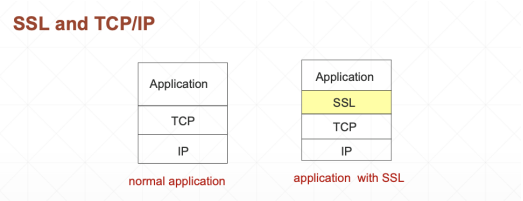
(3) C → TGS $ID_v \| Ticket_{tgs} \| Authenticator_c$
(4) TGS → C $E(K_{c,tgs}, [K_{c,v} \| ID_v \| TS_4 \| Ticket_v])$
$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$
$Ticket_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$
$Authenticator_c = E(K_{c,tgs}, [ID_C \| AD_C \| TS_3])$

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5) C → V $Ticket_v \| Authenticator_c$
(6) V → C $E(K_{c,v}, [TS_5 + 1])$ (for mutual authentication)
$Ticket_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$
$Authenticator_c = E(K_{c,v}, [ID_C \| AD_C \| TS_5])$

(c) Client/Server Authentication Exchange to obtain service

SSL:



Ssl handshake:

1. Client sends a list of algorithms it supports and the client nonce.
2. Server sends back: algorithm + server nonce + server certificate.
3. Client verifies the secret to extract the server's public key, generates the pre_master_secret and encrypts it with server's public key and sends it back to the server.
4. Client and server independently compute encryption and MAC keys from pre_master_secret and `nonces`
5. Client sends a MAC of all the handshake messages
6. Server sends a MAC of all the handshake messages

In step4, there are Kc, Mc, Ks, Ms generated. Different nonces will make them different. The 4 are derived from the key derivation function, shared secret and 2 nonces.

Ms, Mc are different, but the client sends Mc(handshake messages) to the server and the server can verify with Ms(handshake messages received from the client)??

Tls protocol exchange (happens after TCP 3 handshakes)

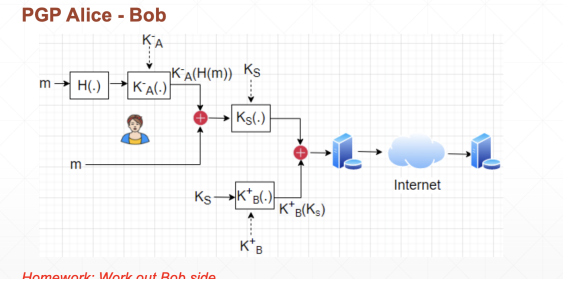
1. Client hello: message contains available ciphers, hash functions and a large random number (nonce).
2. Server hello: it's from the server to the client, along with the choice of ciphers and hashes. The message also contains the server's public key certificate and the nonce.
3. Client checks the validity of the server's certificate and initiates clientKeyExchange message.
4. The server sends a changeCipherSpec and a finished message to indicate that the key generation and authentication are completed.
5. The server also has the shared key now and responds with the ChangeCipherSpec and finished message.
6. The client decrypts messages with the symmetric key and checks integrity.

For ssl/tls:
Asymmetric for authentication: the client receives the server's public key, uses the p+ to encrypt the pre_master_secret.

Symmetric for confidentiality:

MAC for integrity: sending all handshake

Nonce:
Included in serverHello/ clientHello
It's 32 bytes. According to the lecture, nonce is sent by the server in ServerHello. It's used to prevent an attacker from replaying records in the future.
If there were no nonce when calculating the MAC keys, the attacker might capture the packet from the client to the server, and send the packet to the server to impersonate the client.



PGP:

Alice (sender)

1. Uses sha-3 to calculate hash of the m and encrypts it with her Ka-
2. Concatenate the message with the hash value, the content now is encrypted with a shared key Ks
3. She uses Bob's Kb+ to encrypt Ks
4. She sends result of step 2, 3 to Bob

Bob (receiver)

1. Bob uses his Kb- to decrypt Alice's result in step 3 to get the Ks
2. Bob uses Ks to decrypt Alice's result in step 2 to get Ka-(concat(m, hash(m)))
3. Bob uses Alice's Ka+ to get concat(m, hash(m))
4. Bob calculates hash(m) to compare with hash(m) received from Alice for integrity.

Week 4:

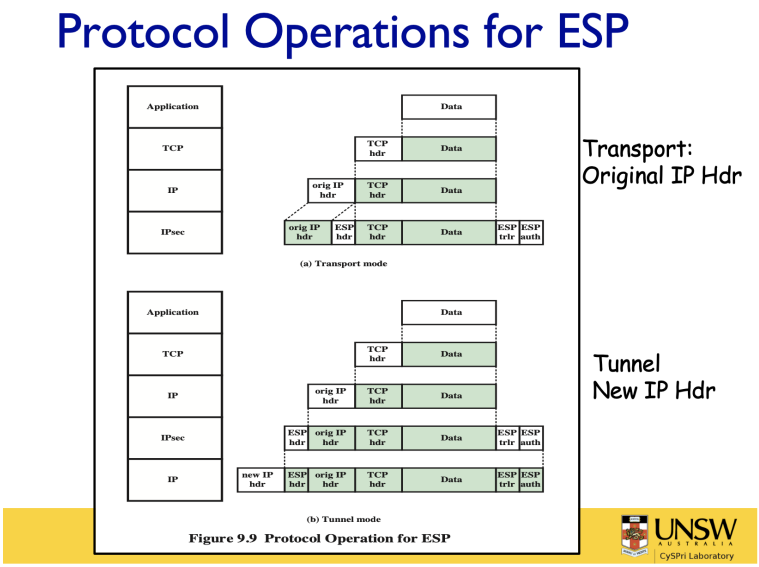
DNS spoofing: it is when an attacker falsifies the DNS entries in a DNS server's cache. By modifying the DNS cache with wrong information, for example, change the IP address of domain name "Google" from the correct and true address to an IP of a malicious website.

Evil twin:
In this attack, the attacker is hosting a fake AP impersonating the real one.

Transport mode:
Protect all data (payload) after the original IP header. Insert IPSec header after the original IP header.

Tunnel mode:
All of IP including IP header is encapsulated. New IP header is added by firewall/routers.

Tunnel ESP vs Transport ESP:



Orig IP hdr is not encapsulated in transport mode.

Before sending data, SA established from the sender to the receiver. Endpoints hold SA state in the security association database.

R1 stores for SA
32 bits SA identifier: SPI
Origin SA interface
Destination SA interface
Type of encryption used
Encryption key
Type of integrity check used
Authentication key

Example SA

SPI: 12345
Source IP: 200.168.1.100
Dest IP: 193.68.2.23
Protocol: ESP
Encryption algorithm: 3DES-cbc
HMAC algorithm: MD5
Encryption key: 0x7aeaca…
HMAC key:0xc0291f…

When the receiver receives the IPsec datagram, it checks the SPI against its SAD, and processes the datagram accordingly.

R2 stores for SAD
SAD parameters
SPI
Sequence number counter
Anti-replay window
AH information
ESP information
Lifetime of this security association
IPsec protocol mode
Path MTU

SAD: how to decrypt the IPsec datagram
SPD: for a given datagram, sending entities need to know if it should use IPSec

IKEv2 (internet key exchange)

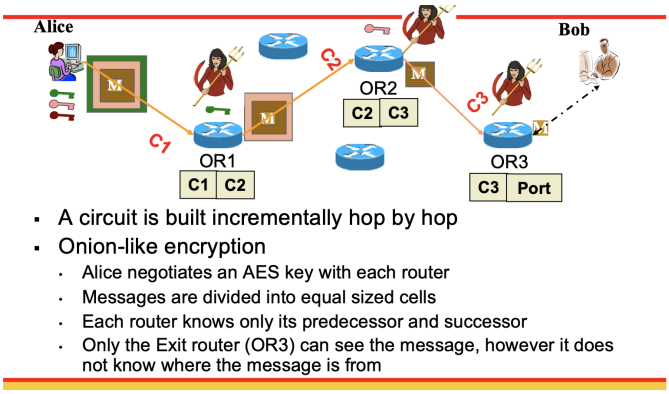
Anonymity
Tor: TCP based, circuit routing.

Components:
Client/OP: the user of the Tor network.
OP fetches directories and creates virtual circuits on the network on behalf of the users.

Destination server:
The target TCP applications

Tor router (Onion router) :
Relays the application data

Directory server:
Holds database of current active ORs



Cells
I2P
Sender only knows about the inbound gateway of the receiver.

Week 5:

OSI 7 layer model, 802.1X is implemented on data link layer

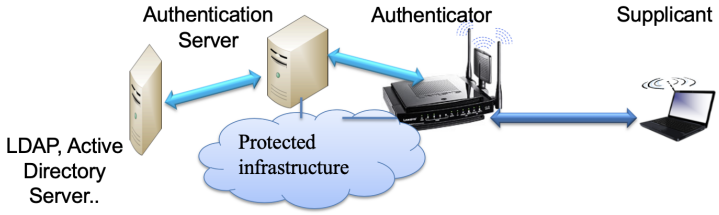
1. Physical layer
2. Data link layer
3. Network layer
4. Transport layer
5. Session layer
6. Presentation layer
7. Application layer

802.1X (wireless)
Port based authentication
Port based - users must authenticate to switch what they are connecting with

Authentication server:
Needs to check the credentials using the RADIUS server, kerberos server, ldap or active directory server

Return one of {access accept, access reject, access challenge for extra credentials}

- Involves 3-party communications (nomenclature from 802.1X standard)
  - Supplicant
    - User
  - Authenticator
    - Ethernet switch, wireless access point
  - Authentication server
    - RADIUS (Remote access dial-in user service) database, Kerberos, LDAP or AD (Can be co-located with Authenticator)



Authen server (uses radius, ldap, kerberos, AD…)
A server that verifies the identity of users or devices who are attempting to access the network.

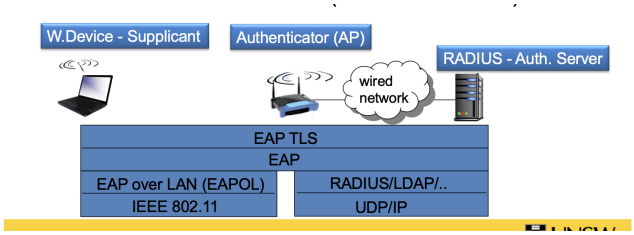
An user must authenticate with the switch or AP before connecting.

Authenticator:
For EAP, the authenticator acts as a relay between supplicant and authen server.

- Configured with address of authentication server:
  - Possible co-location of authenticator server.
  - Shared secret with authenticator server.

The user uses EAP protocol to connect with the authen server, until the identity is verified, they can't use higher layer protocols.

Wireless device to AP
AP to authen server (radius over udp)



Attack on data link layer:

1. Mac spoofing
   Mac address: no 2 devices in the world have the same mac addresses. 802.1X authentication prevents.
2. ARP spoofing (ARP: to map a network address, such as an IP address, to a physical address, such as a MAC address)
   The attackers send fake ARP messages binding the target's IP address to its own MAC address.

3. VLAN hopping -

Double tagging: the attacker sends its frame to more than 1 VLAN by inserting 2 VLAN tags in a frame it transmits
Switch spoofing: an attacking host impersonates a trunking switch responding to the tagging and trunking protocols

4. Switch poisoning attack

802.1x-based authentication can help prevent MAC (Media Access Control) spoofing, which is a type of network attack where an attacker changes the MAC address of their network interface card to impersonate another device on the network.

In an 802.1x-based authentication system, before a device is granted access to the network, it must first authenticate itself to a central authentication server. The authentication process involves the exchange of credentials between the device and the authentication server, and only devices with valid credentials are granted access to the network.

Since the authentication process requires valid credentials that are unique to each device, it is much more difficult for an attacker to impersonate a legitimate device by spoofing its MAC address. This is because the attacker would also need to obtain the valid credentials associated with the device they are trying to impersonate.

Therefore, 802.1x-based authentication can be an effective measure to prevent MAC spoofing and improve overall network security. However, it's important to note that 802.1x-based authentication is not a foolproof solution and should be used in combination with other security measures, such as MAC address filtering, port security, and network access control (NAC), to provide comprehensive network security.

Week 7:

Firewalls and IDS
Firewalls isolate organization's internal net from larger Internet, allowing some packets to pass and blocking others

3 ways of firewalls

1. Stateless packet filters

- Internal network connected to the Internet via *router firewall*
- Firewall *filters packet-by-packet*, decision to forward/drop packet based on:
  - source IP address, destination IP address
  - TCP/UDP source and destination port numbers
  - ICMP message type
  - TCP flag bits (SYN, ACK, FIN)

Decide whether to drop or allow packets coming from internet
ACL: table of rules, processing a packet from top to bottom

2. **Stateful packet filters**
   No existing connection in table: drop the packets

3. Application gateways - allow users instead of IPs.
   Firewalls only read packet headers.
   An application specific filter allows selected internal users to telnet outside

ACL:

If a packet matches an entry in the ACL, the matching process stops and the firewall takes the action specified in the matched rule.

IDS:

Drawbacks of packet filtering:

- Only for TCP/IP headers
- No correlation check among sessions

- IDS: intrusion detection system
  - deep packet inspection: look at packet contents (e.g., check character strings in packet against database of known virus, attack strings)
  - examine correlation among multiple packets
    - port scanning
    - network mapping
    - DoS attack

IDS method:

Signature based:
Requires prior knowledge of potential attacks and only works if the attack signature is in the database.
Uses the pre-defined rules/ patterns in DB.

Anomaly based: define a profile describing normal behaviors, then detect deviations.
Consider normal behaviors of legal users over a period of time, and apply statistical tests to detect intruders.
An attack scenario needs not to be defined a priori.

IDS deployment:

Host-based IDS:

Network-based IDS:

- NIDS blocks malicious traffic that it detects? - false

Dns hijacking:

Dns spoofing (dns cache poisoning):
The attacker intercepts and changes the dns response message to redirect users to a malicious website or IP address.

- Use digital signatures to verify the authenticity of DNS records. (dns server sends the response + K-(hash(response)). On receiving, decrypt with dns server's K+, calculate the hash and verify)

DNS exfiltration:
Sending out sensitive data of an organization through DNS requests.

DNS amplification attack: overloading a DNS resolver with a flood of requests

Example: DNS DDoS:
An attacker uses a great amount of compromised devices, sending high volumes of DNS queries to a DNS server and overwhelming its capacity.

Week 8:

ECC scheme

- Key agreement (ECDH)
  Allows for establishment of shared secrets similar to DH.
  The shared key is then used for symmetric encryption

- Digital signature (ECDSA)
  Allows uses of public/ private key for signing a message and verification of signature, more efficient than RSA based on DSA

ECDH alg
y^2 = x^3 + ax + b

E: y^2 = x^3 + x + 6
Alice and Bob choose public base point B on E: (2, 4)
Alice chooses a random integer 1<\alpha, computes P=\alpha * B, Alice sends P to Bob
Let \alpha = 4,

Same for Bob:
Let \beta = 5

Challenges for broadcast security

Asymmetric key
Sign each digital packet and verify using asymmetric key

Trivial broadcast key distribution

1. Encrypt group key K using the targets' public key individually
2. Only the targets can extract group key using their private keys

16

Trivial broadcast key distribution

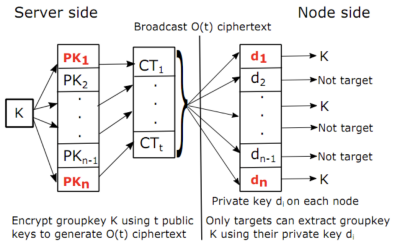


Figure 1: Trivial Broadcast Encryption scheme in an n nodes network targeting t nodes. K: shared group key. PK: Public Key. CT: Ciphertext. d: Decryption using each node's private key.

-

Messages are encrypted with only targets' public keys.
Not target: receives, can't use their K-s to decrypt.
Target: can decrypt with their K-s.

Hash chain basics

19

Hash Chain -basics

- Client generate 1000 hashes for password
  - Suggested by Lamport for password protection
- Server stores H^{1000}(password)
- Client willing to authenticate sends H^{999}(password)
- Server computes H^{1000}(password) = H(H^{999}(password))
  - Match found, store H^{999}(password)for next time
- Eavesdropper can't use H^{999}(password) since server expects H^{998}(password)

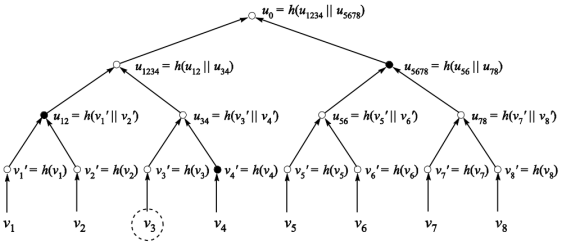
UNSW

(1) The server stores H^1000(pw) (S)
(2) Client sends H^(999) to the server and server matches H(H^999(password)) with (S)
(3) The server stores the received H^(999) in S, and the next time the client needs to send H^(998)

Merkle hash tree:

33

Merkle Tree Another Example



- Authenticate v_3, v_3 is revealed together with v_4', u_{12}, u_{5678})
- verifiers hash the revealed values appropriately and check if the result is u_0
  - u_0 = H(u_{12} || H(H(v_3) || v_4')) || u_{5678})

UNSW

u34 = hash(concat (h(v3), h(v4)))

Week 9:

DER: distributed energy resource

Resource significance level

Network segmentation

Resources with different criticality levels must be located in different security zones.

A zone: 1 or more subnets or broadcast domains where a device in the zone can communicate with other devices within the zone freely.

Communications between 2 different security zones must be routed through the security gateways.
A security gateway: a firewall, however, it can also take the form of a router, layer 3 switch, cellular modem, radio and so on.

Firewall1: controls access between the managing system in the high impact zone and headend in the medium impact zone and another high impact zone.

2.8 Communications between a system/resource in the high-impact zone and a system/resource in the low impact zone must be routed through a DMZ.
2.9 Communications to/from an external network must be routed through a DMZ.

Boundary protection

Communications should always be initiated in a higher criticality zone.
Responses to traffic originating in the higher security zone are typically automatically allowed by so-called stateful firewalls. The firewalls maintain a table of connections that were initiated in higher security zones and allow the traffic in the opposite direction.

Stateful firewalls

Communication partitioning

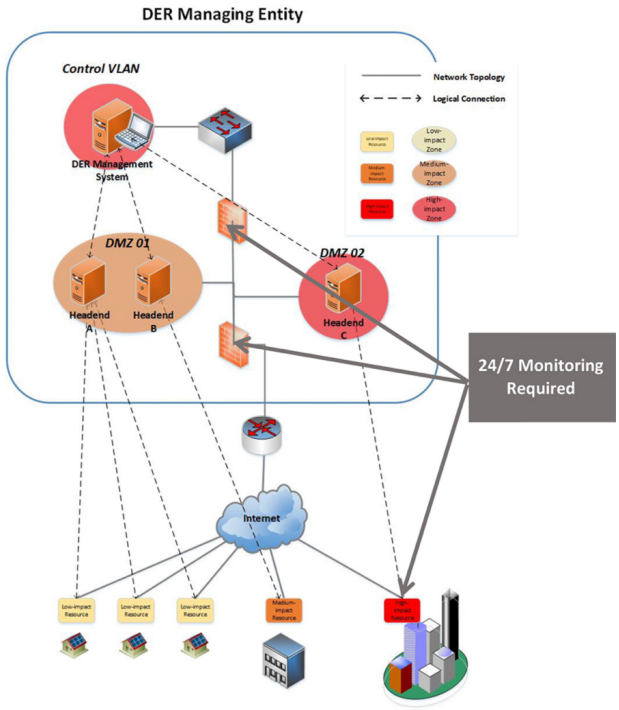


Figure 4 – Boundary protection

Risk-based network service protection

The network infra should be configured to allow only authorized resources to join/connect to the network.
At minimum:
Wired network: disabling unused ethernet ports and using media access control (MAC) address-based port locking.
On an 802.11 wireless network: should at least include wpa2 password and MAC address filtering.

To avoid attackers redirecting traffic to compromised or malicious hosts, the systems providing name/ address resolution to high-impact-level resources must have a technical mechanism to prevent forging or manipulating of DNS data.

DNSSEC:
The dns servers should be configured to use DNSSECs to cryptographically sign records and prevent DNS spoofing or cache poisoning.

Risk based data integrity protection

Data integrity includes 3 key components:

- A mechanism to authenticate each participant in the conversation
- A mechanism to verify the integrity of communications
- A mechanism to detect illegitimate or unauthorized alterations of the communications from/ to high-impact resources.

Hash or checksum is required for the second and third requirement.

All DER communications must be protected with a mechanism to verify integrity.
DER communication to/from high-impact resources must be protected with a mechanism to detect illegitimate alteration of messages between resources connected to the network.

- h(m) encrypted with some private key, and the receiver decrypted with some public key

Risk based data confidentiality protection

Use protocols such as TLS/ SSH/ IPSec, which are for e2e encryption.
The implementation should use PKI. Both end-points have their own key-pairs and associated certificates to make revocation simpler.

Week 10:

ZTA
Access rules are made as granular as possible to enforce least privileges needed to perform the action in the request.

In the abstract model of access shown in Figure 1, a subject needs access to an enterprise resource. Access is granted through a policy decision point (PDP) and corresponding policy enforcement point (PEP).[3]

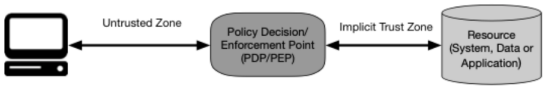


Figure 1: Zero Trust Access

- All data source and computing services are resources
- All communication is secured regardless of network location
- Access to individual enterprise resources is granted on a per-session basis
- Access to resources is determined dynamically. (observable state of client identity, application/ service, the requesting asset…)
-

Logical components

PDP [control plane] -> policy engine, policy admin
PE: (ultimate decision) it has the PE's trust algorithm. It makes decisions, uses the enterprise policy and inputs from external -> input to a trust algorithm.
PA: It executes PE's decision. If a session is authorized, the PA configures the PEP to allow the session to start.

PEP [data plane]: It communicates with the PA to forward requests and/ or receive policy updates from the PA.

Additionally, several data sources provide inputs and policy rules used by the PE
PKI: generating and logging certificates generated by the enterprise. X509??
ID management system:

- **Variations of ZTA approaches**

(ZTA uses micro-seg): the enterprise places infra devices or special purpose gateway devices to act as PEPs protecting small groups of resources

(ZTA using network infra and software defined perimeters):

- **Deployed variations of the architecture**

Device agent/ gateway deployment

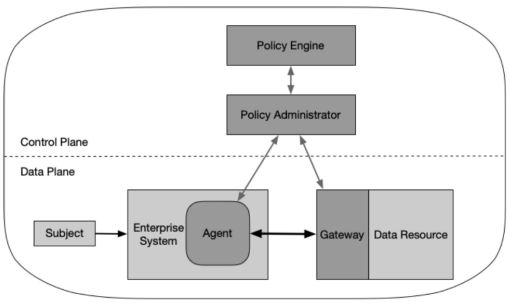


Figure 3: Device Agent/Gateway Model

(1) A subject with an enterprise-issued laptop wants to connect to an enterprise resource.
(2) The request is taken by the local agent and it's forwarded to the policy admin.
(3) Policy admin -> (request) policy engine (evaluation)

If the request is authorized, the PA configures a comm channel between the agent and the relevant resource gateway via the control plane.

Encrypted service data/apps flows begin.

Enclave-based deployment

Resource portal-based deployment

Device application sandboxing

The model may allow attackers to discover and attempt to access the portal or attempt a DoS attack against the portal.

- **Trust algorithm**

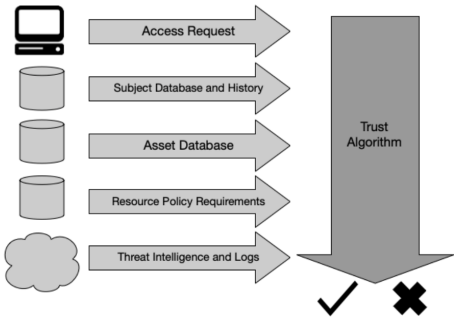


Figure 7: Trust Algorithm Input

- **Network/ Environment components**

There should be a separation of:
(a) communication flows used to control and configure the network
(b) the communication flows of app/service data that do the actual work

1. The enterprise must be able to distinguish between what assets are owned or managed by it, and the devices' current security posture.
2. The enterprise can observe all network traffic - NIDS??
3. Enterprise shouldn't be reachable without accessing a PEP (not accept arbitrary incoming connections from the internet, prevent DoS attacks against resources located behind PEPs, note that not all resources should be hidden in this way - some network infra components, such as DNS servers must be accessible)
4. Remote enterprise assets should be able to access enterprise resources without needing to traverse enterprise network infra first.