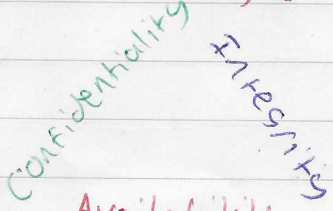
Fundamentals - CS140

2 General Features of Computer Security

1. No such thing as absolute Computer Security
   If we want to know whether a system is secure there are many questions we can ask.
   - Security of which aspects (Secrecy, damage prevention, DoS)?
   - Security from whom?
   - Security to what level?

2. In Security, theory is not the same as practice.
   Security is not just about what theory you use, it is also about.
   - implementation
   - deployment
   - maintenance
   - parties involved
   - location
   - temptation

CIA triangle



Worm and Virus

- Both Self-replicate
  - Virus: replicate into other executable code.
  - Worm: Standalone

Analyzing Security Incidents

- Who is doing it?
- motivation
- What is the damage?
- What should "system security" encompass

Terminology

- Asset: anything we value enough so we want to protect it.
- Vulnerability: a weakness in a system that could be exploited.
- Threat: potential for violation of security, when an attacker has capability and intention.
- Risk: expectation of loss.
  risk = prob x harm
  = threat x vulnerability x harm.
- Attack: assault on security that derives from a threat

Risk Analysis methods:

DREAD - Dmg, Reproducability, Exploitability, Affected Users, Disc...
Quantitave Risk Analysis
Fault Tree Analysis

Countermeasure: Action to reduce a threat/Vuln/harm or by preventing/detecting an attack.

Trust: no system is devoid of trust.

Authentication - CS140

Tendencs in passwords

- 12% of employees used "password" as their password
- A study found 1/4 of passwords are crackable using a dichionary and combinations of the username.
- ≈12% use dichionary words
- ≈81% use alphanumeric passwords.

Problems with storing passwords

- Many people use simple passwords
- Some sites store passwords in plaintext
- Some sites allow for unlimited losin attempts
- Passwords may not be salted

Password Cracking

Combinations: password length
Character set size

using brute force the number of combination checks will be half of this.

User overload

- Users should use different passwords for different sites.
- But it is hard to remember all of these passwords.

Solutions:

- Single protected place - e.g. dashlane
- Dual Factor authentication

Calculating crack times

Assume 8 long password and 94 character set - 6.1×10^{15} poss

- Good PC:
  6.1×10^{8} secs ≈ 19 years
- PC + GPU:
  6.1×10^{5} secs ≈ 7 days
- 25 GPU Cluster
  ≈ 4.8 hours.

Password Strength

Entropy value - How many bits the search space is equivalent to.

e.g. $676 = 2^{x}$ gives $x \approx 9.4$

W = Character set size

L = password length

$$x = L \times \log_2(W)$$

Human generated passwords

- humans do not choose random passwords.
- much less uncertainty in human generated passwords

password hashing:

$x \rightarrow f(x)$ easy $f(x) \rightarrow x$ hard

When logging in, password is hashed then checked against the hashed password in the data base.

Password Cracking - CS140

Online Vs Offline attack

Online:

- fire successive login attempts at a site
- Many sites prevent such behaviour

Offline

- Comprised a computer system and obtained a password file.
- Then try to guess the passwords corresponding to the hashes in the hash file

Cracking methods

Brute force

Dictionary attacks

Look-up tables

Reverse Look-up tables

Rainbow tables

Dictionary attacks

Uses a file containing common words/phrases for passwords

Look-up tables

Store well known passwords and their hashes in a file to lookup the hash for the password

Rainbow table

problem with Reverselookup table

$P_1 \to h_1 \to P_2 \to h_2 \to P_3 \to h_3 \to P_4$

$P'_1 \to h'_1 \to P'_2 \to h'_2 \to P_2 \to h_2 \to P_3$

wastes space ← will repeat.

FIX

For each step in the hash Chain we use a new Reduction Function.

Cracking passwords:

Assume k steps.

Attempt hashing from $R_k \to R_1 \to R_2 \ldots$

if no match hash the following $R_{k-1} \to R_k \to R_1 \to R_2 \ldots$ when we have checked all rotations Can assume password isn't in table

Reverse Look-up tables

- Sacrifices time for storage.
- Uses a technique called hash Chain
- Only store the first and last password in each Chain.

Cracking passwords

- Create a hash Chain and if we observe a "last" password, recreate that hash Chain

Why may h not be found?

defense

No defense against brute force or dictionary.

Using salts can prevent look up tables etc.

Salts

random ly generated number. appended to password before hashing.

Password Cracking 2 - CS 1470

Why Salts help?

They make lookup tables and rainbow tables too space inefficient.

This is because you need to make a chain for each possible salt

Password hash & Linux

When a user is created their info is stored in the passwd file and their hash is stored in the shadow file.

Any user can read passwd

Only root can read shadow

Biometrics

Problems:

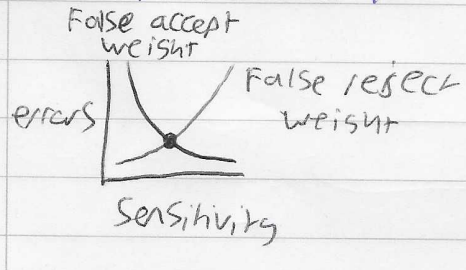
Does not have a clear cut yes or no

False negative:

"Authorized person denied"

False positive:

"baddie allowed in"



Cryptography - CS140

Terminology

Plaintext - Original message

Ciphertext - result of encrypting message

Encryption/Decryption - process of transforming plaintext to ciphertext (or reverse)

More on secret key encryption

"Standard Algorithms":

- Data Encryption Standard (DES)
- Advanced Encryption Standard (AES)

key is shared between sender and receiver.

Decryption is reverse of encryption.

LSB insertion.

replace the LSB of each pixel in an image with the data ~~data to~~ we want to hide.

Transposition or permutation cipher

Re-arrange plaintext in rows of fixed length, then send msg in columns

HEREISMYMESSAGE

HERE
ISMY
MESS
AGE

Confusion

HIMAESEGRMSEEYS

key is knowing what permutation is used

Secret key encryption

key is secret

Same key used for encryption and decryption.

Public key encryption

encrypt and decrypt message with different keys.

Public key - public

Private key - private

Steganography

Hides message in plain sight

Often unreadable

process:

Cover - file to hide data

data - data to hide

key - encryption key

Cover + data + key = Stegano medium

↑ we send this

Code words

Define code words ~~and~~ for normal vocab.

Use a codebook to translate ciphertext.

Monoalphabetic Substitution Cipher

e.g. Caeser cipher.

Another example: define keyword and put at start of new alphabet

ABCDEFGHIJKLMNOPQRSTUVWXYZ
ZEBRASCDFGHIJKLMNOPQTUVWXY

easily cracked by frequency analysis.

Confusion

Cryptography 2 - CS140

Frequency Analysis
Count the letter frequency in ciphertext and compare against letter frequency of plaintext alphabet.

XOR
Add operation is too expensive. And + OR does not give unique encryption.
So we use XOR instead. We use it to combine message and key

Confusion
- Changing a bit of the key changes multiple parts of the ciphertext
- Hides relationship between ciphertext and key

Principles of good encryption
- Confusion
- Diffusion
- hard to break even with several assumptions
  - know encryption process
  - know initial settings
- Management of the encryption scheme must be feasible and cost-effective.

Block Cipher
encrypts blocks of data one at a time.
e.g. DES

Polyalphabetic Substitution Cipher
To avoid frequency analysis. uses multiple ciphertext alphabets.
uses a key to decide which ciphertext alphabet to use.
e.g. Vignere Cipher

One-time pad.
Each letter in the message in encrypted using a different alphabet.
Not a practical solution:
- key generation
- key distribution
- key protection

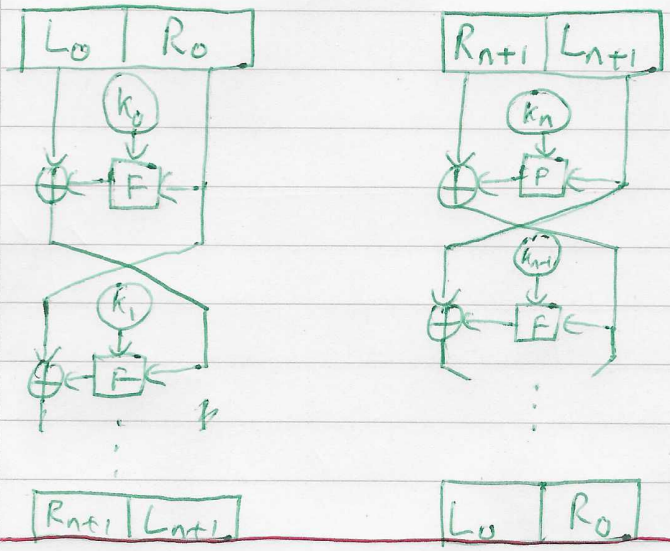
Diffusion
Change a bit of plaintext, half bits of Ciphertext should change.
Hides relationship between plaintext and Ciphertext

Stream Cipher
encrypts each input element one at a time.

DES
Based on Feistal approach
- Block Cipher: 64-bit blocks
- 56 bit secret key
- 16 rounds of encryption operations
- uses substitution (Confusion)
- uses permutation (diffusion)
- Encryption operations are public

Cryptography 3 - CS 180

Feistal approach



Split plaintext into 2 equal pieces (L_0, R_0).

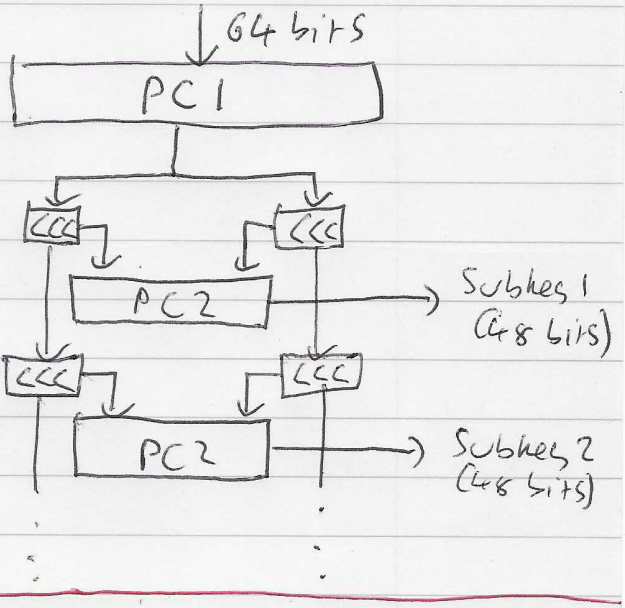
Go through 16 rounds of encryption

Output of each round will be used as the input of the next round.

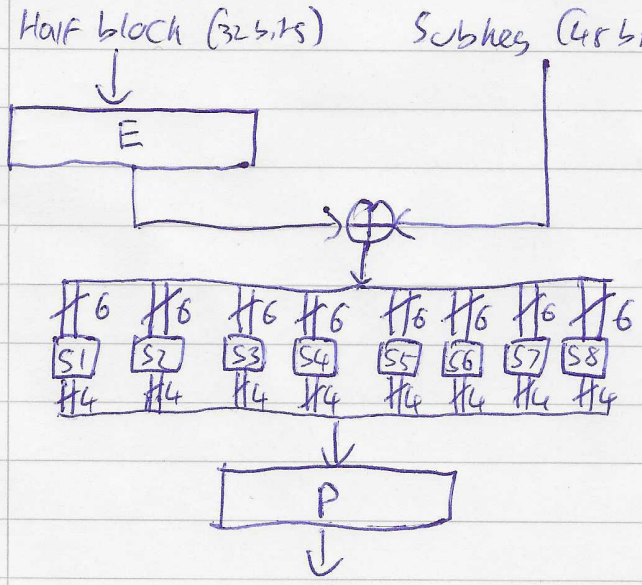
$L_{i+1} = R_i$, $R_{i+1} = L_i \oplus F(R_i, k_i)$

Generation of Subkeys

- perform permuted choice 1:
  - shuffle and select 56 bits from 64 bits
- In each round:
  - key is divided in 2
  - each half is shifted left by 1 or 2 bits
  - perform permuted choice 2
    - shuffle and select 48 bits from 56 bits.



Feistal Function



E: duplicates half the bits to make half block 48 bits

S: use vectorial boolean function to convert 6-bit to 4-bit.

P: permutation each S-box output spread across 4 different S-boxes helps diffusion

Cryptography 4 - CS140

AES

DES is now considered insecure due to the short key.
AES is the new standard
128 bit block size
longer key 128-256 bits
Faster implementation.
Similar block cipher features but uses a SPN-designed for more inherent parallelism.

IS AES secure

Assuming a Super Computer Could Crack DES in 8 seconds
It would take 1.3×10^{15} years to crack AES.
AES-256 can give post-quantum assurance

public key cryptography

A way to communicate that doesn't require a shared key.
Given input to the function it is easy to find the output. (Doesn't work the other way around).

One way function

Easy to compute one way but not the other.
$k = y^x \bmod p$ is one-way
y is a primitive root of p
p is an enormous 512-bit number

AES processing

10 rounds of encryption
Round:
Substitute on 8-bit block using a look-up table.

Shift each row left by its row index.

Multiply each column by a fixed matrix to get columns for a new matrix.

Perform XOR with round key.

Problem with secret key:

The key must be shared.
How do we do this?
- Physical distribution
- key distribution Centre

This is not very efficient

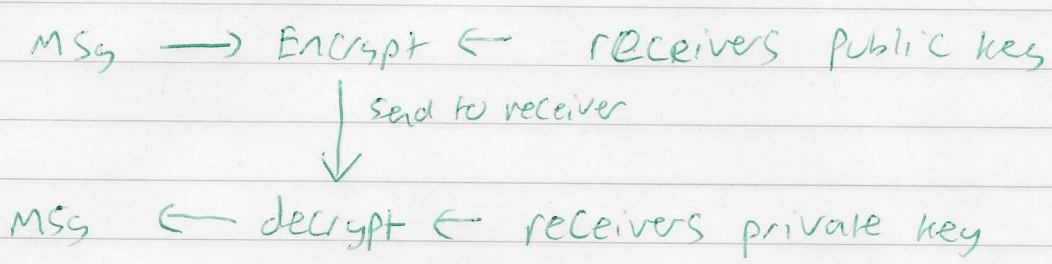
primitive roots

$k = y^x \bmod p$
If y is the primitive root mod p, then
1. Successive powers of y taken mod p generate all numbers from 1 to p-1

What does the primitive root mean?
- If we guess a value of x, the value of k is equally likely to be any number between 1 and p-1

Cryptography 5 - CS140

RSA encryption idea:



KP_x - Public key
KU_x - Private key

RSA encryption

1. Public key

Receiver chooses 2 large secret primes p and q.
Then calculates $n = p * q$.
~~Alice~~ publishes n as their public key.
Then need another publicly known value e.
How e is selected:
- relatively prime to $p-1$, $q-1$ and $(p-1)*(q-1)$
- $1 < e < (p-1)*(q-1)$

2. encryption.

Message = M
Ciphertext = C
$C = M^e \bmod n$

3. Private key and decryption

we know p, q, e and n
w find d s.t. - d is private key.
$e * d = 1 \bmod ((p-1)*(q-1))$ - d is the multiplicative inverse of e mod $(p-1)(q-1)$.

Message is then decrypted with.
$M = C^d \bmod n$

Why is RSA secure?

Attacker must either reverse one-way function (Too difficult) or know d, which requires prime factorisation (Too difficult) This leaves only brute force. A large key makes brute force ineffective.

Cryptographs 6 - CS1140

Encryption / decryption time

Encryption $O(k^2)$

decryption $O(k^3)$

Symmetry of public key encryption:

one direction

$M^e \bmod n = C$

$C^d \bmod n = M$

or

$(M^e \bmod n)^d \bmod n$

=

$(M^d \bmod n)^e \bmod n$

∴

Can encrypt with private key and decrypt with public

Ensure integrity

Sender encrypts msg with private ~~key~~ key.

Sends message and encrypted msg.

reciever decrypts message with senders public key.

If this = message then integrity is ensured

problem with second way of encryption.

Costly process to encrypt.

So instead of encrypting the whole msg we can generate a hash and encrypt that.

public key Vs secret key encryption

Secret key:

- uses XOR, substitution and permutation so faster.
- key is secret so attacker has less info.

Public key

- uses one way function so slower
- needs to have very long key so slower

Reasons to use private key for encryption

- Integrity
- Authentication
- Non-repudiation

Provide authentication and non-repudiation

If receiver uses senders public key to decrypt, this authenticates Sender.

If malicious message can be decrypted with senders public key then Sender cannot deny message was sent by them

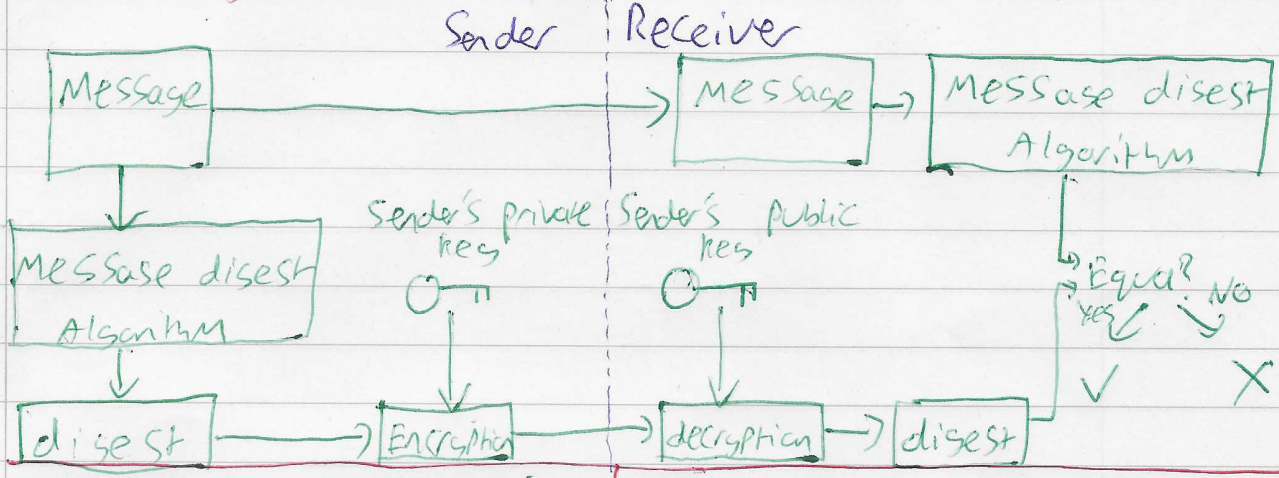
Digital Signature

Generated by encrypting a hash of the message with the Senders private key.

To verify, the receiver compares what they calculate the msg hash to be with the encrypted Signature sent with the message.

Digital Signature contd. – CS140

Digital Signature Scheme



Prevent Spoofing in email by digital Signature

- Create email (e)
- Create hash of email
- encrypt hash with Sender's private key (ku)

Final msg = $e + [hash(e)]_{ku}$

Can provide integrity, authentication and non-repudiation

Difference between encryption and DS

Encryption Scheme:

- Maintain Confidentiality
- Can recover plaintext

disital Signature

- provide integrity check
- authenticate
- Provide non-repudiation
- Cannot recover plaintext

Digital Signature vs. MAC

Disital Signature

- Use private key to encrypt the hash
- Can guarantee integrity and non-repudiation.

MAC

use shared secret key to encrypt the hash.

guarentees integrity but not non-repudiation

Message Authentication Code (MAC)

- Another way of providing integrity.
- Compute hash of the input file.
- Hash encrypted using the Shared Secret key.
- Can not provide non-repudiation
  - because MAC is created with the secret key.
  - Secret key is shared by more than one party.

Why do we need MAC?

MAC is much faster than digital Signature. As MAC uses secret key instead of public key encryption.

Digital Certificate - CS140

Certificate Authority

A CA is a "trusted" third parties.

CA issues certificates

Certifies public keys came from who they say it's from

X.509

This is a format to compose certificates.

Includes:

- Subject: distinguished name of user
- subjects public key
- CA's subject
- DS of CA

How to authenticate via CA

1. A asks for B's certificate
2. B sends certificate
3. Alice uses CA's public key to verify B's certificate
4. If B's certificate is genuine, then B's public key in B's certificate is genuine

Strong Authentication with certificate

- A's certificate proves the public key belongs to A
- Use A's public key to verify the message is really signed by A's private key.
- This message must come from A

Authentication Chain

- CA's can set up Sub-CA's e.g. CA1
- CA1 can also issue certificates to users
- Certificates issued by CA1 are signed by CA1
- CA1 also issues a root certificate to itself

Web of Trust

Alternate authentication scheme.

No Centralized CA.

Each user establishes his personal web of trust.

- Each user creates their own certificate
- users sign each others certificate

If A is certain B's PK belongs to B then validity is set to full

Otherwise set to unknown.

If A is confident that B will be careful when signing others certificates, then A sets "trust" of B to "full"

Otherwise "unknown"

Ultimate Trust

breaks restriction of threshold distance.

Security Protocol - CS140

Security protocol

Fixed pattern of exchanges between 2 or more communication parties to achieve a security-related task

DHM Steps

~~A chooses a secret number~~

2.

1. A and B decide on 2 publicly known values g and P.
2. A chooses secret number C
3. Puts C into one-way function $y^{x} \bmod P \equiv 0$
4. Send O to B
5. Use output received by B into the one-way function $y^{x} \bmod P \equiv K$

K is going to be the Shared Secret key.

B does the same steps

Diffie-Hellman-Merkle key exchange aims:

- A and B wish to communicate privately
- Communication channel is insecure
- Want to use secret key encryption
- How can they establish a secret key across an insecure channel?

DHM Conditions

- Y must be a primitive root of P.
- P must be enormously large
- A and B's secret numbers satisfy the conditions for selecting publicly known value e in RSA public key encryption

Authentication protocols - CS140

Authentication in local communication

working on a local terminal with a fixed link to the host we can be sure the connection is secure

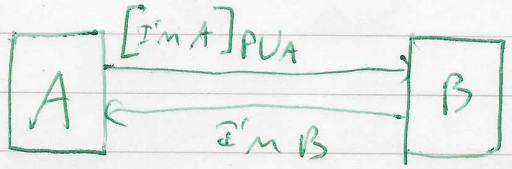
Authentication at a distance

We may need to authenticate a user via an insecure channel

passwords could still be used as a shared secret to provide authentication (need to encrypt it).

If you're not working in a password system you can use digital signature or public key encryption for authentication.

digital signature for Authentication



revisit DS notes

Replay attack

A middle man E can intercept A's msg to B and save it to use at a later date.

B will then mistake E for A.

Solutions to replay attack

1. Session token:
   - B generates a token R
   - A signs R
   - This authenticates A.
2. timestamping
   - A includes a timestamp in the msg.
   - IF E replays the msg the timestamp reveals the message is old.

Unilateral Vs. Mutual authentication

Unilateral: A authenticates B

1. $A \rightarrow B: A$
2. $B \rightarrow A: R$
3. $A \rightarrow B: [R]_{KU_A}$

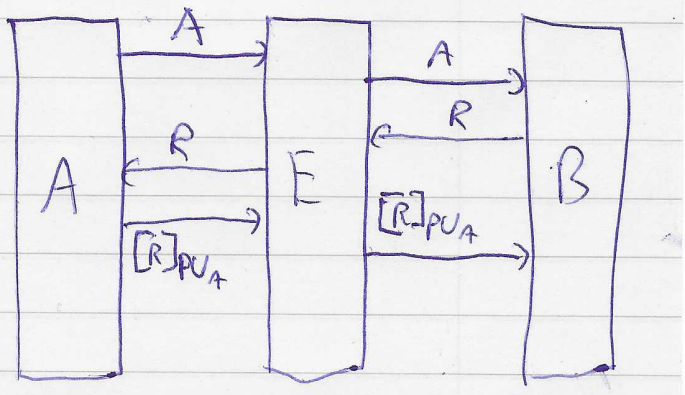
Mutual: do authentication check both ways.

1. $A \rightarrow B: A, R_A$
2. $B \rightarrow A: R_B, [R_A]_{KU_B}$
3. $A \rightarrow B: [R_B]_{KU_A}$

Authentication with encryption

1. $A \rightarrow B: A$
2. $B \rightarrow A: \{R\}_{KP_A}$
3. $A \rightarrow B: R$

Authentication Spoofing



Authentication protocols 2 - CS140

Solutions to Authentication Spoofing

Include identity of either sender or recipient in the protocol.

1. $A \to B: A$ digital signature
2. $B \to A: R$
3. $A \to B: [R, E]_{KU_A}$

1. $A \to B: A$ encryption
2. $B \to A: \{B, R\}_{KP_A}$
3. $A \to B: R$

possible attack

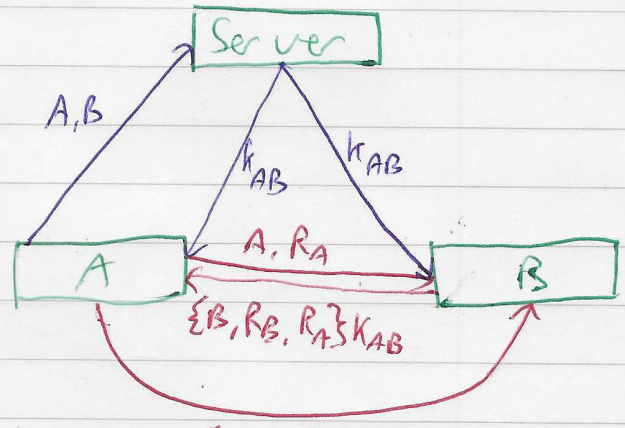
Suppose E grabs hold of session key K_{AB}.
Can then relay step 3 to B and B will complete the protocol thinking E is A.
This can be prevented with time stamps

Needham-Schroeder Authentication protocols.

encryption based, mutual authentication protocol.

1. $A \to B: A$
2. $B \to A: \{B, R\}_{KP_A}$
3. $A \to B: R$

Secret-key based authentication protocol



Problems:

- Server needs to distribute the secret key directly to B.
- B doesn't ask for key but receives one.
- Server may not reach B

Formal process

1. $A \to S: A, B, R_1$
2. $S \to A: \{R_{A1}, B, K_{AB}, \{K_{AB}, A\}K_{SB}\}K_{SA}$
3. $A \to B: \{K_{AB}, A\}K_{SB}, \{R_{A2}\}K_{AB}$
4. $B \to A: \{R_{A2} - 1, R_B\}K_{AB}$
5. $A \to B: \{R_B - 1\}K_{AB}$

A	Server	B
K_{SA}	K_{SA}, K_{SB}	K_{SB}

K_{AB} = Session key.

Virtualization and its security issues - CS 140

Virtualization technology
Allow multiple operating systems to share a computer
A running instance of an OS is called a virtual machine

Virtualization Architecture
Theres a hypervisor in between hardware and OS (s).

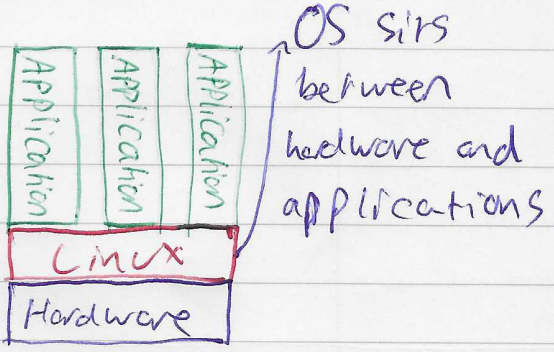
Domain 0.
- Domain management and control.
- Contain device drivers to access hardware.
- Interact with other VMs.

Abstraction of physical resources.
- Hardware is simulated as files in hypervisor.
- Files are transfered to real hardware when necessary

Isolation
each guest OS is encapsulated and the hardware is abstracted
A VM compromised by attackers will not affect the host or other VMs

Transcience
VMs are often only used when needed so are not left on so less prone to security risks

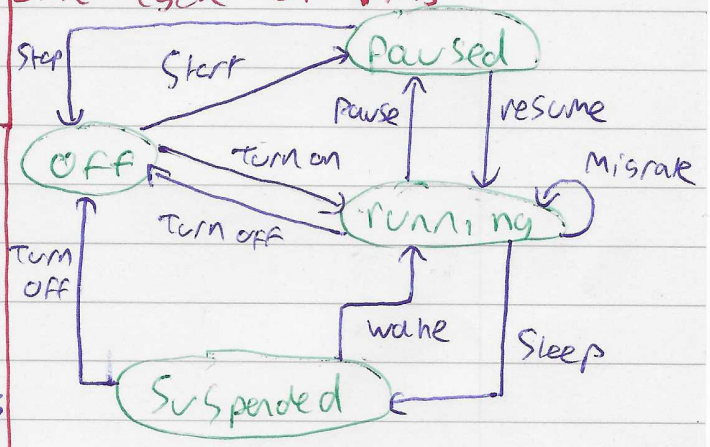
Architecture of a physical machine



OS manages the running of applications and the hardware resources
OS is lowest level software

Domain U
- No direct access to hardware.
- Request to access hardware is routed to domain 0.
- Share resources with other domains

Life cycle of VMs



State restoring
Virtual disk for a VM stored as a file in the hypervisor
VMs can use this to restore the state to a time before a VM was compromised.

Vai SI2 - CS140

External Moderating

VM has lower privilese than the hypervisor
VMs can be monitered by either:
- Hypervisor
- authorised dedicated VM

Latter is preferred to keep hypervisor simple.

Issues with State Recording

Fundamental principle of secure systems is minimizing the amount of time sensitive data remains in the system.
Virtualized systems keep states recorded which undermines this principle.

Issues with mobility.

VM is all virtual so is very easy to steal.
Also, if an intruder accesses the copy of a VM not being used by the Guest OS, the VM will not show any records of intrusion.
Attacker can access and modify while the VM is offline.

Easy creation - Issues

VMs can be created rapidly w/ different configs making it hard to apply uniform Sec measures

Issues with Transcience

In virtualized environments, infected VMs may appear and disappear before they can be detected.
Infected VMs can come, infect other VMs and dissapear before being detected.

Issues with State restoring

If the hypervisor is compromised The State restoring approach does not work and attackers have unlimited freedom and access the hardware
Since VMs can be restored many users do not secure their VMs with virus protection.
A Rollback will remove any security patches made after the state we are rolling back to.
- Challenging for admins to apply security patches

State restoring may re-expose vulnerabilities, re-enable disabled users or reuse retired encryption keys

Issues with Identity

All VMs have the same MAC address and multiples VMs can use the same port.
This stops Non-repudiation

Vai SI3 - CS140

Hypervisor intrusion

Hypervisor controls all VMs on a personal machine.
If hypervisor is compromised Attacker can access all VMs

For non-embedded hypervisor (as it runs on the host OS) If it is compromised, all VMs can be accessed and the host OS will be in danger as well.

Security due to Inter-VM Communication.

Attacker uses 1 VM to access/control other VMs on the same hypervisor.

This is achieved via inter-VM communication

Denial of Service (DoS)

Improperly configured hypervisor can allow 1 VM to use all resources. Thus starving other VMs of resources.

Solution: Hypervisor does not allow VMs to use 100% of resources