MidTerm Notes:

* Threats are what an attacker wants. If there are no threats, there is no security problems.
* Vulnerabilities are weaknesses in the system that allow an attacker to carry out a threat.
* Risk = Assets \* Threats \* Vulnerabilities  
  Security policies define what is secure and what is not secure.
* Security mechanisms implement security policies.
* Assurance is convincing yourself that the security mechanisms accurately implement the   security policy and are trustworthy (i.e., bug free).

8 Design Principle:

1. Principle of Least Privilege (disallow the birth of an enemy)  
       Example: Faculty don't get administrator privileges
2. Principle of Fail-Safe Defaults (spontaneous right action)  
       Example: Default settings maximize security. To open something up, user has to do it.
3. Principle of Economy of Mechanism (do less and accomplish more)  
       Example: Simpler programs have fewer bugs that attackers can exploit  
       (However, an OS that didn't check passwords would be simpler but less secure and  
        the JVM which checks array bounds is more complex but more secure)
4. Principle of Complete Mediation (infinity at a point)  
       Example: Assume nothing, always check security
5. Principle of Open Design (all this is That)  
       Example: Don't assume that attacker can't reverse engineer a program.
6. Principle of Separation of Privilege (sequential unfoldment)  
       Example: Packet has to pass through two firewalls to get to internal network
7. Principle of Least Common Mechanism (cultural integrity)  
       Example: Reduce sharing; keep important stuff off the network if possible.
8. Principle of Psychological Acceptability (the nature of life is the expansion of happiness)  
       Example: Don't make a security feature too hard (or too annoying) or else user's will   
       avoid it.

**CIA Triads:**

Computer security is based on confidentiality, integrity and availability

**Confidentiality**

Confidentiality is the concealment of information or resources.  
Achieved via access-control mechanisms (file permissions) and cryptography

Example: sending credit card number using SSL  
Example: saving exam in a directory which students do not have permission to access.  
Generally, there are three instances in which information is vulnerable to disclosure:

1. when the information is stored on a computer system.
2. when the information is in transit to another system (on the network)
3. when the information is stored on backup tapes.

Other examples  
Example: don't let hackers know what server or OS you are using.  
Example: conceal the mere existence of data  
  
Your confidence in the confidentiality mechanisms is ultimately based on trust in the OS

that supports them.

**Integrity**

Integrity refers to the trustworthiness of data or resources, and is usually phrased in terms of preventing improper or unauthorized change.  
data integrity - content of the information  
source integrity - source of the information  
Two types of integrity mechanisms: prevention mechanisms and detection mechanisms

1. Prevention mechanisms attempt to block
   1. unauthorized changes to data (hacker changing data)
   2. changes to data in unauthorized ways (accountant embezzling data)
2. Detection mechanisms do not try to prevent violations of integrity, they report when data has been modified.

Example: detect that Internet order was modified between the client and the server  
 Example: detect that a system file has been modified by a virus.  
Confidentiality trusts in OS, but integrity also relies on assumptions about the source of the data and about trust in the source.

**Availability**

An attacker might swamp a server with requests, thereby making it unavailable to legitimate users. Called a denial of service attack (DoS)  
How to know if increase in activity is malicious?  
Example: If CNN mentions a web site, its activity will increase, perhaps to a point where it crashes. Whether deliberate or not, it is still a security problem.

**Threats**

A threat is a *potential* violation of security. An action that could cause a threat to be realized is called an **attack**. The perpetrator of an attack is called an attacker.

Example:If I have a beat-up old bicycle that nobody wants there is no threat that somebody will steal it. However, there is a threat that somebody will steal a laptop computer.

Confidentiality, integrity and availability counter threats to the security of a system

Four classes of threats:

* disclosure - unauthorized access to information (confidentiality)
* deception - acceptance of false data (integrity)
* disruption - interruption or prevention of correct operation (availability)
* usurpation - unauthorized control of some part of a system (integrity, availability)

**Types of Access Control:**

* Discretionary access control (DAC): – an individual user can set an access control mechanism to allow or deny access to an object.
* Mandatory access control (MAC): – a system mechanism controls access to an object and an individual user cannot alter that access.
* Originator controlled access control (ORCON): – access based on the creator of an object (or on information it contains)

**Bell- LaPadula Model**

Design to protect classified information

This supports data confidentiality only, It’s assume that data is never going to disclose with unauthorized recipient. It’s built on state machine concept i.e. we can map one state to another state such as we have unaccessed state object and the next state may be accessed object and based on the transition rule we know we are in the secure state on the both end of the spectrum so our system is always secure.

Supports multiple state and the transitions between state. We have to dictate or document every state of object and subject that can take and map the transition between the two and as you implement the control you simply need to control the transition from one state to another.

* Simple Security Property (no read up)
  + Subject may not read  an object at a higher sensitivity level. Eg: if I have clearance of secret I cannot read a document that is classified as Top secret.
* (star)\* security property (no write down)
  + Subject may not write an object at a lower sensitivity level. Eg: We cannot write document in lower clearance level because what happens if you have secret document open and  you copy a paragraph and paste in other document containing lower classification of data. You have just extracted the secret passage of data  and disclosed in lower secure data. This violate the confidentiality model.
* Discretionary security property
  + Access matrix enforces discretionary access control. There is access matrix to decide who can access what document at which particular time.
* For a subject not to have r/w access to an object, following 2 conditions should be in effect.
  + Clearance of S < Classification of O AND
  + Category of O is not a subset of Category of S  ??

**Paul, cleared for (TOP SECRET, {A, C}), wants to access a document classified (SECRET, {B, C}). What type of access does he have?**

**a) read**

**b) write**

**c) both**

**d) neither**

**ANS: d**

**(11) Sammi, cleared for (TOP SECRET, {A, C}), wants to access a document classified (CONFIDENTIAL, {A}). What types of access does he have?**

**a) read**

**b) write**

**c) both**

**d) neither**

**ANS: a**

**(12) Robin, who has no clearances (and so works at the UNCLASSIFIED level), wants to access a document classified (CONFIDENTIAL, {B}). What types of access does she have?**

**a) read**

**b) write**

**c) both**

**d) neither**

**ANS: b**

**CONFIDENTIAL is greater than UNCLASSIFIED and the empty set (the categories of Robin) is a subset of {B} so the object dominates the subject and hence Robin can write.**

**(13) Anna, cleared for (CONFIDENTIAL, {C}), wants to access a document classified (CONFIDENTIAL, {B}). What types of access does she have:**

**a) read**

**b) write**

**c) both**

**d) neither**

**ANS: d**

**(14) Jesse, cleared for (SECRET, {C}), wants to access a document classified (CONFIDENTIAL, {C}). What types of access does he have?**

**a) read**

**b) write**

**c) both**

**d) neither**

**ANS: a**

**Remember, no writes down.**

Important Topics from Webinar Discussion

CIA Triads

Threat, Attack, Attacker, Vulnerability, Asset

Four Classes of Threats

Eight Design Prinfciple

Access Control Matrix

Types of Access Control: DAC,MAC

Security Policy vs Mechanism

Two different type of Security Policy

Dealing with confidentiality

Dealing with integrity

Hashing:

* Simple form of hash is a number
* MD5 128 bit hash (32  hexadecimal characters)
* SHA1 160 bit hash (40 hexadecimal characters)
* It enforces the data integrity.
* To ensure data hasn’t been modified or tampered hashes or checksums often used to validate data. If the message is modified in transit but the hash doesn’t change then the message is modified and has lost its integrity. If message can be modified, it is possible that hash also can be modified along with message. So, hash often encrypted to ensure it isn’t modified.

Encryption:

* It enforces confidentiality
* Prevents unauthorized disclosure

Digital Signature:

* Message is hashed( providing integrity)
* Hash encrypted with sender’s private key
* Public key can decrypt (providing authentication and  non-repudiation)

Classic cryptosystem:

Public key cryptography:

https://docs.google.com/drawings/d/sARbBrwcePdXacz7YSWxcuA/image?w=434&h=53&rev=69&ac=1

Relationship between plain text and chiper text

Strength vs weakness security policy

Online attack and offline attack

Digital Certificates:

* digital certificate provides identifying information,
* It is forgery resistant and can be verified because it was issued by an official, trusted agency.
* The certificate contains
  + the name of the certificate holder,
  + a serial number,
  + expiration dates,
  + a copy of the certificate holder's public key (used for encrypting messages and [digital signatures](http://searchsecurity.techtarget.com/definition/digital-signature))
  + The digital signature of the certificate-issuing authority ([CA](http://searchsecurity.techtarget.com/definition/certificate-authority)) so that a recipient can verify that the certificate is real.

To provide evidence that a certificate is genuine and valid, it is digitally signed by a root certificate belonging to a trusted certificate authority. Operating systems and browsers maintain lists of trusted CA root certificates so they can easily verify certificates that the CAs have issued and signed. When PKI is deployed internally, digital certificates can be self-signed.

PKI: Public Key Infrastructure

A public key infrastructure (PKI) supports the distribution and identification of public encryption keys, enabling users and computers to both securely exchange data over [networks](http://searchnetworking.techtarget.com/definition/network) such as the Internet and verify the identity of the other party.

Primary elements of PKI

* CA
* Registration Authority: subordinate CA
* Certificate Database: store certificate request, issues and revokes certificate
* Certificate Store:resides on local computer and store issues certificate and private keys

Certificate Authority:  is a trusted entity that issues Digital Certificate and acts as the root of trust and provides services that authenticate the identity of individuals, computers and other entities

RSA:one way function

-Easy to generate cipher function but hard to generate reverse decipher function.

-Used trapdoor to generate decipher function.

• Public key: – n = p∗ q where p and q are prime

- e prime with (p-1)(q-1)

• Private key: – d so that e∗ d mod (p-1)(q-1) = 1

• Cipher: c = m^e mod n

• Decipher: m = c^d mod n

Backdoor:

Method of bypassing encryption of normal authentication, securing unauthorized remote access to computer, or obtaining access to plain text in cryptographic system

It may be in the form of hidden part of program or may be hardware feature so that if anything worse happens still we have backdoor to restore back to normal situation.

Eg: default passwords can function as backdoor if they are not changed by the user.

Clark-Wilson:

1. Uses well-formed transactions as the basic operation.  
D + YB - W = TB

D is deposits made today  
W is withdrawals made today  
YB is yesterday's balance  
TB is today's balance

**Task: Add a new TP and its associated CDIs to the certified relation**  
(translation: add a new program and the tables it may access to a configuration file)  
To achieve this, the following certification rules must be verified:  
1. CR1: Verify that any IVPs that validate action of new TP work properly  
2. CR2: Verify that TP transform CDIs from a valid state to a valid state.  
3. CR4: Verify that the TP appends enough information to reconstruct the operation to  
a log file (called an append-only CDI)  
4. CR5: If the TP takes a UDI, make sure it transforms it into a CDI before using it.  
**Task: Add a user to the allowed relation**  
(translation: give a user permission to run a program)  
To achieve this, the following certification rule must be verified:  
CR3: Verify that the user is not already associated with a TP that  
violates the principle of separation of duty. If not, the add user and TP to the  
allowed relation.  
**Task: A user wants to run a TP on a CDI**  
To achieve this, the following enforcement rules must be carried out  
ER1: Verify that TP is associated with the CDI in the certifies relation  
ER2: Verify that the user who is running the TP is in the allowed relation.  
ER3: Authenticate the user before letting them run the TP (principle of separation of privilege).  
Finally, let user run TP on the CDI and log the transaction  
**Task: Change list of CDIs associated with a TP**  
To achieve this, the following enforcement rule must be carried out:  
ER4: Only person responsible for the certified relation can do this  
and that person must not be a user of the TP.

Kerbores

PGP flow and diagram

Replay Attack by Trudy

1. Alice --> Cathy: { request for session key to Bob }kAlice  
*When Alice subscribed to Cathy's service she was given kAlice. Only Alice and Cathy know this key*  
2. Cathy --> Alice: { ksession}kAlice || {ksession}kBob  
*Cathy generates a session key, ksession that Alice can use in her communication with Bob. Cathy sends back ksession encrypted with Alice's key (which she can decipher) and also sends back ksession encrypted with Bob's key. Since Alice doesn't know kBob she can't decipher it but that doesn't matter, she is just going to send it to Bob.*  
3. Alice-->Bob: { ksession }kBob  
*When Bob receives this he can can decipher it using kBob which is a key that he shares with Cathy. Like Alice he too has subscribed to Cathy's services.*  
4. Alice-->Bob: { Transfer $500 dollars to Dan }ksession  
*Alice then can use ksession to communicate securely with Bob.*Unfortunately, here is what an attacker, Trudy,  can do.  
1. Intercept and record messages 3 and 4 above.   
2. Trudy resends { ksession }kBob  
*Bob has no idea that Trudy is sending this.*  
3. Trudy resends { Transfer $500 dollars to Dan }ksession  
*Again Bob has no idea that Trudy is sending this. Furthermore, Trudy can replay this message several more times making Alice much poorer.*

Needham- Schroeder Protocol

A defense against replay using two random numbers.  
1. Alice-->Cathy: { Alice || Bob || rand1 }  
*Alice tells Cathy that she wants to talk to Bob and needs a session key.*   
2. Cathy-->Alice: { Alice || Bob || rand1 || ksession || {Alice || ksession}kBob}kAlice  
*Cathy sends back an identifier of the session (Alice || Bob), rand1 (which convinces her   
she is talking to Cathy), the session key, and message encrypted with Bob's secret key (which   
he shares with Cathy). This message identifies Alice and has the session key.*  
3. Alice-->Bob: { Alice || ksession }kBob  
*This is the same as the previous protocol, so Trudy may record this*  
4. Bob-->Alice: { rand2 }ksession

*Bob is now checking that it is indeed Alice*  
5. Alice-->Bob: { rand2 - 1 }ksession  
*It won't do any good for Trudy to replay this because next time Bob will send a different  
rand2.*  
6. Alice-->Bob: { 1. message }ksession  
  Alice-->Bob: { 2. message }ksession  
  Alice-->Bob: { 3. message }ksession  
*The numbers 1, 2, 3 are segment numbers and prevent Trudy from replaying a message.*

If session keys are generated with a non-secure random function (one in which it is possible  
to detect patterns) then Trudy could defeat this protocol as follows.  
1. Subscribe to Cathy  
2. Get several ksession keys from Cathy and figure out the algorithm used to generate them so that given ksessionn it is possible to predict what  ksessionn+1 is.  
3. Get a session key from Cathy  
4. Eavesdrop on Cathy to see who the next person is who requests a session key. Assume that it is Alice. Trudy will know what the session key assigned to Alice is. Since Trudy knows the session key she will be able to decipher Bob's random number and respond with the correct enciphered response.  
5. Trudy can now replay step 3 of the Needham-Schroeder protocol.  
 Trudy-->Bob: { Alice || ksession }kBob  
*Note: Trudy does not know what kBob is but she does know what ksession is. Because she doesn't know what kBob is she is forced to do this replay*  
6. Bob-->Alice: { rand3 }ksession  
*Trudy intercepts this. Since she knows ksession she will be able to get rand3*  
7. Trudy-->Bob: { rand3 - 1 }ksession  
*Since Trudy knows ksession she can send back the response that Bob is expecting.*   
8. At this point Bob thinks that Trudy is Alice and since Trudy has ksession  
she can send anything she wants to Bob and Bob will think that Alice is sending it. Furthermore,  
she can decipher anything that Alice sends during that session.

Denning and  Sacco Protocol

Denning and Sacco suggested using timestamps to make the Needham-Schroeder protocol more robust in case the session key is cracked. Their solution enables Bob to detect the replay of step 3(from Nadheem protocol).  
1. Alice-->Cathy: { Alice || Bob || rand1 }  
*Alice tells Cathy that she wants to talk to Bob and needs a session key.*   
2. Cathy-->Alice: { Alice || Bob || rand1 || ksession || {Alice || T || ksession}kBob}kAlice                                                                                                   
*Cathy sends back an identifier of the session (Alice || Bob), rand1 (which convinces her she is talking to Cathy), the session key, and message encrypted with Bob's secret key (which he shares with Cathy). This message identifies Alice and has the session key. What is new here is that a timestamp T is also passed back*  
3. Alice-->Bob: { Alice || T || ksession }kBob                                                
*Except for the timestamp T, this is the same as the previous protocol, but if Trudy records this and she plays it back later, Bob will detect that the timestamp T is too old and know that a replay is being attempted.*  
4. Bob-->Alice: { rand2 }ksession  
*Bob is now checking that it is indeed Alice*  
5. Alice-->Bob: { rand2 - 1 }ksession  
*It won't do any good for Trudy to replay this because next time Bob will send a different  
rand2.*  
  
The problem with Denning-Sacco is that it requires that the clocks on the sender and receiver  
machines are synchronized. This is, in general, difficult to do over the network.

SSL

The secure socket layer provides secure communication using public key and conventional cryptography. It runs in user space on top of the transport layer (TCP/IP).

What is Authentication? Why it is important??

Good Password Vs Bad Password

DoS vs DDoS