January 22nd- In class presentations

Kaspersky General

- Advanced persistent threats
    1. No More Big APTs- The security industry has become skilled at predicting and preventing security threats. Because of this, it is likely that attackers will begin to make more attacks rather than more skilled (predictable) attacks. This makes it harder to determine where the attack is coming from.
    2. Network hardware and Internet of Things- If they compromise a network, they can use the machines as a botnet to attack and disrupt other devices or infrastructure, as well as using it as a means of communication.
    3. Public Retaliation- Attacks have influenced public opinion in the past, so there will likely be more attacks used to create propaganda and influence people.
    4. Exploitation Tools are easy to come by and can be used by inexperienced people to take advantage of a network
    5. Malware rings- The kernel mode (ring 0)
        - If the device is functioning and there is malware inside of it, it has the ability to send screenshots to the attacker
    6. Spear phishing- Social media leads to stealing the credentials and information to sell or use to assist in a further attack
    7. Destructive Destroyer- Deleting important files/sabotage(generally political)
    8. Advance Supply Chains- Injecting malicious code in public repositories
    9. Crypto currency for financial gain

Kaspersky Cryptocurrency

- What happened in 2018?
    - Price increased in the 1st half of 2018 and there was an increase in attacks
    - Attackers would mine other peoples Bitcoins without them knowing
    - In the second half of 2018, the price began to drop because of this
- Predictions for 2019:
    1. People will stop trying to implement block chain

2. People will use cryptocurrency less so the price will drop more
3. There will be no return to the high exchange rates like in 2017

Kaspersky Financial

- There has been an increase in automation so people do not have to be in the office as much. This increases the risk of being attacked.
- The interest for industrial enterprises will go up. Remote access will be a weak point.
- Since industry is easy to disrupt, it will be an easy target
- Industrial threats move at a much slower rate
- The general estimation of threat levels:
    - When a threat occurs on an automation system and no serious damage occurs, it will likely go unnoted.
    - We will only focus on systems that alert large emergencies
    - How do we handle large threats?
- Attacks on social media, through theft, and on small companies

Beyond Trust

- Attack Vectors/Targets: Objectives of the attackers
    1. Privilege Attack: Stems from poor password use.
    2. AI: Systems will see an increased number of attacks from the computer learning
    3. Corporate Attacks: More attacks on homebased routers that destroy the computer from the inside
- Android plans to cut off third parties
- Millennial security: What we deem as private today may not be private tomorrow
- The rise of information brokers: Trust companies with your information

Fire Eye

- Interviewed people within the Fire Eye company
- CEO laid out 3 trends
    1. Nations going into cyber offence
    2. No proper rules of engagement/punishment

3. Not enough resources unless a big company

- 3 Solutions:
  1. Update defenses
  2. Educate people
  3. Work on diplomacy: if one person commits an international crime, outline how to deal with that
- Intelligence declassified- upcoming risks
  - AI could make specified attacks for certain companies
  - More attacks on mobile devices which could cause more harm. Not a risk now because quantum computing and AI are not advanced enough
- Cybersecurity as it leads to air travel
  - The more pressing concern is the prevalence of ransomware regarding flight information systems. Large airlines will pay a lot of money to make sure their systems stay online.
  - It is predicted that there will be more attacks on airports because the jurisdiction is not always very clear
- Cybersecurity and hacking—Newcomers and how they use hackers to achieve their government goals
  - Online stores and how hacking will increase in 2019. They will hack online stores to get credit card information
  - Use reverse engineering to hack online banking accounts
  - The supply chain and how some companies can hack other companies for their ideas
- Threat intelligence
  - In 2015, Iran signed a deal with the US for coexistence to cut down on Iranian threats and cyber activity.
  - They had been trying to hack the water and amenities. This occurrence has decreased since the deal was signed
  - Governments world wide are debating on setting up cyber standards
  - Large companies have similar agreements but governments currently do not.
- Global Insights
  - It is expected that attacks will happen at the Olympics

    - Expected for low level hackers to access tools for cybercrimes: need to focus on the individual rather than large groups
    - Increase influence from social media narratives
    - Quite a few attacks from companies outsourcing third parties
    - More critical infrastructure attacks
    - Companies focused more on compliance when they should be focusing more on security efforts
- Overall
    - Accountability in cyberwarfare—need more laws for cyber crimes
    - Cloud storage is becoming more popular so it is likely that there will be more attacks happening in the cloud
    - In addition to planes being attacked, there is fear of customer information being stolen and fake tickets being sold. This scares the potential customers and could lead to more loss to the company from people being too afraid to fly.
- Hacker Predictions
    - As more and more people move their information to the cloud, hackers will be less likely to attack personal computers
    - The Internet of Things will become more popular and attacks will become more complex
    - Social Engineering is the most common way that people were attacked in 2018
    - People were using password protected attacks that led the user to the virus
    - Increase AI products deployed that were supposed to monitor attacks. Attackers were finding ways to avoid that by going around these programs or blending in with non-malicious ones.
- Staffing Cloud and Consolidation
    - 2-3 Millon jobs without workers—if every student were to take one, they would still not be filled
    - How to train the current employees to cover these open spots
    - Everyone moving to the cloud—Although companies have a firewall, it does not completely protect them from attacks

- You cannot spend the same amount of time protecting everything, you need to choose your focus
- It is important to have visibility in the cloud—Who is doing what in the cloud and who has permission?
- Is someone in the cloud trying to get access to something that they are not supposed to?
- Sim card spoofing
- Know what is going on in the cloud

Due Dates:

1/29 Assignment due

1/31 Presentation about success in cybersecurity (Choose topic and put on class GoogleDoc)

2/5 Quiz

January 24th – In class presentations (continued)

Forcepoint

- Vast number of unfilled cyber security jobs
    - Report predicted that there is no true cybersecurity AI right now and the chance that there will be one in 2019 is very slim
    - There may be machine learning but not true AI
    - With misconceptions, there are some over exaggerations of the AI
    - A slowdown for AI funding is eminent
- Attackers will interrupt industrial IoT devices
    - Since they are networked and always on, it could result in a manufacturing disruption
        1. Increasing network connectivity to edge computing
        2. Difficulty in securing devices
        3. Exponential number of devices connected to the cloud
    - Fear that since companies are focusing on speed, that could be taken advantage of.
- A counterfeit reflection
    - Ways that we log on and interact with our computers puts our security at risk.
    - Username and password aren't secure because that doesn't prove that you are who you say you are.
    - Two factor authentication could help this but is still not bullet proof
    - The next step is biometrics. The prediction is that "behavioral biometrics" would become more popular by monitoring how you use the computer to make sure you are that person
- Court room face off

    - It is expected that there will be a case involving deliberate data breech in 2019.
    - 24% of UK employees have unintentionally given out information that they are not supposed to.
    - Tesla Motors employee who leaked information claimed to be whistle blower but the company said he was just a disgruntled employee. This case became more of a "my word against his" case.
- A collision course to cyber cold war
    - Embargos are being raised on all types of goods including software
    - Raise in IP theft is expected
    - There is incentive to get access to software through any means possible—reminiscent of the cold war.
    - Through the years to come, governments may become stricter on the protection of their software, leading to a split between the nations
- People are now starting to trust companies to protect their privacy and data
- Cybersecurity cultures that do not adapt and fail
    - More businesses are relying on technology to store their private data
    - In order for companies to have Integrity, they must implement security from top to bottom.

IBM

- How social security numbers are going to be used less
    - Benefit programs are one of the few that still use them
- With GDPR (General Data Protection Regulation) coming in, they had to get rid of the WHIS that helped them identify malicious domains. These domains will likely increase because of this.
- Automated customer service stations will likely be hacked more often in the next year
- Companies are relying more on insurance to cover them, rather than actively protecting their data
- Mega hacking will become more and more prominent
    - Hotels and airlines will be targeted because they hold a lot of data

- Ten predictions for 2018 (all came true):
    1. The deadline for GDPR was in May and he predicted that many companies would not be compliant
    2. Ransomware was going to be the most dangerous threat
    3. Cryptocurrency would be targeted
    4. APT groups would continue to increase pressure on Western organizations
    5. Cloud security would be a top priority
    6. Cyberbullying continues to be an issue and has even increased
    7. IoT devices and mobile device threats are becoming more prevalent
- 2019 Predictions
    1. Cloud storage and IoT will be targeted
    2. Increase in cybercrime as a service
    3. Cyberbullying involving cybercrime where people gathering data can use it to embarrass people
    4. ICS threat—Vulnerabilities and attacks will become much more prevalent

McAfee

- 2019 Predictions
    - Coding competition where you could go and encrypt some software
    - Doing more by adding features like adding RDP
    - Bringing in artificial learning
    - AI will increasingly become an attack vector
    - Hackers will now make attack vectors that create a multipronged attack
    - Social Media is becoming known as an attack vector
- Expectation based on predictions
    - Found that 22% of data in the cloud is sensitive
    - With people filling their home with devices, the network is growing.
    - McAfee expects tablets and smartphones to be targeted

Sophos

- The news coverage regarding the security industry is predominantly negative while the other industries are celebrated for their successes.
    - The general public does not hear about the measure being put in place to protect this
    - Attackers are having to ramp up their skill set to combat to defenses as well as spend more money to keep their operations going
    - In 2018, there was an advanced persistent threat that was prevalent
- One thing that we are praised for is when something we do forces attackers to change
- Popular ransomware applications
    - The easiest way to defend it is block off ports by default
    - Kit that can be downloaded to protect services
    - Big market in developing ransomware and attacking people
    - Attacks are carried out hundreds of times a day so everyone must be protected
    - Ransomware is a serious threat that attackers have put a lot of money in
- Attacker techniques with what is already on your system
- Risky file types
    - Over the past few years, the list of risky file types has expanded
    - The best thing to do is just download from somewhere that you know is safe
- The growing and persistent threat of malware
    - Two types of malware:
        1. Phishing in the app
        2. Crypto mining code that can run without the app being run
    - Increase in IoT devices as botnets

Symantic

- Attackers and Defenders using AI
    - Attackers will use AI to make more sophisticated attacks
    - Defenders will depend increasingly on AI to counter attacks and identify vulnerabilities

        - Growing of the 5G infrastructure and development—A catalyst for more models, architectures, and vulnerabilities
        - The fear is that entire countries will be attacked by shutting off utilities
        - Rise in new attacks on home base WIFI routers
        - Companies are going to third parties to keep track of the personal information to keep it more secure and make them less liable
        - Exploitation of supply chain
        - Attacks have become increasingly common
            1. Attacker replaces software update
            2. Attackers try to compromise chip
- Security and privacy concerns
    - More regulations and more laws will probably be passed out in the upcoming year to protect the users

Trend Micro

- Social Engineering will replace exploit kits as an attack vector
    - The number of identity thefts increased
- Enterprises
    - Employees working from home are creating vulnerabilities
    - Through an open port, attackers could take complete control of speakers and get access to files on the network
- Government
    - The fight against fake news
    - Social media allows fake news to spread easily
    - As governments gear up for cyberwar, innocent civilians will be harmed
- Security Industry
    - Cyber criminals will use more techniques
    - Cyber criminals will create new techniques to attack
    - There are going to have more engineers behind cyber security because we cannot rely on software defense tools
- Industrial Control Systems

    - Some countries might attack other countries to hurt their economy
    - Human lives may be in danger as well
- Cloud Infrastructure
    - More cloud related vulnerabilities will be discovered
    - As more businesses use the cloud, more research will take place
- Router based attacks
    - Routers will continue to be an attack vector for cybercriminals wanting to take control of connected devices
    - Senior citizens may be attacked leading to their personal information being leaked

Watchguard

- Security Predictions
    1. AI chatbots will go rogue and steal information
    2. Utilities and Industrial control will be targeted with ransomware
    3. The UN will have to have a cybersecurity treaty
    4. Infrastructure shut downs by simple nefarious acts to terrorize people into doing what they want
    5. File-less, self-propagating vapor worms attack
    6. WPA3 encryption will be circumvented by threats
    7. Major biometric hack with single-factor identification
    8. Attackers will hold major domains hostage

January 29th

Deadlines:
2/14- Proposal due (1 paragraph: 250 words on what you want to do) with timeline and team members
3/12- Progress report due (1 page on what you have done so far) with updated timeline & team member evaluation

Last 2 Presentations:

Zscalar

1. Increase in attacks that target cloud applications
2. The government is going to start looking toward the private sector for their own cloud security
3. More state employed white hat hackers will moonlight with criminal organizations
    - This is very popular in Russia
4. Cyber risk rather than spending for cyber security
    - Look to find more efficient ways to protect against cyberattacks
5. Increase in IoT exploits and botnet recruitment
    - IoT increases every year
6. Cyber-attacks will affect stock prices
7. Security market will consolidate
    - “Complexity is a problem”
8. Attackers will also consolidate to also strengthen themselves
9. Supply chains security will increase

Palo alto

1. Cryptocurrency malware will either stay at the same place as it is now, or it will rise
2. Attacks on macros embedded in documents will increase
3. Email attacks will increase
    - Spearphishing

In class assignments:

1. Classify each of the following as a violation of confidentiality, integrity, availability, or a combination:
   a. John peeks at Alice's password when she is logging in: **Confidentiality**
   b. John logs into Alice's account using her password without her knowing about it: **Integrity** (origin)
   c. There is a process running in Alice's machine, which is updating a database from a remote machine. John interrupts the process, results in inconsistent databases. **Availability** and **Integrity**
   d. John copies a file from Alice's account and then deletes the file from Alice's directory: **Confidentiality** and **Availability**

2. Confidentiality > Integrity:
   - Classified Information (inherently confidential information)
   - SSN

   Integrity > Confidentiality:
   - Bank Statements
   - Medical information
   - Votes in an election

3. The Principles:
   - Open design → No Secrecy
   - Fail-safe defaults → No default permit
   - Least privilege → No more privilege than needed
   - Economy of mechanism → No complexity
   - Separation of privileges → No single responsibility
   - Complete mediation → No bypass
   - Least common mechanism → No sharing
   - Psychological acceptability → No hardships/surprises

Security Principles

- 1975 Saltzer and Schroeder's fundamental principles of security
- Benefit:
    - Used in design, implementation, and/or configuration to prevent loopholes
    - Used as the basis of a review checklist
- Main goal: **restriction with simplicity**
    - Only a small part of security incidents are intentional.
    - We want to restrict what accidents might happen

The Principles

1. **Least Privilege**
    - A user/application/service should be given only those **privileges** *necessary* to complete the task
        - Privilege means **permissions** determining direct actions on the entity in question
        - Function controls assignment
        - Minimal set of rights
            - Rights added as needed, discarded after use
            - Ex: If you give a user more privilege that they need, the damages that they could cause could be great
    - **Principle of Least Authority**
        - A user/application/service should be given only those **authorities** as *necessary* to complete its task
        - Authority means what **effects** it has on the entity in question either **directly** or **indirectly** through another user/application/service
        - If it is important to have access to the file, then you should either remove the permission to communicate or the permissions to the file
2. **Fail-Safe Defaults**
    a) Default action is to deny access
        - **Exclude-fail** is better that **permit-fail**

- Better restrict permissions more than needed rather than accidentally allowing everyone in
- People that need the permissions will let you know. People that don't need the permissions will not tell you that they have them

    - Permit as needed

b) If unable to complete task, undo
    - If something does not execute properly, it rolls back to the last secure/safe state

3. **Economy of Mechanism**
    a) Keep it as simple as possible
    b) Simpler means that less can go wrong
4. **Complete Mediation**
    - Check *every* access, *every* time
        - There could be changes in a process since the last time it has gone through
    - No bypass
5. **Open Design**
    - Strength of security should not depend on secrecy of design or implementation (or configuration)
        - Does not apply to information such as passwords or cryptographic keys
6. **Separation of privilege**
    - Require multiple conditions to grant privilege/access
        - Separation of duty
        - Example: 2 Factor Authorization, bank vault requires two keys
        - Users working together to accomplish a task
7. **Least Common Mechanism**
    - Mechanisms/Resources should not be shared
        - Information can flow along shared channels
8. **Psychological Acceptability**
    - Security mechanisms should not add difficulty of accessing resource
        - Hide complexity introduced by security mechanisms
        - Ease of installation, configuration, use

    - Principle of Least Astonishment
        - Mechanisms should be designed so that users understand the reason the mechanism works the way it does and it's simple to understand
            - The result of performing some operation should be obvious, consistent, and predictable

Term Project—Education, Outreach, Research

- What we will do:
    - Investigate
    - Integrate & Analyze
    - Innovate
    - Implement
    - Assess
    - Report
- Types
    - Idea
    - Survey
    - Lesson learned
    - Case study
- Idea Paper
    - New solution
    - New problem
    - Application of existing solution to existing problem

January 31st—Reports on Positive News in Cybersecurity

Government involvement in cybersecurity

- Hit with a lot of attacks lately
- Recently, since 90% of internet stuff is privately managed, the government has decided to step in
- They decided to pass new regulations on how to handle defense and offense for a safer experience

UK has new funding to drive diversity

- At least 500,000p toward cybersecurity jobs and training

Stanford article about surviving two ransomware attacks

- They were protected because:
    1. Constantly updating software
    2. Education of community to recognize and report

Guideline summary for connected security systems

Site taken down

- Crime agency locked site that allowed people to purchase DDOS attacks
- Site taken down and suspects arrested

Gen Cyber

- Program funded by NSA and NSF to educate kids on cybersecurity principles through games
- Most kids enjoyed it and were interested after

D3 security case management

- Digital forensics could not handle intake but system was created to handle this volume

TVA cybersecurity

- Taking proactive measures and employing more people with state-of-the-art facility

A group searched to find as many websites as possible with a specific virus

- They found multiple and were able to contact the providers and have them shut down

President Trump signing in bill

- Authorized the education of people in law enforcement so they would be able to make arrests in cybercrime

Holdings Inc 5 new laws

- New York state department of financial services regulation
- Designated chief security officer, risk assessment, security events, annual penetration testing, vulnerability assessment

Oil and gas pipelines implementing blockchain

- Pipelines have had sensors digitized so they can be accessed from anywhere
- Incidents of ransomware that could threaten spill or exposure
- Blockchain means that there is not one single access point. This creates a more secure connection

Cybersecurity in medical devices

- US DFA draft guidance document that goes over cybersecurity guidelines that medical professionals should go over before releasing products

Researchers take down of 52,000 servers

- Eltest found on dark web
- 2 million users effected—mobile devices part of botnets
- Able to track down payment history of Eltest to buyer
- Thanks to research, most were taken down

Europe's new law—GDPR

- Protects individuals in countries
- Spell out why data is being accessed
- Users can access the information about them and control that information
- Users have more say in what they do with information
- Privacy and cybersecurity will be taken more seriously
- Violators will face fines based on revenue

California Law

- Everyone who wants to do business in CA has to have reasonable security measures

Grey Hat hacker cleaning up software

- Vulnerability in brand of routers
- Hacker was a system admin who updates routers for people to keep them from being hacked
- Over 100,000 routers
- These are now secured and no longer vulnerable

Idea of cyberwarriors stop major ISIS attack

- Analyze and track communication between terrorist groups
- Saved a lot of lives by stopping the communication

Wannacry attack

- Ransomware attack from 2017 that encrypted files and held them for ransom unless you paid
- Based on prime numbers to encrypt and decrypt
- Would delete from history but not memory
- Wannacry only on windows systems

National Australian bank claims cybersecurity history

- Races to find all the flags in the system
- Goal to encourage professionals to increase their skills and speed

Network Engineer

- Ransomware attack that encrypted network share with patient data

Cybersecurity company helped Facebook to stop spreading false information

- Fireeye noticed many pages owned by a new site
- The person who owned the site did not add up
- Purged 652 false accounts and pages

8-million-dollar investment into cybersecurity platform

- Grow the knowledge of cybersecurity experts
- Keep cyber professionals' skills constantly updated

Lack of cyber professionals

- Reward 20-million-dollar grant to distribute cyber-curriculum to K-12 children
- 25 course progressive-learning
- Result in approx. 7 million cyber literate students
- Better foundation

Female Workforce

- The percentage of women in the cybersecurity workforce was projected to increase by about 10%

Disposed IO

- Aims to standardize best practices
- The connotation of hacker and ambiguity of the language/laws for hacking
- Hackers with good intentions had faced jailtime
- New legal language to protect organization and hackers

- There were 18 organizations that had adopted the language and this is now up to 26

Black market site taken down

- They had disguised themselves in the open
- 36 names were released and the site was taken down
- 17 of these people were arrested (at time of article)

Budget aspects of cybersecurity

- Security implemented in new technologies
- Business growing across the country

National Cyber Security Center in the UK—Russian intrusion into UK and US

- Companies thought that it was too hard to understand when they were infected
- Increase the amount of conferences to educate them

FBI announced 13 count indictment

- Resulted in FBI being able to take down two add networks
- Identify crime network in Germany

Michael Fugatt

- Veteran that applied for IT jobs and was turned away
- Went through program and was able to get a job and turn his life around

California Consumer Privacy Act

- Modeled after GDPR
- Allows people to know what data of theirs has been collected
- Hopefully increase federal regulation and privacy to prevent data breaches

Ransomware

- Tool was released that allows people to decrypt ransomware

Girl Scouts of America

- 3 Cybersecurity badges introduced

NC judge ruled that biometric access was under the same umbrella as passwords

3 Ukrainian members of a hacker group arrested

- They had hacked over 100 companies through phishing emails

Increased production of devices that work together and help them to fix the vulnerability of pacemakers and make them more safe

Non-profit cybersecurity organization

- Found sites and contacted provider to take down sites
- 4,000-5,000 sites per day

Investment in cybersecurity continues to rise

- Security testing is expected to grow
- Net worth expected to grow 11% each year

Iowa state university working with electric company to secure power grid

- Power companies are generally behind so good that it is being assisted

MIT—exploit in intel processors

- Works on system "speculative exploitation"
- DAWG—dynamically allocated way-guard

Dark-website taken down

- Specialized in drug trafficking
- One of Finland's largest hosted websites
- Investigations were being done on a cybercriminal when they found this site

Cybersecurity review of 2018

- In June, the US took down 70 email scammers

Congress beginning to see cybersecurity as important

- Election security—distributing funds to states to increase security
- Impose fines in case of breach in security
- Congressmen making runs for presidential election will lead to more focus on cybersecurity
- Increase in cybersecurity defense as well as more cybersecurity conscious elected officials

Microsoft shutdown websites owned by hacker groups

- Targeted tanks, the Senate, and Microsoft
- Clever URLs to convince people to browse websites
- Taken down before they could be used in any major cyber-attacks

Estonia cybersecurity

- In 2018, there was a multi-volunteer cybersecurity army that collaborated with official defense to do training in the cyber space for a more robust defense
- US looking into countries like this to build their own defense

Cybersecurity taking center stage

- States emerging as cyber-leaders

New field risen

- Hunters go seeking attacks

Live-patching for Linux

- Solves the problem with the kernel
- Atomic live-patching

- Never have to restart system to get security updates

Using AI and machine learning to form new defenses and attacks

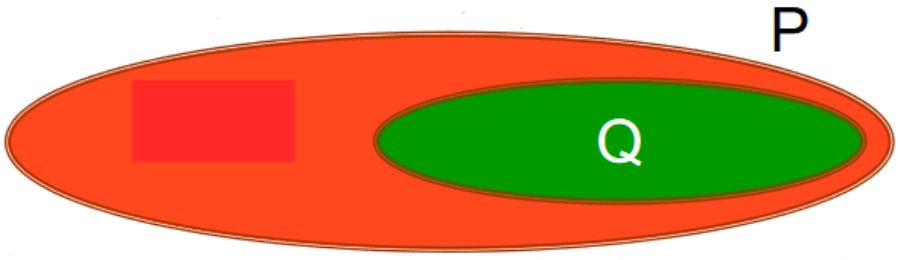
February 5th—Security Policies and Models

Policy, Model, and Mechanism

- **Security Policy**
  - Statement of what is (and is not) allowed
- **Security Model**
  - Representation of policy
- **Security Mechanism**
  - Methods, tools, and procedures to enforce policy by implementing model

MECHANISM → implements → **MODEL** → formalizes → **POLICY**

- **Protection State** (P): System states relevant to protection
- **Authorized State** (Q): System states authorized to reside
  - Security Policy – Identifies Q within P
  - Security Model – Characterizes Q more formally
  - Security Mechanism – Ensures Q and prevents P – Q



- Example:
  - Password protected credit card account: **Mechanism**
  - Only active students can access Eagle Online: **Policy**
  - Iff x is a subset of all students, X, in TTU representing active students, then X can have access to Eagle Online: **Model**

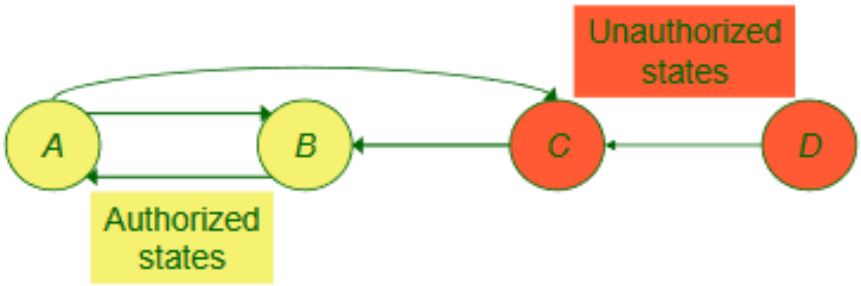
Security Policy

- The set of laws, rules, and practices that regulate how an organization manages, protects, and distributes sensitive information
- Two Types:
  1. **Military**:
     - Focus on *confidentiality*
     - Expressed with security labels:
       - Classification of objects and Clearance of subjects
       - Combination of *hierarchal* sensitivity and non-hierarchical compartments

    - Access is controlled by security labels/flags, “need to know” rules
    - All about how you classify subjects/objects
  2. **Commercial Policies**
    - Focus on *integrity*, *availability*, and *non-hierarchical confidentiality*
- From behavioral perspective, if we consider computer systems as finite-state machines with
    - Set of states
    - Set transition functions that change state
- Then we can define security policy as on which partitions system states into
    - Authorized states
    - Unauthorized states

Finite State Machine Example

- Secure System
    - Starts in authorized state
    - Never enters unauthorized state



- The diagram shows an <u>unsecure</u> system because you can access C which is an unauthorized state

Confidentiality Property

- X set of entities, I information
- I has *confidentiality* property with respect to X
    - If **_NO_** x ∈ X can **obtain** information from I
        - NO , ANY, or ALL
        - I is confidential to X if no member from x can gain access to I

Confidentiality Policy

- About *secrecy* of information
    - Does not care about trust
- Confidentiality levels designate the extent of secrecy
    - Ex. Military environments
- Deals with prevention of:

    - Unauthorized rights
    - Transfer of information without transfer of rights (*information flow*)
- Temporal Context (*dynamic change of rights*)
- Highly developed in Military/Government

Integrity Property

- X set of entities, I information
- I has integrity property with respect to X
    - If **_ALL_** $x \in X$ **trust** information in I
- Types of integrity:
    - Trust I itself, its conveyance and storage (*Data Integrity*)
    - Trust I, where I is information about origin of something or an identity (*Origin Integrity*, authentication)
    - Trust I, where I is resource: means resource functions correctly
- About **trustworthiness** of information
    - Does not care about secrecy
- Integrity levels designate levels of trustworthiness
- Deals with
    - Who is allowed to change the data?
    - Conditions under which data can be altered
    - Limiting change of data
- Highly developed in commercial world

Availability Property

- X set of entities, I data/resource
- I has availability property with respect to X
    - If **_ALL_** $x \in X$ can **access** I
- Types of availability
    - **Traditional**: you have access or you don't (*Binary*)
    - **Quality of Service**: A level of access (*Graded*)
- Deals with
    - What services must be provided
    - To what extent
- Mostly developed in commercial world

Enforcing Policy

- Explicit Policy: X cannot view Y's notes
- Implicit Policy: Y must protect notes
    - If X cheats, *only* X can be held accountable

- Explicit Policy: X cannot view Y's notes
  Y must protect notes
  - If X cheats, *both* X and Y can be held accountable
- Security policy must be as *explicit* as possible
- Security model must be *unambiguous* and *explicit*

Trust Relationships

- Trust and assumption play crucial role in policy, especially integrity policy
  - Trust is based on assumptions
  - Attackers look for assumptions and trusted users to find possible weak points in implementation of policy

Role of Trust

- **Higher level assumption example**:
  Administrator patch with the trust that…
  - Patch came from vendor
  - Not tampered with in transit
  - Vendor tested patch thoroughly
  - Vendors test environment corresponds to local environment
  - Patch is installed correctly

- You are making all of these assumptions… assumptions are dangerous

- **Lower level assumption example**:
  A security-related program S is formally verified to work with operating system O
  - Proof has no errors
    - Bugs in automated theorem provers
  - Preconditions hold in environment in which S is to be used
  - S transformed into executable whose actions follow source code
    - Compiler bugs, linker/loader/library problems
  - Hardware executes the program as intended
    - Hardware bugs

- There is no way to not make assumptions, but you need to document them and know where they can get invalidated

"Secure" vs. "Trusted"

- Secure System
  - A Goal
  - Asserted based on features
  - Either/or
- Trusted System
  - A Characteristic
  - Judged based on evidence analysis
  - Degree of Trustworthiness

Security Model

- Primary purposes
  - Provide a framework to aid in understanding
  - Provide an *unambiguous*, often formal, representation of a general security policy
- Focuses on
  - Points of interest in policies
  - Specific characteristics of policies

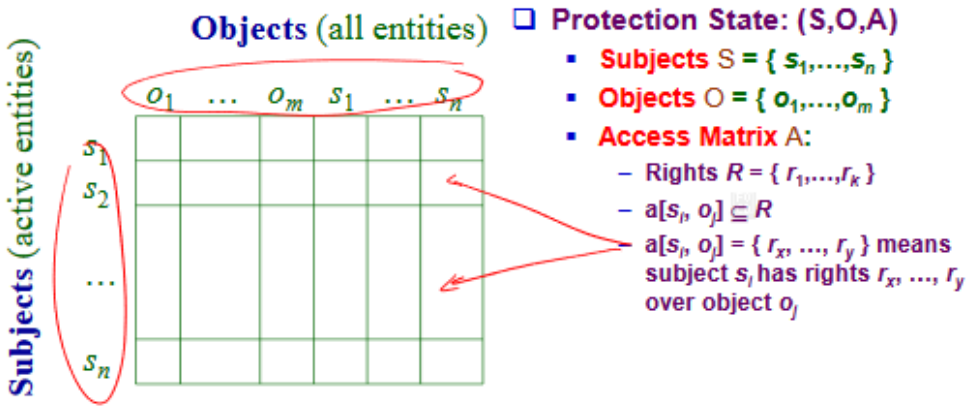
Example Security Models

- Access Control Matrix
- Bell LaPadula
- Biba Integrity
- Chinese Wall
- Clinical Information Systems Security (CISS) Policy
- Clark-Wilson

Access Control Matrix (ACM) Model

- Characterizes access right relationships between subjects and objects
- Specification in the form of "Matrix" framework
  - A **row** shows the *rights of one entity* (over the entities in the protection system)
  - A **column** shows the *rights over one entity* (by other entities in the protection system)

ACM Representation



Example ACM 1

- Process 1 can read or write file 1 and can read file 2
- Process 2 can append to file 1 and read file 2
- Process 1 can communicate with process 2 by writing to it
- Process 2 can read from process 1
- Each process owns itself and the file with the same number
    - Processes: p1, p2
    - Files: f1, f2
    - Rights: r, w, x, a, o

	f1	f2	p1	p2
p1	rwo	r	o	w
p2	a	ro	r	o

February 7th-

Security Policies and Models (continued)

Example ACM 2

Subjects & Objects: Telegraph, nob, toadflax

Rights: own, ftp, nfs, mail

	telegraph	nob	toadflax
telegraph	own	ftp	ftp
nob		ftp, nfs, mail, own	ftp, nfs, mail
toadflax		ftp, mail	ftp, nfs, mail, own

The subject **telegraph** is a personal computer with **ftp client** but no servers, so neither of the other systems can access it, but it can **ftp to them**. The subject **nob** is configured to **provide NFS service** to a set of clients that does not include the host **toadflax**, and both systems will exchange mail with any host and allow **any host to use ftp**.

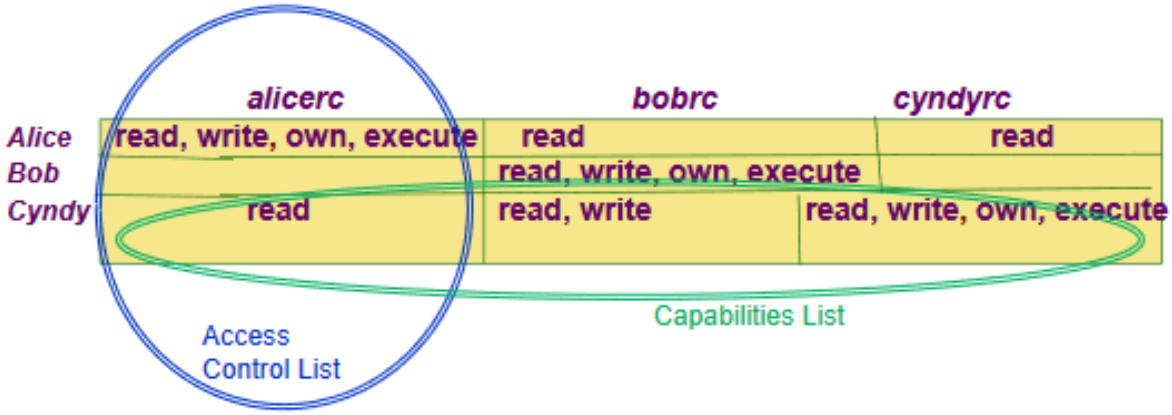
- If you are told to put down everything that is not explicitly denied, then you should add any implicit information

Reference Monitor and Validation Mechanism

- **Reference Monitor** (RM) – access control concept of an abstract machine that mediates all accesses to objects by subjects
- **Reference Validation Mechanism** (RVM) – an implementation of the reference monitor concept
- Properties (Should be true for any ACM mechanism)
    - RVM is Complete (always invoked)
    - RVM is Tamperproof (cannot be compromised)
    - RVM is Simple (verifiable)

ACM Implementation in Operating Systems

- In practical, since the access control matrix is "sparse"
    - Implementation is either by row or by column
    - Row-wise implementation
        - **Capabilities List** (per subject)
        - Each subject has a list of objects it can access containing allowable ways to access
    - Column-wise implementation
        - **Access Control List** (per object)
        - Each object has a list of subjects that can access it containing allowable ways to be accessed



- If you have lots of objects, and few subjects, then you should create a capabilities list
    - If someone is fired and you need to revoke them, all you need to do is access the capabilities list and revoke that, rather than pull up each object
    - If you need to remove a secret file, then you can use the Access Control List

Study the practicalities behind the Capabilities List and Access Control List and when you would use each of them

Capabilities List

- Aka Directory list
- Each subject is associated with a list of (o, A[s,o]) – which are assumed unforgeable
- Subject presents its "ticket" or "capability" to the Reference Monitor when making a request
    - ✓ Easier implementation
    - ✓ Easier to revoke access for certain subject
    - Diffuse concept of ownership (per object)
    - Difficult to revoke access for certain object

Access Control Lists

- Each object is associated with a list of (s, A[s, o]) – assumed unforgeable
- Reference Monitor must search all objects for each access request by a subject
    - ✓ Strong Ownership (per object)
    - ✓ Easier to revoke access for certain object
    - Difficult implementation
    - Difficult to revoke access for certain subject

Security and Safety

- Safety
    - Refers to abstract model
    - Does a model that is safe w.r.t. all rights (privileges) guarantee a secure system?
        - ***No**, there could be flaws associated with the implementation of it
- Security
    - Refers to implementation
    - Does a secure (rather trusted) system corresponds to a model that is safe w.r.t. all rights?
        - ***Yes**, if something is trusted, then it must have come from a safe model.

Security Models – BLP, Biba

Bell-LaPadula Model

- Cornerstone of many security models
- Proposed by David E. Bell and Len LaPadula in 1973 to formalize DoD multilevel security policy
- Focus on secrecy of information
    - Protects flow of information
- Based on hierarchical confidentiality
    - Security levels arranged in linear ordering
        - Subjects have *security clearance* L(s)
        - Objects have *security classification* L(o)
- Combines two (for reading and writing) mandatory access control rules (relationship of security levels) with discretionary access control (the required permission)

Reading Information

- **Information flows UP***
- "Reads up" disallowed, "reads down" allowed
    - Sometimes called "no reads up" rule
- Simple Security Condition (1)
    - Subject s can read object o iff L(o) <= L(s) and s has permission to read o
    - Subjects can view content at or below their own security level
    - Cannot read anything higher than your information level

Writing Information

- Information flows UP
- "Writes up" allowed, "writes down" disallowed
    - Sometimes called "no writes down" rule
- *-Property (1)
    - Subject s can write object o iff L(s) <= L(o) and s has permission to write o
    - Subjects can create content at or above their own security level

Example

security level	*subject*	*object*
Top Secret	Tamara	Personnel Files
Secret	Samuel	E-Mail Files
Confidential	Claire	Activity Logs
Unclassified	Ulaley	Telephone Lists

- Tamara can read Personnel Files, E-Mail Files, Activity Logs, Telephone Lists, but can only write Personnel Files
- Ulaley can write to Personnel Files, E-Mail Files, Activity Logs, Telephone Lists, but can only read Telephone Lists

Basic Security Theorem

- If a system is initially in a secure state, and every transition of the system satisfies the simple security condition (1), and the *-property, then every state of the system is secure.

Bell-LaPadula Model (2)

- Expands notion of security level to include categories (types of information)
- Security level is
    - (*clearance/classification*, category used)
- Examples
    - (Top Secret, { NUC, EUR, US})
    - (Confidential, {EUR, US})
    - (Secret, {NUC, US})

Dominance relationship

- Captures both security level and category information
- (L C) dom (L', C') iff L' <= L and C' $\subseteq$ C
- Examples
    - (Top Secret, {NUC, US}) **dom** (Secret, {NUC})
    - (Secret, {NUC, EUR}) **~dom** (Confidential, {US, EUR})
    - (Top Secret, {NUC}) **~dom** (Confidential, {EUR})
    - (Secret, {EUR}) **~dom** (Top Secret, {})

Reading Information

- "Reads up" disallowed, "reads down" allowed
- Simple Security Condition (2)
    - Subject s can read object o iff L(s) dom L(o) and s has discretionary read access to o
- You can only read if you dominate

Writing

- "Writes up" allowed, "writes down" disallowed
- *-Property (2)
    - Subject s can write object o iff L(o) dom L(s) and s has discretionary write access to o
- You can only write if what you are writing to dominates you

Basic Security Theorem

- If a system is initially in a secure state, and every transition of the system satisfies the simple security condition (2), and the *-property (2), then every state of the system is secure

Problem

- Colonel has (Top Secret, {NUC, EUR}) clearance
- Major has (Secret, {EUR}) clearance
- Colonel cannot send message to Major
    - Violates no write down

Solution

- Define maximum, current security levels for (trusted) subjects
    - *Maxlevel*(s) dom *curlevel*(s)
        - Colonel has *maxlevel* (Top Secret, {NUC, EUR})
        - Colonel sets *curlevel* to (Secret, {EUR})
- Example
    - Treat major (Secret, {EUR} clearance) as an object (Colonel is writing to him/her)
    - Now L(Major) dom *curlevel* (Colonel)
        - Colonel can write to major without violating "no writes down"

*BLP does not protect integrity

Biba Integrity Model

- Developed by Kenneth J. Biba in 1977
- First attempt to model integrity constraints
- The goal of this model is to prevent unauthorized change (**corruption**) of information
- Based on multilevel integrity
- The higher the integrity level, there is more confidence
    - That data is accurate and or/reliable
    - That a program will execute correctly
- **Integrity* levels are NOT confidentiality levels
- Dual of BLP model
    - Combines both MAC and DAC
- It is possible for a document to have a high confidentiality and low integrity because they are two different things

Biba's Model

- **Information flows DOWN**
- Allow someone at a higher level to write to something at a lower level because you trust them more
- Simple integrity property
    - $s \in S$ can read $o \in O$ iff $i(s) <= i(o)$
        - No read down
        - Subjects can view content at or above their own integrity level
- * Integrity property
    - $s \in S$ can write to $o \in O$ iff $i(o) <= i(s)$
        - No write up
        - Subjects can create content at or below their own integrity level

Comparison of BLP and Biba

- **BLP**:
    - Focuses on confidentiality only
    - Information flows up
    - Applies to military/government sector mostly
- **Biba**:
    - Focuses on integrity only
    - Information flows down
    - Applies to commercial sector mostly
- **Both**:
    - Combines MAC and DAC
    - Intended for static policy
    - Do not address management of access control
    - Can be implemented at the same time

Bell-LaPadula Model Examples

1. Paul, cleared for (TOP SECRET), wants to access a document classified (SECRET)
    - **Has read but not write privileges.**
2. Anna, cleared for (CONFIDENTIAL), wants to access a document classified (CONFIDENTIAL)
    - **Has read and write privileges.**
3. Jesse, cleared for (SECRET), wants to access a document classified (CONFIDENTIAL)
    - **Has read but not write privileges.**
4. Sammi, cleared for (TOP SECRET), wants to access a document classified (CONFIDENTIAL)
    - **Has read but not write privileges.**
5. Robin, who has no clearances (and so works at the UNCLASSIFIED level), wants to access a document classified (CONFIDENTIAL)
    - **Has write but not read privileges.**

1. Paul, cleared for (TOP SECRET, {A, C}), wants to access a document classified (SECRET, {B, C})
    - **Has no privileges.**
2. Anna, cleared for (CONFIDENTIAL, {C}), wants to access a document classified (CONFIDENTIAL, {B})
    - **Has no privileges.**
3. Jesse, cleared for (SECRET, {C}), wants to access a document classified (CONFIDENTIAL, {C})
    - **Has read but not write privileges.**
4. Sammi, cleared for (TOP SECRET, {A, C}), wants to access a document classified (CONFIDENTIAL, {A})
    - **Has read but not write privileges.**
5. Robin, who has no clearances (and so works at the UNCLASSIFIED level), wants to access a document classified (CONFIDENTIAL, {B})
    - **Has write but not read privileges**

Biba Model Examples:

1. Paul, cleared for (VERY HIGH), wants to access a document classified (HIGH)
    - **Has write but not read privileges.**
2. Anna, cleared for (HIGH), wants to access a document classified (HIGH)
    - **Has read and write privileges.**
3. Jesse, cleared for (MEDIUM), wants to access a document classified (HIGH)
    - **Has read but not write privileges.**

February 2nd – Security Models CW, CISS, CLW

Comparison of BLP and Biba (Continued from last lecture)

- How Biba covers both MAC and DAC
    - DAC: object owner gives discretionary access
    - MAC: the rule is decided by a higher authority

Chinese Wall Model

- Problem:
    - Tony advises American Bank about investments
    - He is asked to advise Toyland Bank about investments
        - **Conflict of interest**
- Brewer and Nash (1989)
- Derived from British Law
- **Hybrid Model** – commercial policy; applied in commercial sectors
- Focus on confidentiality and integrity
- Characteristics
    - Conflict of interest
    - Dynamic change of access rights

Principle

- Combines commercial discretion with enforceable mandatory controls
- Data sanitized/unsanitized
    - Allows sanitized data to be viewed by everyone
    - For unsanitized data, access is controlled
- Dynamically establishes access rights of a user based on access history

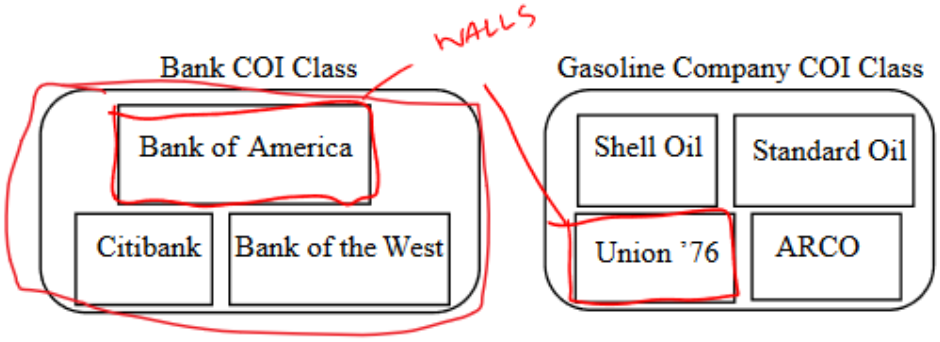
Sanitization

- Public information may belong to a CD (Company Dataset)
    - As is publicly available, no conflicts of interest arise so is should not affect the ability of analysts to read
    - Typically, all sensitive data is removed from such information before it is released publicly (called ***sanitization***)

Definition

- Three levels of abstraction:
    1. Objects: items of information related to a company
    2. Company dataset (CD): contains objects related to a single company
    3. Conflict of interest class (COI): contains datasets of companies in competition
        - Assume: each object belongs to exactly one *COI* class

Example



- Protects information within the walls – the boundaries protect what information can go outside the dataset
- Conflict of interest: What can people read/write?
- What should John be able to read?
    - If he has permission to read from Bank of America, then he does not have permission for Citibank, but he does have permission to read Union
    - If he has not read from anything yet, then he can read from any one.
    - He can go back and read from Bank of America after having permission to read once

- Rules:
    1. Sanitized information is public
    2. If they have not read then they have access to (one of) anything
    3. What they are trying to read should not belong to the same interest class

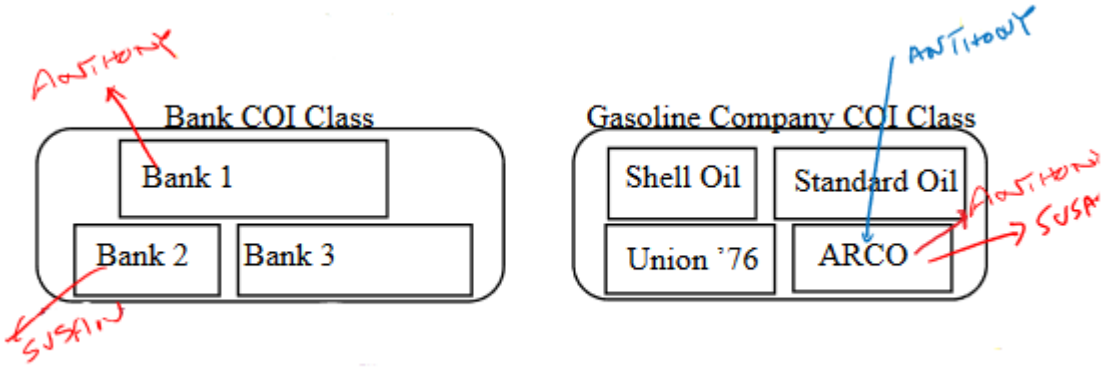
Temporal element

- What you have done in the past effects what you can do in the future
- If Anthony reads any CD in a COI, he can *never* read another CD in that COI
    - It is possible that information learned earlier may allow him to make decisions later

CW-Simple Security Condition

- Let PR(S) be a set of objects S has already read
    - Initially, PR(S) = ∅, so initial read request is granted
- s can read o iff either condition holds:
    1. There is an o' such that s has accessed o' and CD(o') = CD(o)
        - Meaning s has already read something in o's dataset
        - Already permitted
    2. For all o' ∈ O, o' ∈ PR(s) ⇒ COI(o′) ≠ COI(o)
        - Meaning anything s has read before does not belong to o's conflict of interest class
        - Has no knowledge of competitors
    3. o is a sanitized object

Writing



- Anthony, Susan work in same trading house
- Anthony can read Bank 1's CD, Arco Gas' CD
- Susan can read Bank 2's CD, Arco Gas' CD
- If Anthony could write to Arco Gas' CD, Susan can read it
    - Hence, indirectly, she can read information from Bank 1's CD, a clear conflict on interest

CW -*- Property

- s can write to o iff <u>both</u> of the following hold:
    1. the CW-simple security condition permits s to read o; and
        - <u>Has read permission</u>
    2. For all unsanitized objects o', if s can read o', then CD(o') = CD(o)
        - Only 'secret' knowledge is from this one
- Says that s can write to an object if all the (unsanitized) objects it can read are in the same dataset

Implications

- Flow of sanitized data is unrestricted
- Flow of unsanitized information confined to its own CD (within the wall)
- A subject is allowed access to only one CD in each OI
- Past history of access affects future accesses

Compare to Bell-LaPadula (BLP)

- Fundamentally different
    - CW has no security labels, BLP does
    - CW has notion of past accesses, BLP does not
- BLP cannot track changes over time
    - Susan becomes ill, Anna needs to take over
        - CW history lets Anna know if she can
- Freedom at first but access constraints change over time
    - Initially, subjects in CW can read any object
    - BLP constrains set of objects that a subject can access
- BLP can give a subject full access/privilege
    - CW cannot grant full access/privilege
        - Only one CD access from a COI

Criticism of Chinese Wall

- Some assumptions maybe realistic and/or flawed
    - Company datasets in different COIs might actually be in competition
    - Company datasets in the same COI might NOT actually be in competition
- Therefore, define COI classes **NOT** based on businesses classes **BUT** based on common interests

In class exercise:

In reference to Chinese Wall model and the following diagram, what can Amanda **READ from and WRITE to** in each of the following cases:

Wireless Phone
Verizon
T-Mobile
AT&T
Sprint

Beverage
Pepsi
Coca Cola
Dr. Pepper
Canada Dry

Tech Comp
Microsoft
Apple
Mozilla
Yahoo

a. When she has not started to work for any of these companies?
    - She can read any **ONE** company in Wireless Phone, Beverage, **OR** Tech Comp
    - She can write to **NONE** yet without read access
b. Once she gets read access to Pepsi
    - She can read **ONE** of either Wireless Phone **OR** Tech Comp
    - She can write to Pepsi
c. Once she gets read access to Pepsi and Apple
    - She can read any **ONE** Wireless
    - She **CANNOT** write because she might transfer information between the two
d. When she wants to read Coca Cola's public portfolio?
    - She can read it because it is public.

February 14th – Rest of Security Models CW, CISS, CLW

*Quiz on Tuesday, February 19th over all 5 security models

- What is the main characteristic/focus?
- How are they different?
- Does it apply to a certain sector?
- What is the specific application?
- How do they read/write?
- Revisit in-class work and pop-quizzes

*Assignment 5 due on February 19th

Clinical Information Systems Security (CISS) Policy

- Anderson (1996)
- Prototypical HIPPA – Health Information Privacy Protection Act
- Hybrid model intended for medical records
    - Focus is Patient privacy and record integrity
- Entities:
    - **Patient**: subject of medical records (or agent)
    - **Personal Health Information**: Data about patient's health or treatment enabling identification of patient
    - **Clinician**: health-care professional (or group of professionals) with access to personal health information while in service

Assumptions and Principles

- Assumes health information involves 1 person at a time
    - Not always true
    - For babies it might include their parents
- Principles derived from medical ethics and practices
- Principles
    - Access
    - Audit

- Creation
- Deletion
- Confinement
- Aggregation
- Enforcement

Access Principle 1 – **Access Control List**

- Each medical record has an access control list (ACL) naming the individuals or groups who may read, update, and append information to the record. The system must restrict access to those identified on the access control list
  - (Notice delete is not mentioned above)
  - Idea is that clinicians need access, but no-one else. Auditors get access to copies, so they cannot alter records

Access Principle 2 – **Responsible Clinician**

- One of the clinicians on the ACL must have the right to add other clinicians to the ACL.

Access Principle 3 – **Notification**

- The responsible clinician must notify the patient of the names on the ACL whenever the patients medical record is opened
  - Except for situations given in statutes, or in cases of emergency, the responsible clinician must obtain the patients consent (for treatment)
  - Patients must know of any violations of security

Access Principle 4 – **Audit**

- The name of the clinician, the date, and the time of the access of a medical record must be recorded
  - *Don't delete* information; *update* it
  - Similar information must be kept for *deletions*

Creation Principle

- A clinician may open/create a (new) record, with the clinician and the patient on the ACL. If a record is opened (created) as a result of a referral, the referring clinician may also be on the ACL.

Deletion Principle

- Clinical information cannot be deleted from a medical record
    - Unless patient *died*
    - Until *appropriate time* has passed
        - Typically 8 years.

Confinement Principle

- Information from one medical record (A) may be appended to a different medical record (B) if and only if the access control list of the second record (B) is a subset of the ACL of the first (A)
    - This keeps information from leaking to unauthorized users. All users have to be on the access control list
- Ex: A way for the general physician to get information from your heart doctor if and only if the general physician is in the ACL of the heart doctor's medical record

Aggregation Principle

- Measures for preventing aggregation of patient data must be effective. In particular, a patient must be notified if anyone is to be added to the ACL for the patient's record and if that person has access to a *large number* of medical records
    - The fear here is that a corrupt investigator may obtain access to a large number of records, correlate them, and discover private information about individuals which can then be used against the patient's interest.

Enforcement Principle

- Any computer system that handles medical records must have a subsystem that enforces the preceding principles

- The effectiveness of this enforcement must be subject to evaluation by independent auditors
    - This policy has to be enforced, and the enforcement mechanisms must be auditable (and audited)

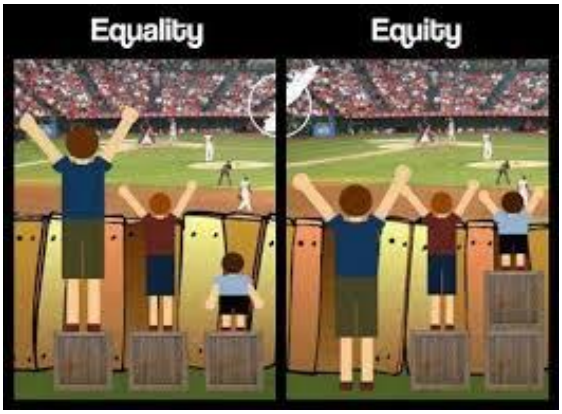
Compare to Bell-LaPaluda

- CISS focuses on objects being accessed; B-LP on the subjects accessing the objects (capabilities)

Points to Ponder

- Who has access to records?
    - Doctor, patient, auditor
- Who can create a record?
    - Clinician
- Who can append to a record?
    - Clinician/responsible clinician
- Who can delete a record?
    - Doctor's office
    - If the patient dies or the appropriate time has passed
- Under which circumstance should the patient be notified?
    - Someone has access to a lot of data, every time a doctor is added, every time data is opened, or if there is a breach
- What needs to be audited?
    - Home, date, time

Equality vs Equity



- **Equality** – Everyone is given the exact same benefits
- **Equity** – Everyone is given enough benefits to level the playing field

Clark-Wilson Model

- An integrity model better suited for the commercial world
- Ensures the internal data is accurate and consistent to what it represents in the real world
    - Internal consistency and external reality
- Integrity
    - Focuses on users and their actions on the data
        - Rather than the data itself
    - Defined by a set of constraints
        - Data in a *consistent* or valid state when it satisfies the constraints
        - Example: Bank
            - D todays deposits, W withdrawls, YB yesterday's balance, TB today's balance
            - Integrity constraints: TB = D + YB – W
- Characteristics
    - Focus on: Data < Transactions > Users
        - Certain data items are constrained in that only certain processes may manipulate them
        - Users are constrained in how they manipulate data

Clark-Wilson Model Pillars

- Well formed transactions
    - Transitions system from one valid state to another
    - Must preserve consistency
- Separation of duty
    - Combinations of operations performed by different people
        - Certification and implementation/enforcement

Concepts

- Data
    - **CDIs**: Constrained Data Items
        - Data subject to integrity controls
    - **UDIs**: Unconstrained Data Items (compare to sanitized and bank portfolio)
        - Data *not* subject to integrity controls
- Procedures
    - **IVPs**: Integrity Verification Procedures
        - Procedures that *test* the CDIs conform to the integrity constraints
    - **TPs**: Transformation Procedures
        - Procedures that *take* the system from one valid state to another

*What is the difference between IVP and TP?

Policy Rules for Integrity Assurance

- Defined by 5 certification and 4 enforcement rules
- Rules:
    - Certification Rules (CR)
        - security administration/system custodian
        - Monitors integrity
    - Enforcement Rules (ER)
        - System itself
        - Preserves integrity

Certification Rules 1 & 2

- **CR1** – When the IVP is run, it must ensure all CDIs are in a valid state
- **CR2** – For some associated sat of CDIs, a TP must transform those CDIs in a valid state into a (possibly different) valid state
    - Defines *certified relation* that associates a set of CDIs with a particular TP

Enforcement Rules 1 & 2

- **ER1** – The system must maintain the certified relations and must ensure that only TPs certified to run on a CDI manipulate that CDI
    - System must maintain, enforce certified relation
- **ER2** – The system must associate a user with each TP and set of CDIs. The TP may access those CDIs on behalf of the associated user. The TP cannot access a CDI on behalf of a user who is not associated with TP and CDI
    - System must enforce access based on user ID (*allowed relation*)

Users & Rules

- **CR3** – The allowed relations must meet the requirements imposed by the principle of separation of duty
    - If the relation is not correctly implemented, then everything falls apart
- **ER3** – The system must authenticate each user attempting to execute a TP
    - Type of authentication undefined, and depends on the instantiation
    - Authentication *not* required before the use of the system, but *is* required before manipulation of CDIs (Complete mediation)

Logging

- **CR4** – All TPs must append enough information to reconstruct the operation to an append-only CDI
    - The CDI is the log
    - When you identify something as a CDI, you must also know the allowed relation(who can perform the actions on the log?) and the certified relation (which actions can be done on this log?)

Handling Untrusted Input

- **CR5** – Any TP that takes as input a UDI may perform only valid transformation, or no transformations, for all possible values pf the UDI. The transformation either rejects the UDI or transforms it into a CDI.

Separation of Duty in Model

- **ER4** – Only the certifier of the TP may change the list of entities associated with that TP. No certifier of a TP, or of an entity associated with that TP, may ever have execute permission with respect to that entity
    - Enforces separation of duty with respect to *certified* and *allowed relations*

A Flawed Implementation

- Rule ER2 sets up something similar to an access control list – a triple

  user_id, TPi, (CDIa, CDIb, …)
- An implementation of this policy used two relations/tables

  [user_id, TPi] & [TPi, (CDI's)]
- Consider a system with
    - 2 users (s1, s2)
    - 2 TP's (TP1, TP2)
    - 3 CDI (CDI1, CDI2, CDI3)
- Policy is –
    - S1, TP1, CDI1
    - S1, TP2, CDI2
    - S2, TP1, CDI2
    - S2, TP2, CDI3
- Resulting Bindings
    - User-Program bindings

      (S1, TP1)

      (S1, TP2)

      (S2, TP1)

      (S2, TP2)
    - Program-Data bindings

      (TP1, CDI1)

      (TP1, CDI2)

      (TP2, CDI2)

      (TP2, CDI3)

Comparison to Biba

- Biba based on *multilevel data integrity*
- Clark-Wilson focuses on transaction integrity
- Biba
    - Subjects are given access to objects *directly*
    - *No* notion of *certification rules*; trusted subjects assume actions obey rules
    - Untrusted data made trusted by trusted subjects
- Clark-Wilson
    - Subjects are given access to objects *through programs*
    - Explicit certification *requirements*
    - Trusted entity *method* to upgrade untrusted data

February 19th—Quiz 2 & Cryptography

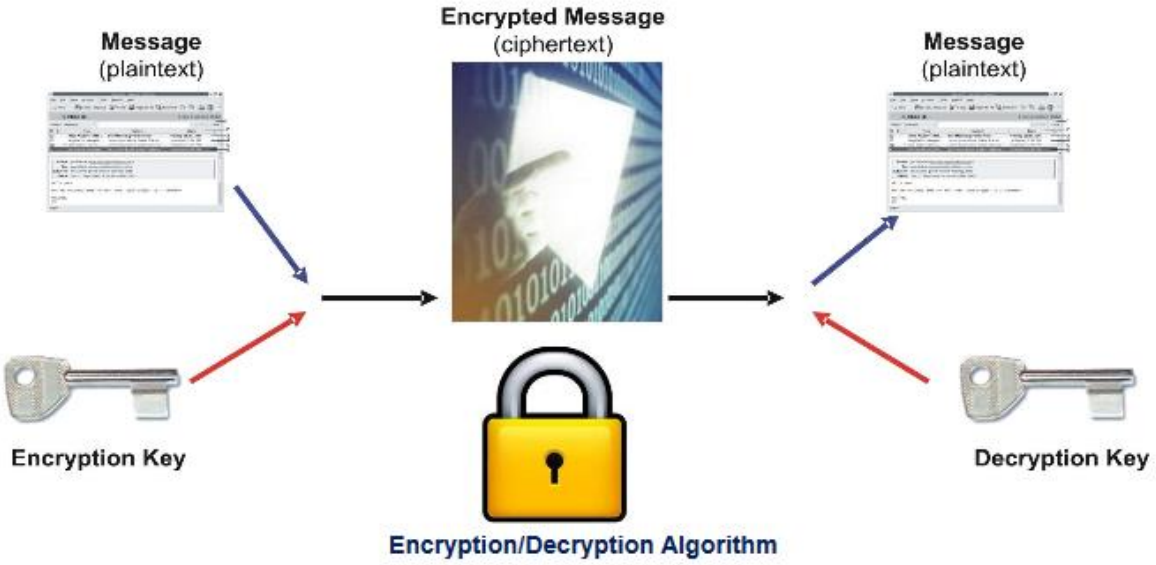
Cryptography

- Art of secret writing
- Concealing apparent meaning
- Primarily deals with *confidentiality*
    - Important integrity aspects
- Goal:
    1. Should not be understood by unauthorized third party
    2. Should prove *correctness*
    3. Should prove *origin*
    4. Should prove *accountability*
        - (does not prove availability)

Early History

- Non-standard hieroglyphics
    - Egypt 2500 BC
- Atbash Cipher
    - 500 – 600 BC
- Caesar Cipher
    - Plaintext is *HELLO WORLD*
    - Change each letter to another letter following a rule
        - X goes to A, Y to B, Z to C
    - Ciphertext is *KHOOR ZRUOG*

Overview



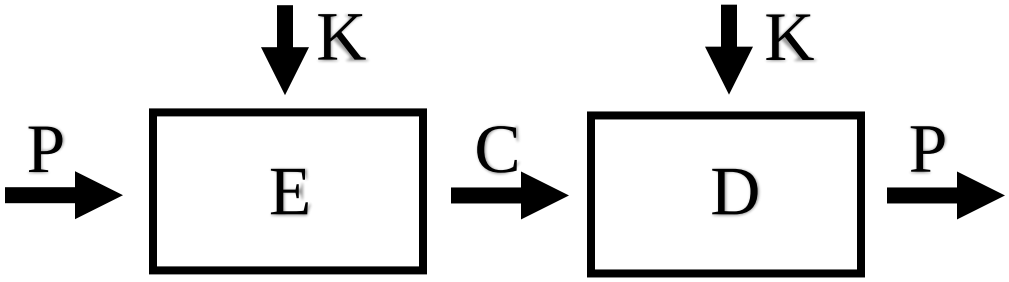
- The "magic" lies in the algorithm

Terminology

- **Plain text**: Original text
- **Cipher text**: Coded text
- **Crypto algorithm**: Function to change plain text to cipher text
- **Key/cryptovariable**: Used in coding/decoding text
- **Encryption**: Collective term for encoding and enciphering
    - **Encoding**: Coding entire words/phrases
    - **Enciphering**: Coding each letter/symbol individually
- **Decoding/Deciphering/Decryption**: Reverse
- **Cryptosystem**: System for encryption and decryption
- **Cryptography**: Use of encryption/decryption
    - Works for sender and receiver
- **Cryptology**: Study of cryptosystem
- **Cryptanalysis**: Attempts to break a cryptosystem
    - Works for interceptor

Cryptosystem

- Quintuple (E, D, P, K, C)
    - **P** – set of *plaintexts*
    - **C** – set of *ciphertexts*
    - **E** – set of *encryption functions* e: P x K → C
    - **D –** set of *decryption functions* d: C x K → P
    - **K** – set of *keys*



Cryptanalysis

- Assume adversary knows algorithm used, but not key
- Four basic types of attacks:
    1. Ciphertext only – adversary has only *ciphertext*; goal is to find *plaintext*, possibly key
    2. Known plaintext – adversary has *ciphertext* with corresponding *plaintext*; goal is to find *key*
    3. Chosen plaintext – adversary may supply specific plaintexts of interest and obtain corresponding *ciphertext*; goal is to find *key*
        - Adversary has access to cryptosystem on sender's (encryption) side
        - The attacker has more flexibility/freedom to look at words they want to study in the cryptosystem
    4. Chosen ciphertext – Adversary may supply specific ciphertexts of interest and obtain corresponding plaintext; goal is to find the *key*
        - Adversary have access to cryptosystem on receiver's (decryption) side

Basis for Attacks

- Brute Force
- Mathematical Attacks
    - based on analysis of underlying mathematics
- Language attacks
    - *Adhoc* – use experience
        - Based on guess, no solid principle
        - *Wklv phvvdjh lv qrw wrr kdug wr euhdn*
            - *This message is not too hard to break*
    - Statistical analysis
        - Examine ciphertext, correlate properties with language
        - Using models of the language (*frequency* and *distribution* of letters) make assumptions
            - Pairs of letters (*digrams*), triplets of letters (*trigrams*), word boundaries, etc.
            - Most common letters: ETANOS

Types of encryption/decryption

- Based on:
    - How keys are shared/distributed
    - Key size
    - How plain text is transformed into cipher text (operation)
    - How plain/cipher text is processed

Classical/Symmetric Cryptography

- Sender, receiver share common key
    - Keys may be the same
    - Trivial to derive from one another

Based on operation

- Two basic types:
    - Transposition ciphers
    - Substitution ciphers
        - Combinations are called *product ciphers*

Transposition Ciphers

- *Rearrange* letters in plaintext to produce ciphertext (the letters themselves are not changed)
- Goal is *diffusion*
- Slower
- Example: (Rail-Fence/Columnar Cipher)
    - Plaintext is *HELLO WORLD*
    - Rearrange as:

      HLLOOL

      ELWRD
    - Ciphertext is *HLOOL ELWRD*
- Example 2:
    - Plaintext is *CATS ARE CUTE*

      CSET

      AACE

      TRU
    - Ciphertext is *CSET AACE TRU*

Attacking the Cipher

- Most common English Digrams (2-grams)
    - TH, HE, IN, EN, NT
- Most common English Trigrams (3-grams)
    - THE, AND, THA, ENT, ING
- Assumption: If 1-gram frequencies match English frequencies, but other n-gram frequencies do not, probably transposition

- Anagramming Technique for Cryptanalysis
    - Correlating observed pattern with known properties and rearranging letters to form n-grams with highest frequencies

Example

- Ciphertext: *HLOOLELWRD*
- Get frequencies of English letters
    - Example: http://www.cryptograms.org/letter-frequencies.php
- Find frequencies of 2-grams beginning with H
    - HE = 0.0305
    - HO = 0.0043
    - HL, HW, HR, HD < 0.0010
- Frequencies of 2-grams ending in H
    - WH = 0.0026
    - EH, LH, OH, RH, DH <= 0.0002
- Implies E follows H
- Arrange them so the H and E are adjacent

    HE

    LL

    OW

    OR

    LD
- Read off across/down, then down/across to get the original plaintext

Substitution Ciphers

- *Change/replace* characters in plaintext to produce ciphertext
- Goal is *confusion*
- Faster
- Example (Caesar cipher)

Formal Representation of Caesar

- Caesar cipher for English Alphabets
    - P = { sequences of letters }
    - K = { i | i is an integer and 0<= i <= 25 }
    - E = { Ek | k E K and for all letters p, Ek(p) = (p + k) mod 26 }
        - A = 1 + 2 = C
    - D = { Dk | k E K and for all letters p, Ek(p) = (26 + c - k) mod 26 }
        - C = 3 – 2 = A

Attacking the Cipher

- **Exhaustive Search**
    - If the key space is small enough, try all possible keys until you find the right one
    - Caesar cipher has **26** possible keys
- **Statistical analysis**
    - Compare to model of English

February 21st—Crypto Continued

COLLECT MOMENTS NOT THINGS

JVSSLJA TVTLUAZ UVA AOPUNZ

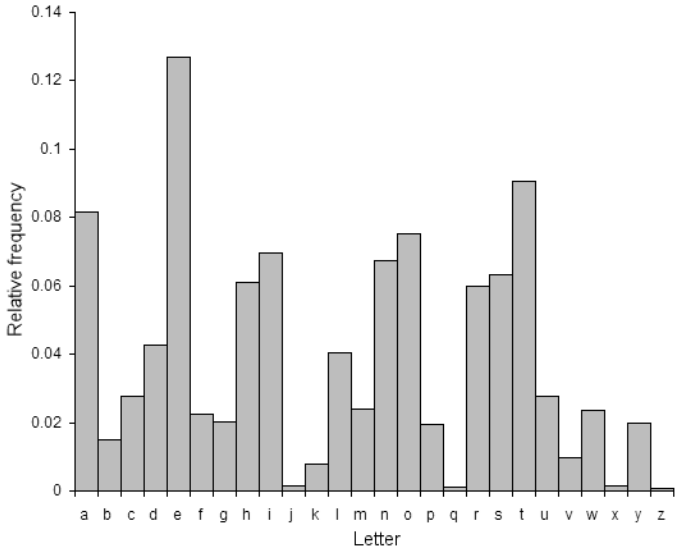
Statistical Attack

- Compute frequency of each letter in ciphertext (KHOOR ZRUOG):

  G: 0.1 H: 0.1 K: 0.1 O: 0.3

  R: 0.2 U: 0.1 Z: 0.1

- Apply model of English Language



f(x) Character Frequencies (Chart XX)

a/0	0.080	h/7	0.060	n/13	0.070	t/19	0.090
b/1	0.015	i/8	0.065	o/14	0.080	u/20	0.030
c/2	0.030	j/9	0.005	p/15	0.020	V/21	0.010
d/3	0.040	k/10	0.005	q/16	0.002	w/22	0.015
e/4	0.130	l/11	0.035	r/17	0.065	x/23	0.005
f/5	0.020	m/12	0.030	s/18	0.060	y/24	0.020
g/6	0.015					z/25	0.002

Statistical Analysis

- φ(i) **Correlation of frequency** of letters in ciphertext with corresponding letters in English, assuming key is *i*
    - $\varphi(i) = \Sigma 0 \leq c \leq 25\ f(c)f'(e - i)$
    - $\varphi(i) = 0.1f'(6 - i) + 0.1f'(7 - i) + 0.1f'(10 - i) + 0.3f'(14 - i) + 0.2f'(17 - i) + 0.1f'(20 - i) + 0.1f'(25 - i)$
        - f(c) frequency of character c in ciphertext
        - f'(x) is frequency of character with index = x in English

      Ciphertext is `HKOOR ZRUOG`
- The Result:
    - Most probable keys, based on φ:
        - i = 6, φ(i) = 0.0660
            - Plaintext is `EBITL TLOLA`
        - i = 10, φ(i) = 0.0635
            - Plaintext is `AXEEH PHKEW`
        - i = 3, φ(i) = 0.0575
            - Plaintext is **`HELLO WORLD`**
        - i = 14, φ(i) = 0.0535
            - Plaintext is `WTAAD LDGAS`
    - Only English phrase found is for i = 3
        - Key = 3 (or 'D')

Caesar's Problem

- Key is too short (**Mono-alphabetic**)
    - Only *26 possibilities*
    - Can be found by exhaustive search
    - Statistical frequencies *easily* derived/found
    - Less confusion with 1-to-1 mapping
- So make it longer (**Poly-alphabetic**)
    - *26^n possibilities*

- Multiple letters in a key
- Idea is to smooth the statistical frequencies to make *cryptanalysis harder*
- More confusion with *1-to-many* mapping

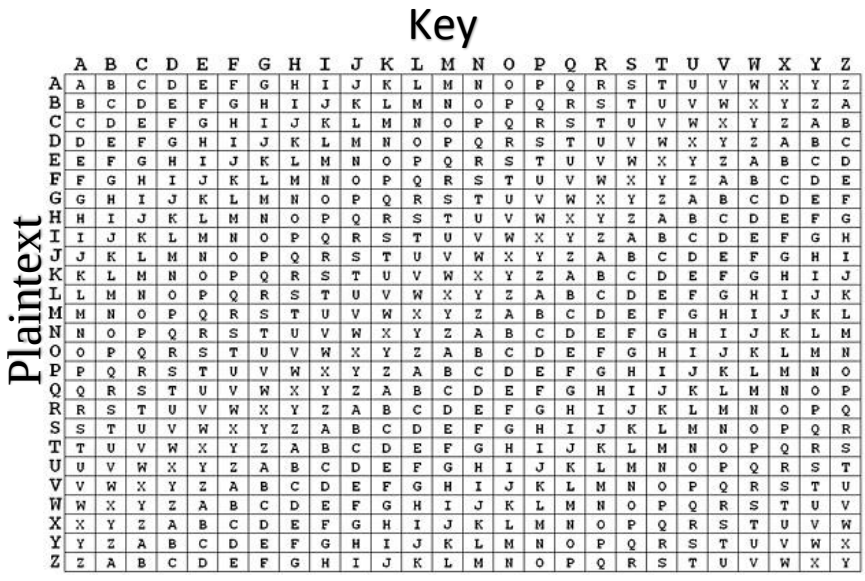
Vigenere Cipher

- Substitution like Caesar cipher, but polyalphabetic key
- Use a *word/phrase*, *repeatedly*
- Example:
  - Message: `THE BOY HAS THE BALL`
  - Key `VIG`
  - Encipher using Caesar Cipher for each letter:

Key:	`VIGVIGVIGVIGVIGV`
Plain:	`THEBOYHASTHEBALL`
Cipher:	`OPKWWECIYOPKWIRG`

Useful Terms

- **Period**: length of key
  - In the earlier example, the period is 3
- **Tableau**: table used to encipher and decipher
  - Vigenere cipher has key letters on top, plaintext letters on the left



Attacking the Cipher

- Cryptanalysis Approach
    - Find out period/key or n
    - Break message into n different parts
        - Each part being enciphered using the same key letter
    - Solve each part individually (each becomes Caesar Cipher)

Key: VIGVIGVIGVIGVIGV

Plain: THEBOYHASTHEBALL

Cipher: OPKWWECIYOPKWIRG

Finding the Period

- *Kaskski's Observation*:
    - Repetitions in the ciphertext occur when characters of the key appear over the same characters in the plaintext
    - Period is a *factor of distance* between such repetitions
- Example:

Key: **VIG**VIGVIG**VIG**VIGV

Plain: **THE**BOYHAS**THE**BALL

Cipher: **OPK**WWECIY**OPK**WIRG

    - Note that key and plaintext line up over the repetitions. As distance between repetitions is 9, the period is a factor of 9 (that is 1, 3, or 9).

The Target Cipher

- We want to break this cipher:

```
ADQYS MIUSB OXKKT MIBHK IZOOO
EQOOG IFBAG KAUMF VVTAA CIDTW
MOCIO EQOOG BMBFV ZGGWP CIEKQ
HSNEW VECNE DLAAV RWKXS VNSVP
HCEUT QOIOF MEGJS WTPCH AJMOC
HIUIX
```

Letters	Start	End	Distance	Factors
MI	5	15	10	2, 5
OO	22	27	5	5
OEQOOG	24	54	30	2, 3, 5
FV	39	63	24	2, 2, 2, 3
AA	43	87	44	2, 2, 11
MOC	50	122	72	2, 2, 2, 3, 3
QO	56	105	49	7, 7
PC	69	117	48	2, 2, 2, 2, 3
NE	77	83	6	2, 3
SV	94	97	3	3
CH	118	124	6	2, 3

Estimate of Period

- OEQOOG is probably not a coincidence
    - It's too long for that
- Most others (7/10) have 2 in their factors
- Almost as many (6/10) have 3 in their factors
- Begin with period of 2 x 3 = 6
    - ***Greatest Common Divisor*** of two of the largest repetitions (OEQOOG and MOC)

Validating the period

- **Index of coincidence** (IC) is the probability that two randomly chosen letters from ciphertext will be the same
    - Greater value of IC means greater probability, hence smaller key
- IC = $[n (n – 1)]^{-1} \Sigma_{0 \le i \le 2} [F_i(F_i – 1)]$
    - Where n is the length of the ciphertext of and F_i the number of times character *i* occurs in ciphertext

- Calculate Index of coincidence for KHOOR ZROUG
  - Number of times each letter repeats:

    K = 1 H = 1 O = 3 R = 2

    Z = 1 G = 1 U = 1
  - IC = [10 (10 – 1)]^-1*[1(1– 1) + 1(1– 1) + 3(3– 1) + 2(2– 1) + 1(1– 1) + 1(1– 1)]
    - All letters that only occur once (highlighted above) become 0…
  - IC = [10 (10 – 1)]^-1*[3(3– 1) + 2(2– 1)]
  - IC = [10 (9)]^-1*[3(2) + 2(1)]
  - IC = [90]^-1*[6 + 2]
  - IC = [90]^-1*[8] = **8 / 90**

Compute IC

- Tabulated results publicly available for IC in English for different periods:

1	0.066	**5**	0.055
2	0.052	**10**	0.041
3	0.047	**Large**	0.038
4	0.045		

- In the example, IC for the whole text = 0.043
  - Indicates a key of slightly more than 5

Splitting Into Alphabets

Alphabet 1:	AIKHOIATTOBGEEERNEOSAI	#1, 0.069
Alphabet 2:	DUKKEFUAWEMGKWDWSUFWJU	#2, 0.078
Alphabet 3:	QSTIQBMAMQBWQVLKVTMTMI	#3, 0.078
Alphabet 4:	YBMZOAFCOOFPHEAXPQEPOX	#4, 0.056
Alphabet 5:	SOIOOGVICOVCSVASHOGCC	#5, 0.124
Alphabet 6:	MXBOGKVDIGZINNVVCIJHH	#6, 0.043

Frequency Examination

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	
1	**3**	1	0	0	4	0	1	1	3	0	1	0	0	1	3	0	0	1	1	2	0	0	0	0	0	0	**A**
2	1	0	0	2	2	2	1	0	0	1	3	0	1	0	0	0	0	0	**1**	0	4	0	4	0	0	0	**S**
3	1	2	0	0	0	0	0	0	**2**	0	1	1	4	0	0	0	4	0	1	3	0	2	1	0	0	0	**I**
4	2	1	1	0	2	2	0	1	0	0	0	0	1	0	4	3	1	0	0	0	0	0	0	**2**	1	1	**X**
5	1	0	4	0	0	0	2	1	2	0	0	0	0	0	**5**	0	0	0	3	0	0	3	0	0	0	0	**O**
6	0	1	1	1	0	0	2	2	3	1	1	0	1	2	1	0	0	0	0	0	0	**3**	0	1	0	1	**V**

Begin Decryption

- First matches characteristics of unshifted alphabet
  - 1st Key = A
- Third matches if I shifted to A
  - 2nd Key = I
- Sixth matches if V shifted to A
  - 6th Key = V

Using Vigenere Cipher:

Alphabet 1:	AIKHOIATTOBGEEERNEOSAI	#1, 0.069
Plaintext 1:	AIKHOIATTOBGEEERNEOSAI	Key = A
Alphabet 2:	DUKKEFUAWEMGKWDWSUFWJU	#2, 0.078
Plaintext 2:	LCSSMNCIEMUOSELEACNERC	Key = S
Alphabet 3:	QSTIQBMAMQBWQVLKVTMTMI	#3, 0.078
Plaintext 3:	IKLAITESEITOINDCNLELEA	Key = I
Alphabet 4:	YBMZOAFCOOFPHEAXPQEPOX	#4, 0.056
Plaintext 4:	MPANCOTQCCTDVSOLDESDCL	Key = M
Alphabet 5:	SOIOOGVICOVCSVASHOGCC	#5, 0.124
Plaintext 5:	EAUAASHUOAHOEHMETASOO	Key = O
Alphabet 6:	MXBOGKVDIGZINNVVCIJHH	#6, 0.043
Plaintext 6:	RCGTLPAINLENSSAAHNOMM	Key = V

- Read each Plaintext one letter at a time, travelling from top to bottom.

- Once you make it all the way through the list, the plaintext is:

  ALIME RICKP ACKSL AUGHS ANATO MICAL INTOS PACET
  HATIS QUITE ECONO MICAL BUTTH EGOOD ONESI VESEE
  NSOSE LDOMA RECLE ANAND THECL EANON ESSOS ELDOM
  ARECO MICAL

  Key: **ASIMOV**

- Solution:

  A LIMERICK PACKS LAUGHS ANATOMICAL INTO SPACE THAT IS
  QUITE ECONOMICAL BUT THE GOOD ONES IVE SEEN SO SELDOM
  ARE CLEAN AND THE CLEAN ONES SO SELDOM ARE COMICAL

One-Time Pad

- A Vigenere cipher with
  - Random key
  - At least as long as the message
  - Used only once
- Provides perfect secrecy IF the key is
  - Truly random
  - Never reused
  - Kept secret
- However
  - Not used very often because of *key distribution problems*

Good Cipher Characteristics

- Shannon 1949
  - Need for protection should determine cost/effort
  - None or *minimal complexity* for usage
  - Simple/reasonable implementation
  - Errors should not propagate
    - One corruption should not affect later messages
  - Cipher text should not be longer than the original text

- More…
    - Confusion such that by changing P attacker cannot predict whole C
    - Diffusion such that any changes in P are spread out to much of C
    - Based on sound principles
    - Peer/expert reviewed
    - Withstand “test of time”

February 26th – Cryptography Continued

DES (Data Encryption Standard)

- FIPS 46-3 (Federal Information Processing Standard)
- Most important classical cryptosystem
- History
    - 1073/1974 call for proposal by NBS for a public encryption algorithm that could protect unclassified but sensitive information
        - High level of security/easy to understand/publishable/available/adaptable/ economical/efficient…
    - Data Encryption Algorithm (DEA) designed by IBM, based on Horst **Feistel**'s Lucifer cipher, reviewed by NSA
    - Adopted by NBS/NIST in 1976
        - International standard for use in public and private sector

Overview of DES

- Symmetric cryptography algorithm
    - *Same algorithm* for encryption/decryption
    - *Same key* for encryption/decryption
- A block cipher
    - *Bit-oriented*
    - Encrypts blocks of 64 bits using a 64-bit key
    - Outputs 64 bits of ciphertext

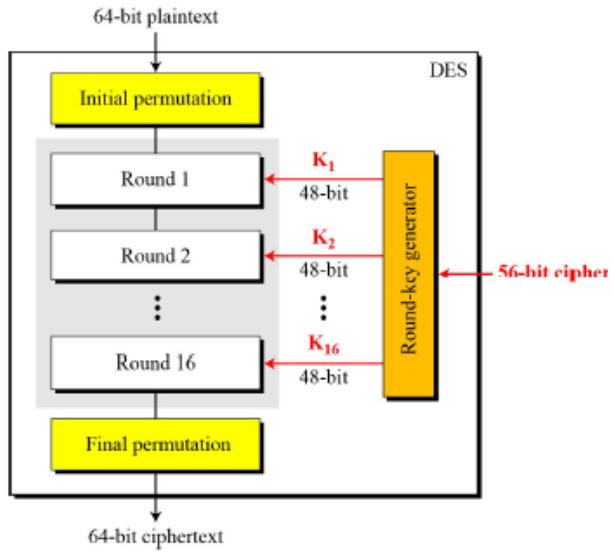
Stream and Block Cipher

- **Stream Cipher**
  - Enciphers *character by character*
  - *Faster* decoding
  - Computational power can be an issue for large plain text
  - *Low* diffusion
  - *Low* error propagation
  - *Susceptible* to malicious insertion
  - Ex: Caesar cipher
- **Block cipher**
  - Enciphers a *group of characters* each time
  - *Slower* decoding
  - Computational power is less of an issue
  - *High* diffusion
  - *High* error propagation
  - *Immune* to malicious insertions
  - Ex: Rail Fence cipher, DES

Strength of DES

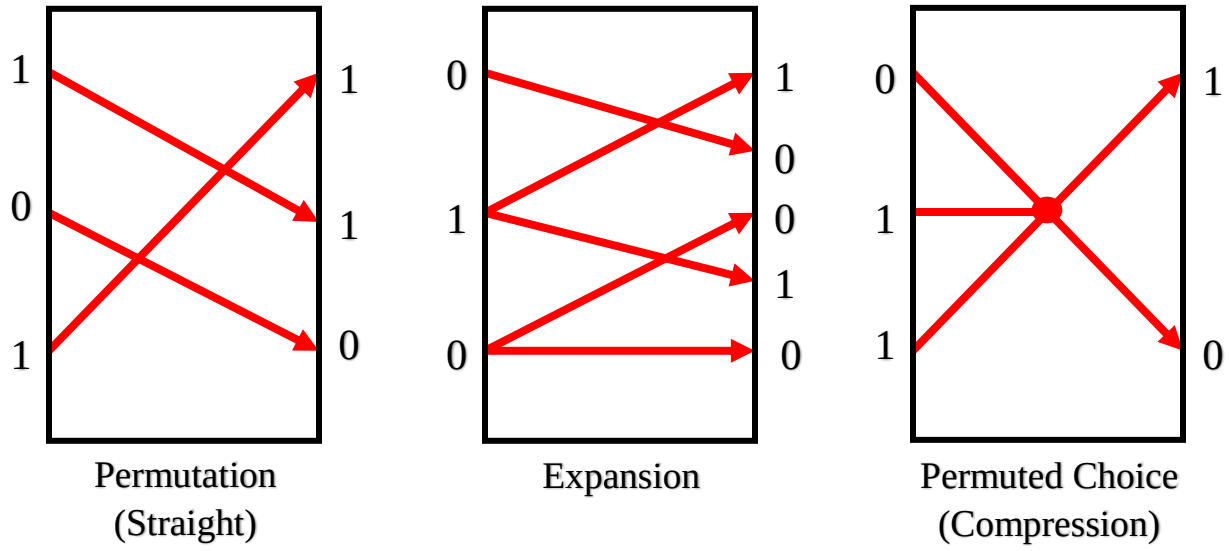
- A product cipher
  - Performs both *substitution* and *transposition* (permutation) on the bits
    - Employs both *confusion* and *diffusion*
- Feasible implementation
- Strong cryptographic properties (avalanche and completeness effect)

General Structure of DES

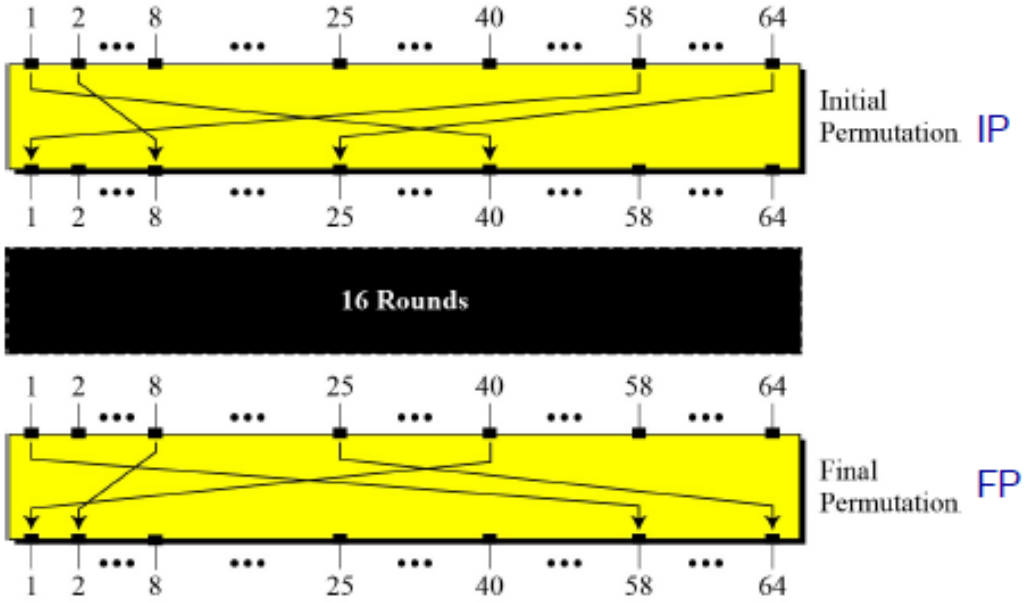


- 16 rounds (iterations) with 16 different system generated round keys (subkeys) that are derived from a single user supplied key
  - Sequential rounds
  - Subkeys used in reverse order for decryption

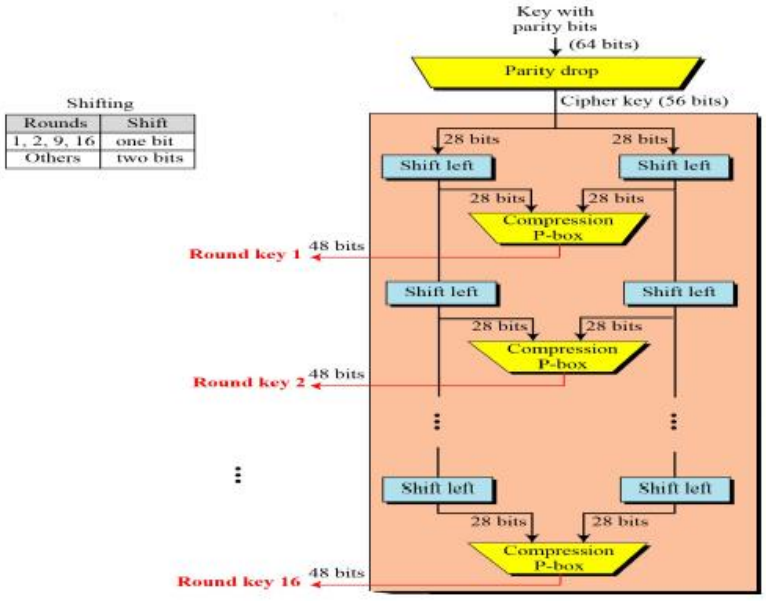
Types of Permutation



Initial and Final Permutation

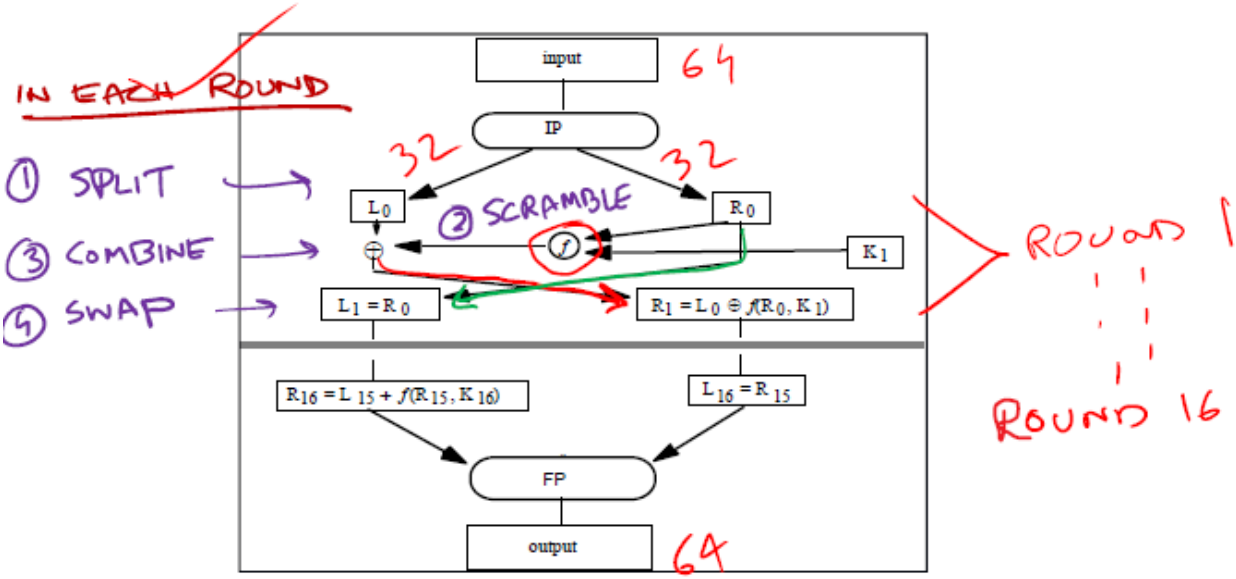


Key Generation in DES



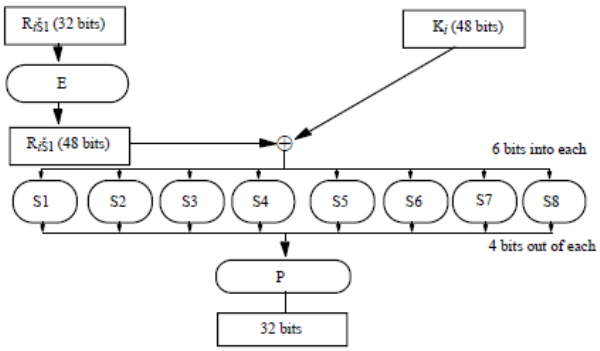
- Effective key length 56 bit (8 bit used for parity/error checking)
- Divides 64 into 28-bit halves and then shifts left by one bit
- 56 bits go into compression P-box and 48 come out. Some of the bits are changed, others disappear

Workings of DES



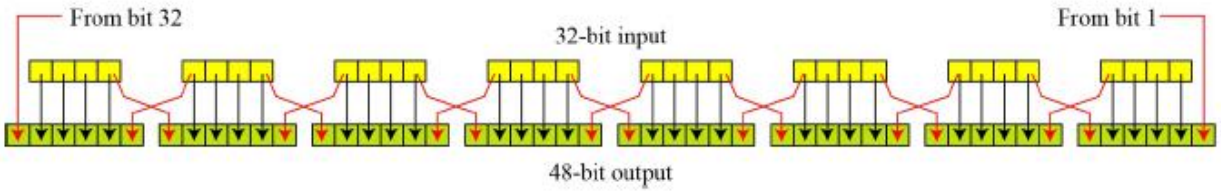
- In each round:
    1. Split – Input is 64, split in half
    2. Scramble – Feistel function (heart of DES)
        - Takes two inputs: the right side and the key (use the subkey for the first time)
    3. Combine – Output goes to XOR
        - Two inputs: left side and result of Feistel
        - XOR supports diversity
    4. Swap – The output from the right becomes the left and vice versa
- This repeats 16 times

Core of DES



- Feistel (f) Function
- Aka Scrambler
- S1 – S8 are called S boxes
- Each takes 6 bits and spits 4 out; total becomes 32 bits

DES Expansion Permutation



DES S-Box – Look up table Example

S6		Middle 4 bits of input															
		0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
Outer bits	**00**	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
	01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
	10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
	11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

- For 5th S-Box:
  - If input is: 011011
    - Outer bits are 01
    - Middle 4 bits are 1101
  - Output will be 1001
  - Example of substitution

Strong Cryptographic Properties of DES

- Avalanche effect
  - Small change in plain text or key makes significant change in the ciphertext
  - Example:
    - Plain text: `0000000000000000` Key: `228145104385VWUP`
    - Ciphertext: `7J388FNSWKR51234`
    - Plain text: `000000000000000`**`1`** Key: `228145104385VWUP`
    - Ciphertext: `6DBWK3479SH00FKJ2`
- Completeness effect
  - Each bit of the ciphertext depends on many bits of plaintext

IA & Crypto Midterm Study Guide

Term Comparisons:

1. Least Privilege vs. Least Authority
    - **Least Privilege**: The least permission necessary to do something **directly** and then discarded after use
    - **Least Authority**: The least permission necessary to have a **direct** or **indirect** effect
2. Accountability vs. Availability
    - **Accountability**: Holding a person accountable for their actions
    - **Availability**: Data and resources are kept usable
3. Incapacitation vs. Interception
    - **Incapacitation**: Prevent/Interrupt system operation by disabling a component (Disruption)
    - **Interception**: Data is accessed by an unauthorized source while it is being shared between authorized sources (Disclosure)
4. Spoofing vs. Sniffing
    - **Spoofing**: Disguising communication from an unknown source as being from a trusted source
    - **Sniffing**: Interception of data by capturing network traffic
5. Confidentiality vs. Privacy
    - **Confidentiality**: Keeping data and resources hidden
    - **Privacy**: Individual rights
6. Denial of Service vs. Denial of Receipt
    - **DoS**: An interruption in an authorized user's access to a computer network (Disruption)
    - **DoR**: User denies that they ever received something (Repudiation)
7. Least Common Mechanism vs. Ease of Mechanism
    - **LCM**: Resources should not be shared
    - **EOM**: Keep it as simple as possible (No Complexity)
8. Transposition Cipher vs. Substitution Cipher

- **Transposition**: *Rearrange* letters in plaintext to produce ciphertext (Goal is *diffusion*)
- **Substitution**: *Change/Replace* letters in plaintext to produce ciphertext (Goal is *confusion*)
    - Combinations of these are called **Product Ciphers**

9. Caesar Cipher vs. Vigenere Cipher
    - **Caesar**: Shifts each individual letter with a *monoalphabetic* key
    - **Vigenere**: Shifts each letter with a *polyalphabetic* key (a repeated word or phrase)
10. Confusion vs. Diffusion
    - **Confusion**: Replacing one character with another
    - **Diffusion**: Distribution/meaning spread throughout
11. Stream Cipher vs. Block Cipher
    - **Stream**: Enciphers character by character
    - **Block**: Enciphers a group of characters
12. Cryptography vs. Cryptanalysis
    - **Cryptography**: The use of encryption/decryption (Sender and Receiver)
    - **Cryptanalysis**: Attempts to break a cryptosystem (Interceptor)
13. Adhoc Attacks vs. Statistical Attacks
    - **Adhoc**: Uses experience to guess (no solid principle)
    - **Statistical**: Uses models of languages to make assumptions
14. Encoding v. Enciphering
    - **Encoding**: Coding entire words/phrases
    - **Enciphering**: Coding each letter/symbol individually
15. Secure vs. Trusted
    - **Secure**: A goal based on features. It either is or it isn't secure.
    - **Trusted**: A characteristic based on evidence analysis. Can come in different degrees
16. Safety vs. Security
    - **Safety**: A safe system **does not** guarantee a secure system
    - **Security**: A secure system **does** guarantee a safe system

Quiz 1 Solutions:

1. Using a weak password (vulnerability) on my phone (asset) allowed a hacker to guess (threat) it
2. Differences (See Above)
3. **Principle of Least Astonishment**: Mechanisms should be designed so that users understand the reason the mechanism works the way it does and its easy to understand
4. For resources (not data), why is it important to protect availability? You need to be able to access a computer (resource) so that you can have access to the data on it
5. What is the primary goal/theme of Saltzer and Schroder's secure design principles? **Restriction with Simplicity**
6. 5 aspects of "Defense in Depth": **prevent, deter, detect, respond, recover**
7. Matching:
    a. Confidentiality – Secrecy
    b. Integrity – Trust
    c. Availability – Accessibility
    d. Accountability – Non-repudiation
8. IA is perception
9. A process is requesting permission to write to files that are not required for the process's task completion. This request should be denied on the basis of **Least Privilege**
10. Two processes are required to complete a transaction. One initiates the transaction and the other completes after validating. This arrangement follows the principle of **Separation of Duties**
11. Two users from different project teams are prohibited from using the same database. This arrangement follows the principle of **Least Common Mechanism**
12. An authentication system is an example of **Complete Mediation**
13. Article from Wired magazine:
    a. Which CIA triad did this attack violate?
        - Confidentiality – Access to confidential information
        - Origin Integrity – Acted as professors
    b. Which categories of threat does this incident relate to?
        - Disclosure – credentials

- Deception – Phishing emails

c. Which categories of attacks does this incident relate to?
   - Exposure – Access to credentials
   - Masquerade – Acted as professors
   - Intrusion – Hacked into the system

Quiz 2 Solutions

1. Differences (See Above)
2. On my computer (asset), my information should be kept private, so I use a secure password.
3. The user can only read and write at the same level that they are at.
4. If a user has ever had access to something in the same COI class, then they are not allowed to access anything else in that class.
5. In Clark Wilson, the focus is on data transactions.
6. Matching:
   a. CISS – Health Sector
   b. Biba – Commercial Sector
   c. BLP – Military Sector
   d. Chinese Wall – Financial Sector
   e. CISS – Patients' Interest
   f. Biba – Information Flows Down
   g. BLP – Information Flows Up
   h. Chinese Wall – Conflict of Interest

   i. Example of Constrained Data Item – Log of Transaction (This must be tamper-proof)
   j. Transformation Procedure – Handles Untrusted Input
   k. Integrity Verification Procedure – Ensures Validity of Trusted Data
   l. Example of Unconstrained Data Item – Company portfolio (Open to Public)

   m. CISS – Least Privilege
   n. Clark Wilson – Separation of Duty
   o. Chinese Wall – Least Common Mechanism
   p. Reference Monitor – Complete Mediation
   q. Certification Rule – Monitors Integrity & Carried out by External Entity
   r. Enforcement Rule – Preserves Integrity & Carried out by System

7. In reference to Clark-Wilson Model, a “certified relation” associates: Transformation procedures and constrained data items
8. In reference to Clark-Wilson Model, an “allowed relation” associates: Users, Transformation Procedures, and Constrained Data Item
9. Using BLP:
    a. Amy (Unclassified, {X}) W File1 (Top Secret, {X,Y})
    b. Jill (Secret, {X, Y}) N File2 (Top Secret, {X,Z})
    c. Susan (Classified, {Y,Z}) R File3 (Classified, {Z})
    d. Bill (Classified, {}) N File4 (Unclassified, {X})
    e. Ana (Secret, {X}) B File5 (Secret, {X})
10. Fill in the Blanks:
    a. In CISS, the auditors can **Read** audit logs
    b. In CISS, data from file A can be appended to file B, if and only if **B’s** access control list is a subset of **A’s** access control list
    c. In CISS, the focus is on **integrity/medical**, while in BLP, the focus is on **confidentiality/government**.
    d. In capabilities list, the focus is on the **subject**, while in ACL, the focus is on **object.**
    e. Clark Wilson ensures internal **consistency/accuracy** with external **reality.**
    f. Clark Wilson ensures well-formed **transactions.**
11. Show ACM for the following

	OS	Web App	Personal Data	Audit Log
Steve	RWX	RWX	RWX	RWX
Megan	X	X	RW	
Jimmy	RWX	RWX	RWX	R
Trevor		RWX	RWX	
Jake				

Assume Steve leaves and Megan is promoted to system admin. Revise ACM:

	OS	Web App	Personal Data	Audit Log
Steve				
Meg	RWX	RWX	RWX	RWX

12. In reference to the Chinese Wall Model, what can Anderson do in the following cases:
    a. When he has not started to work for any of these companies:
        - Anderson can read any **one** from phone, beverage, or tech
        - He cannot write any
    b. When he has access to Sprint
        - Read any one beverage or tech
        - Write sprint
    c. When he has read access to Sprint and Apple?
        - Read any one beverage
        - Write none
    d. When he wants to read Verizon's public annual report?
        - He can, it's public
13. True or False:
    a. F – Data at a higher confidentiality level is more secret than data at a lower integrity level.
        - Can't compare confidentiality and integrity
    b. T – Data at a higher integrity level is more trustworthy than data at a lower integrity level
    c. F – A trusted system may not originate from a safe model.
        - Trusted is always proved to be safe
    d. F – In CISS, data can only be deleted if the patient dies
        - Or the appropriate amount of time has passed
    e. F – Any clinician in a patient's ACL can add another clinician
        - Only the responsible clinician may do this
    f. F – If there are 3 company datasets in one COI class, the we need a maximum of 3 analysts to cover all companies
        - We need at least 3 because there can not be one analyst working on two datasets in the same COI class
    g. F – Commercial policy focuses on integrity, availability, and hierarchical confidentiality.
        - Focuses on integrity, availability, and **non**-hierarchical confidentiality

h. T – Capabilities lists allow easier revocation of user rights
i. T – Explicit security policy allows us to hold violators accountable
j. F – In Chinese Wall, one does not place restrictions on unsanitized data
k. F – In Clark-Wilson Model, one does not place restrictions on constrained data
    - Constrained = Unsanitized
l. T – In Chinese Wall, you are free to access anything initially (once)

Emily Loftis

Mid Term Exam Date: 02/27/19

CSC 4575/5575

Time: Until 4:20 PM Total points: 80

1. What are the FIVE layers of the "defense in depth" strategy? No explanation needed [2.5]

1) **Prevent**
2) **Deter**
3) **Detect**
4) **Response**
5) **Recover**

(Intro to Basic Security Concepts slide 20)

2. What are the EIGHT primary Saltzer and Schroeder's design principles? No explanation needed [4]

1) **Least Privilege**
2) **Fail-Safe Defaults**
3) **Economy of Mechanism**
4) **Complete Mediation**
5) **Open Design**
6) **Separation of Privilege**
7) **Least Common Mechanism**
8) **Psychological Acceptability**

(Security Principles slide 3)

3. Which conditions allow one to WRITE in each of the following models? [4]

a. BLP
   - **Subject s can write to object o iff $L(s) \leq L(o)$ and s has permission to write to o.**
   - **A subject can write to an object as long as the object is at the same or a higher security level than the subject.**

   (Security Models – BLP, Biba slide 5)

b. Biba
   - **Subject s can write to object o iff $i(s) \geq i(o)$ and s has permission to write to o.**
   - **A subject can write to an object if the object is at same security level or lower than the subject.**

   (Security Models – BLP, Biba slide 16)

c. Chinese Wall
   - **The CW-simple security condition permits s to read o (has read permissions)**
   - **S can write to an object if all the unsanitized objects it can read are in the same dataset.**

   (Security Models – CW slide 11)

d. CISS
   - **(Access Principle 1 – ACL) Each record has an ACL naming the individuals or groups who may read, update, and append information to the record.**

   (Security Models – CW, CISS, CLW slide 18)
   - **(Creation Principle) A clinician may open a new record, with the clinician and the patient on the ACL.**

   (Security Models – CW, CISS, CLW slide 22)
   - **Based on the above points, clinicians and other groups listed on the ACL are allowed to write in this model.**

4. What IS the MAIN difference between: [5X2=10]

- Safety and security
   - **Safety: refers to the abstract model; a safe model does not guarantee a secure system**
   - **Security: refers to implementation; a secure system does guarantee a safe model**

   (Security Policies and Models slide 33)

- Chosen plain text attack and chosen cipher text attack
   - **Chosen plain text: Adversary may supply specific plaintexts of interest and obtain corresponding ciphertext; goal is to find the key. Adversary has access to cryptosystem on sender's (encryption) side**
   - **Chosen cipher text: Adversary may supply specific ciphertexts of interest and obtain corresponding plaintext; goal is to find key. Adversary has access to cryptosystem on receiver's (decryption) side**

   (Cryptography slide 8)

- Confusion and diffusion
   - **Confusion: Goal of substitution ciphers; Changing/replacing characters in plaintext to produce ciphertext**

   (Cryptography slide 19)
   - **Diffusion: Goal of transposition ciphers; Rearranging letters in plaintext to produce ciphertext (letters not actually changed)**

   (Cryptography slide 15)

- Certified relation and allowed relation
   - **Certified: Associates a set of CDIs with a particular TP**

   (Security Models CW, CISS, CLW slide 35)
   - **Allowed: Associates a user with each TP and set of CDIs**

   (Security Models CW, CISS, CLW slide 36)

- Privilege and authority
    - **Privilege: means permissions determining direct actions on the entty in question**
    - **Authority: means what effects it has on the entity in question either directly or indirectly through another user/application/service**

  (Security Principles slide 4)

5. What are the two pillars of the Clark Wilson Model? [2]

1) **Well-formed transactions**
2) **Separation of duty**

(Security Models CW, CISS, CLW slide 32)

6. If a system implemented BLP, in what security level of BLP would you place a Virus in order to prevent it from spreading? Explain. [2]

- **For BLP information flows up, so subjects can create content at or above their own security level. If you place a Virus at the highest security level, then it would only be able to write to the highest level and not easily spread to any other.**

  (Security Models – BLP, Biba slide 5)

7. List any two good cipher characteristics described by Shannon [2]

1) **Need for protection should determine cost/effort**
2) **None or Minimal complexity for usage**
3) **Simple/reasonable implementation**
4) **Errors should not propagate**
5) **Cipher text should not be longer than original text**

(Cryptography slide 51)

8. If the period of a poly-alphabetic cipher in English is 3, what is the possible key space? [1]

Key space = 26^3

(Cryptography slide 21)

9. From frequency examination with the following table, what are the likely keys for EACH of the 3 rows of cipher text frequency counts? [3]

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**	Likely Key
1	1	0	0	1	0	0	0	0	0	4	0	0	1	0	1	2	0	0	2	0	1	0	1	2	1	**K**
3	2	0	1	0	1	1	1	2	0	0	1	4	0	1	0	4	0	1	5	0	0	0	0	0	0	**A**
1	1	2	0	1	0	1	2	0	1	0	0	0	0	3	1	3	0	1	0	1	0	2	0	1	0	**O**

(Cryptography slide 42)

10.Give one example EACH of a mono-alphabetic cipher, a poly-alphabetic cipher and a product cipher. [1.5]

1) **Mono-alphabetic: Caesar Cipher**
(Cryptography slide 28)
2) **Poly-alphabetic: Vigenere Cipher**
(Cryptography slide 29)
3) **Product: DES**
(Cryptography slide 56)

11. Using the S-Box below and the rule we discussed in class, what would be the output for the following inputs? [2]

110001 <u>0101</u> **000101 <u>0111</u>**
(Cryptography slide 65)

↓	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1110	0100	1101	0001	0010	1111	1011	1000	0011	1010	0110	1100	0101	1001	0000	0111
01	0000	1111	0111	0100	1110	0010	1101	0001	1010	0110	1100	1011	1001	0101	0011	1000
10	0100	0001	1110	1000	1101	0110	0010	1011	1111	1100	1001	0111	0011	1010	0101	0000
11	1111	1100	1000	0010	0100	1001	0001	0111	0101	1011	0011	1110	1010	0000	0110	1101

12. If cipher text is C and key is 5, what is the plain text? Use the equation (discussed in class) to derive your answer. [1.5]

D = (26 + c – k) mod 26
D = (26 + 3 – 5) mod 26
D = 24 = Y
(Cryptography slide 20)

13. With key=CAT , change the following Vignere ciphertext into multiple Caeser ciphertext [1.5] Vignere ciphertext: **WTNOPFHMPWGN**

1) **WOHW**
2) **TPMG**
3) **NFPN**
(Cryptography slide 40)

14. Calculate the index of coincidence for EEWPGWCYE [2].

Show your work (fraction is fine).

❑ $IC = [n\,(n-1)]^{-1} \sum_{0 \le i \le 25} [F_i\,(F_i - 1)]$

E = 3; W = 2; P = 1; G = 1; C = 1; Y = 1
IC = [9 (9 - 1)]^(-1)*[3 (3 - 1) + 2 (2 - 1)]
IC = (6 + 2) / 72 = 8 / 72
(Cryptography slide 38)

15. What is the avalanche effect? [2]
- **Small change in plain text or key makes significant change in the ciphertext.**

(Cryptography slide 66)

16. We refer to DES as a symmetric block cipher. Why? [2]
- **It is a block cipher because it processes each character in groups.**
- **It uses the same algorithm and keys for encryption/decryption.**

(Cryptography slide 54)

17. We refer to DES as a product cipher. Why? [1]
- **It performs both substitution and transposition (a.k.a. permutation) on the bits (employs both confusion and diffusion).**

(Cryptography slide 56)

18. What makes one time pad secure? [2]
- **It is truly random, never reused, and kept secret.**

(Cryptography slide 50)

19. What was Kaskski's contribution? [2]
- **He recognized that repetitions in ciphertext occur when characters of the key appear over the same characters in the plaintext. The period is a factor of distance between repetitions.**

(Cryptography slide 34)

20. Identify the types of Permutation used for processing Cryptography block ciphers. Write E for expansion, P for straight and PC for compression [1.5]

Input 16 bits, output 12 bits	**Permutated Choice (Compression)**
Input 16 bits, output 24 bits	**Expansion**
Input 16 bits, output 16 bits	**Permutation (Straight)**

(Cryptography slide 58)

21. Explain why [3]:

a. Without any confidentiality control in a system, there is risk of integrity loss for system data and resources.

- **Without confidentiality control, you can look at everyone's password and use that to get into the system and make changes to files and passwords, leading to loss of integrity.**

b. Without any integrity control in a system, there is risk of confidentiality loss for system data and resources.

- **If you can make changes to a system, then you can change the security levels which would allow unauthorized users to have access to data that they should not, leading to loss in confidentiality.**

(Intro to Basic Security Concepts slide 11)

22. Identify whether the following statements apply to Stream or Block ciphers. Mark S for Stream and B for Block. [2].

B High diffusion

B Immune to malicious insertions

B High error propagation

B Slower decoding

(Cryptography slide 52)

23. Fill in the blanks [1.5]:

DES Sub Key Size **48** (Cryptography slide 63)
DES Key Size **64** (Cryptography slide 60)
DES Effective Key Size **56** (Cryptography slide 60)

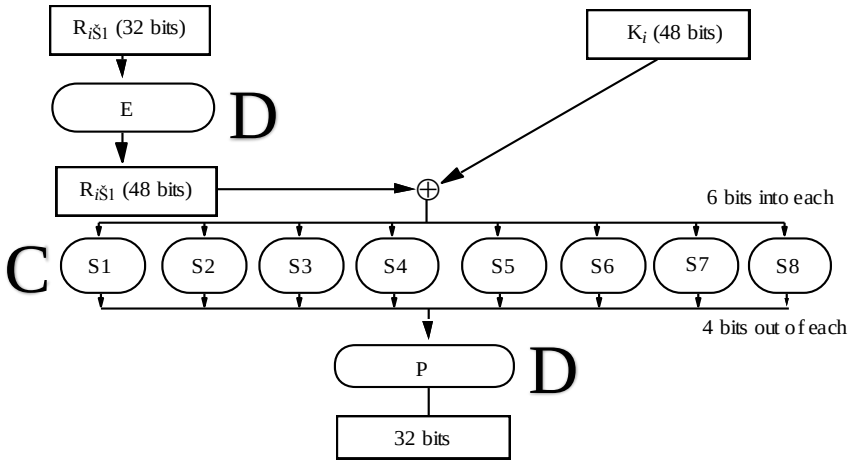
24. What is the period for this cipher text? How did you identify it [2]?

G	L	Q	T	A	L	M	H	Q	T	A	I	N	W	X	K	Z	U	V
1																		19

The distance between the Qs is 6 = (3*2).
The distance between the Ls is 4 = (2*2).
The greatest common factor is 2.
The period is 2

(Cryptography slide 30)

25. In the Feistel Function below, MARK to show where the following takes place [3]:
Confusion (mark as C) Substitution
Diffusion (mark as D) Transposition



(Cryptography slide 63)

26. For ONE of your personal electric/digital assets, IDENTIFY a security threat, a vulnerability and USE them to articulate a risk statement. [3].

- **If I use a weak password on my phone, then it would be easier for a hacker to gain access.**
  - **Asset = Phone**
  - **Security Threat = Hacker gaining access**
  - **Vulnerability = Weak Password**

  (Class Quiz 1 Review)

27. True or False [14]:

F Strength of encryption/decryption algorithm depends on their secrecy.

F Encrypted messages remain secret because unauthorized users cannot see them.

F Encryption/decryption keys should be as secret as the algorithms.

T Substitution ciphers are subjected to frequency analysis attacks.
(Crypto slide 23)

T In transposition cipher, the goal is diffusion.
(Crypto slide 15)

T Statistical attacks are based on models of language.
(Crypto slide 23)

T A key with longer period makes smaller index of coincidence.
(Crypto slide 38)

F The period of a key is smaller than the key size.
(Crypto slide 30)

F Anagramming is a technique applied for decoding substitution ciphers.
(Crypto slide 16)

T In poly-alphabetic ciphers, different plaintext can transform into same ciphertext.
(Crypto slide 29)

T In mono-alphabetic ciphers, same plain text will transform into same cipher text.
(Crypto slide 28)

T Cryptanalysis is breaking ciphers. (Attempt to break the cryptosystem?)
(Crypto slide 6)

T Inference does not require direct access to sensitive information.

T I am determined to work very hard on my term project.

28. What is _____ ?

Choose any question related to the material covered for this test (but has not appeared in this test) AND ANSWER accordingly. EXTRA EASY POINTS!! [2]

March 12th – Exam Review & Crypto Continued

Strong Cryptographic Properties of DES

- **Avalanche Effect**
  - *Small* change in plain text or key makes *significant* change in the ciphertext
- **Completeness Effect**
  - Each bit of the ciphertext depends on *many* bits of plaintext
- (These are not the main reasons that DES is strong, they are the outcomes)
  - DES is strong because it applies confusion and diffusion so many times

Weak Properties of DES

- Out of 2^56 keys
  - **4 weak keys** (all 1s, all 0s, half 1s & 0s, half 0s & 1s)
    - They are their own inverses
      - C = E(P, K)
      - P = E(C, K)
    - Effect of decryption with a weak key can be achieved by encryption with the same
  - **12 semi-weak keys** (6 K1, K2 pairs)
    - Each has another semi-weak key as inverse
      - C = E(P, K1)
      - P = E(C, K2)
    - Effect of decryption with a semi-weak key K1 can be achieved by encryption with the corresponding semi-weak key K2.
- **Complementation Property**
  - Knowing half of DES keys, you can guess the other half
  - Brute force attack effort could be reduced by a factor of 2
- S-boxes exhibit irregular properties
  - Did not randomize input properly
  - Output of fourth S-box depends on input to third S-box

- P-box (permutation) mystery
    - No cryptographic significance of initial and final permutation

Controversy with DES

- Considered too weak
    - Short key size (56 bit)
    - Design decisions not public (initially)
        - "Suspicious" S-boxes
            - May have trapdoors
- Diffe, Hellman (1977) designed a parallel machine to break DES by brute force and predicted that in a few years, technology would allow DES to be broken in days
    - 1997, 3500 machines worked in parallel working for 4 months
    - 1998, DES Cracker in a few days
    - 1999, DES Dracker was distributed in less than a day

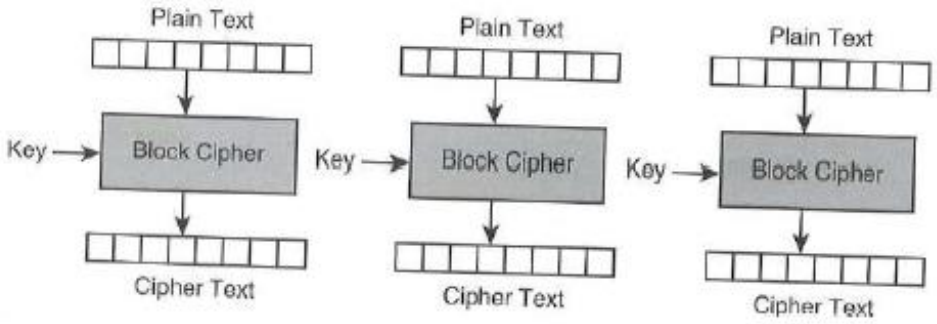
Cryptanalysis for DES

- Biham and Shamir's **Differential cryptanalysis** in 1990
    - A chose plaintext attack
        - Required 2^47 plaintext, ciphertext pairs to break the cipher
    - Observing strength change with change in algorithm through statistical analysis
    - Revealed several properties
        - Small changes in S-boxes weakened the cipher
            - Optimal design
        - Longer key size did not make it resistant to differential cryptanalysis
- Matsui's **Linear cryptanalysis** in 1994
    - Known plaintext attack
    - Relies on linear approximation and statistical analysis
    - Improved result
        - Require 2^43 plaintext, ciphertext pairs

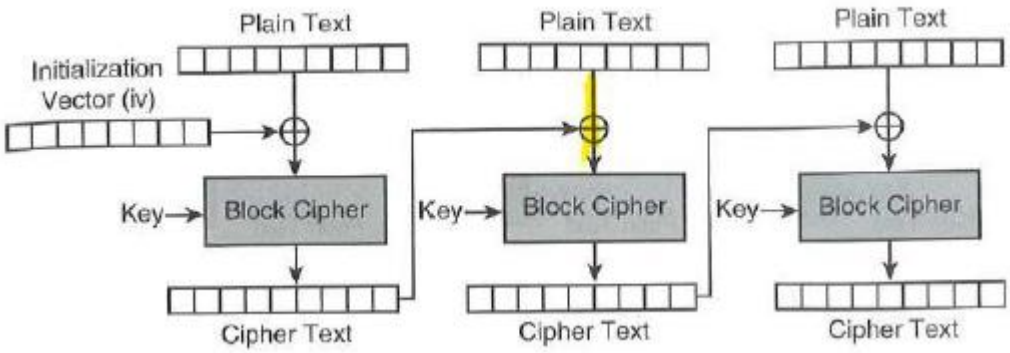
March 14th

Single Execution DES (or other Block Cipher) Modes

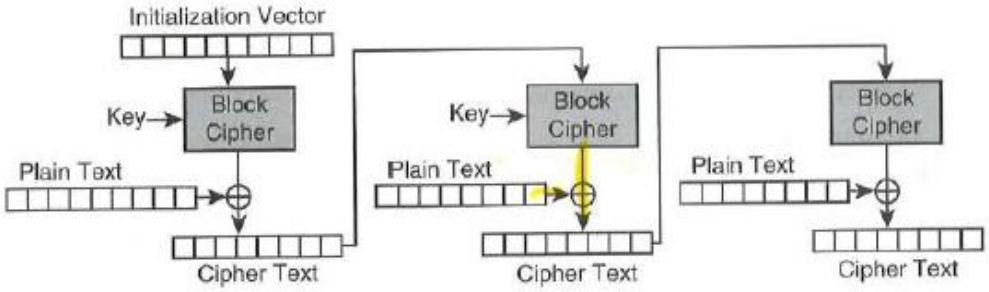
- **Direct Mode or Electronic Code Book Mode (ECB)**
    - Each block encrypted separately
    - Susceptible to attacks
    - Does not add anything to DES



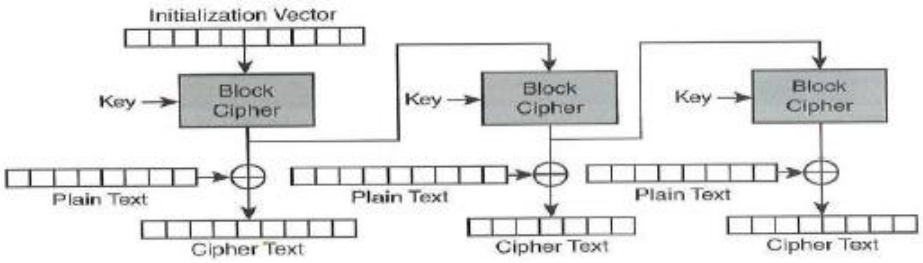
- **Cipher Block Chaining Mode (CBC)**
    - Ciphertext output from each block is used for the next block
        - XORed with next plaintext block before encryption
        - First block encrypted with IV (*initialization vector*)
    - Self-healing property of CBC (*Self-synchronizing*)
        - If one block of ciphertext is corrupted, *error propagates* to at most *two* blocks
            - i.e. if ck is corrupted, it will only propagate up to ck+1, and ck+2 would be error free
            - Plaintext "heals" after 2 blocks



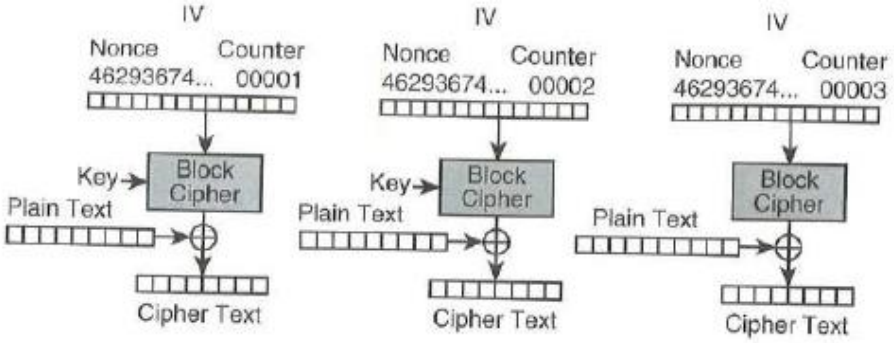
- **Cipher Feedback Mode (CFB)**
    - Ciphertext output from each block is used for the next block
        - Serves as the only input to encryption
        - Plaintext of next block is XORed with the encryption result to produce the next ciphertext



- **Output Feedback Mode (OFB)**
    - Output from each encrypted block is used for the next block
        - Serves as only input to encryption
        - Plaintext is XORed with the encryption result to produce the ciphertext



- **Counter (CTR)**
    - Uses a "**nonce**" (a random number that is used once—each nonce is different) concatenated with a counter, which is encrypted by the block cipher, and the output XOR's with the plaintext block to produce the next ciphertext



Mode Comparisons

	Input to Block Cipher	**Cipher Text Generated By**	**Each block connected with previous block**
ECB	Plain Text	Output of block cipher	No
CBC	XOR and plaintext and output of previous block	Output of block cipher	Yes
CFB	Ciphertext of previous block	XOR of plaintext and output of block cipher	Yes
OFB	Output of previous block cipher	XOR of plaintext and output of block cipher	Yes
CTR	Random number	XOR of plaintext and output of block cipher	Yes

Which Mode is Better?

- Predictability
    - Do identical messages need to produce the same cipher message or different one?
    - Same message, same key = counter feedback or any other mode with different initialization vector
    - Different message, use different IV and same key (and not Counter Feedback)
- Error Propagation
    - Error prone medium?
- Security against adversaries
- Efficiency/speed
    - Need to be parallelizable (can only be done for decoding)
    - All except OFB because the input to the next block cipher is just the output from the one before it.

Multiple execution DES Modes

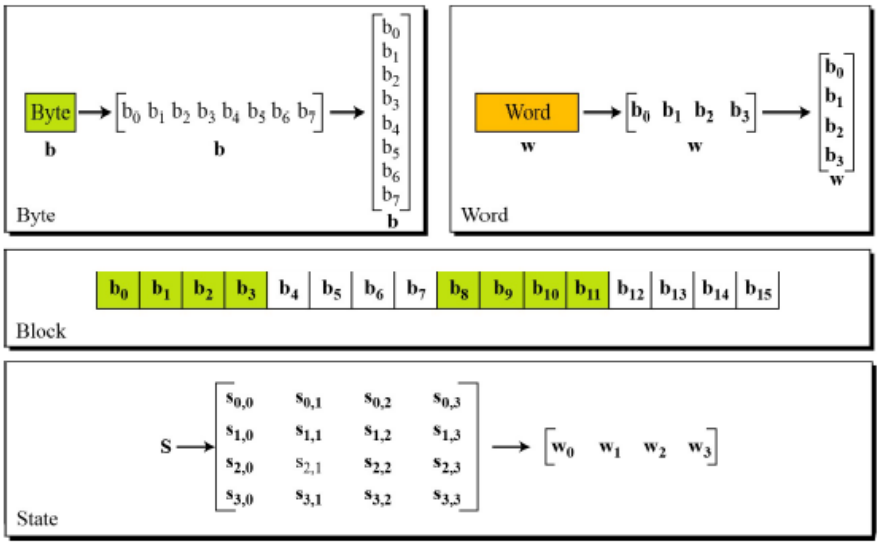
- **Double Execution (Double DES)**
  - Encrypt-Encrypt Mode (2 keys: k, k')
    - $C = DES_k(DES_{k'}(m))$
  - Double DES is only as secure as DES
    - Effective Key length is same. Only doubles the work for the attacker
    - Susceptible to "**meet in the middle attack**", a known plaintext attack on Double DES
  - Meet in the Middle Attack
    - A known plaintext attack where the attacker does time memory tradeoff analysis while encrypting from one end and decrypting from the other end to meet in between where the match is found to guess the same keys
    - $C = E_{k2}(E_{k1}(P))$
    - $P = D_{k1}(D_{k2}(C))$
    - $M = E_{k1}(P) = D_{k2}(C)$
      - Use brute force to come up with all possible middle text based on the encryption; compare the results and when they match, then you have the answer
      - Does not make it twice as hard to crack DES; 2^(56+1) = 2^57 (too much effort so such little effect
- **Triple Execution (Triple DES)**
  - Modes:
    - Encrypt-Decrypt-Encrypt Mode (2 keys: k, k')
      - $C = DES_{k(}DES_{k'}{}^{\wedge}\text{-}1(DES_k(m)))$
    - Encrypt-Encrypt-Encrypt Mode (2 keys: k, k')
      - $C = DES_{k(}DES_{k'}(DES_k(m)))$
    - Encrypt-Encrypt-Encrypt Mode (3 keys: k, k', k'')
      - $C = DES_{k(}DES_{k'}(DES_{k''}(m)))$
  - Triple DES much more secure than DES
    - Double effective key length (112 bit). Still secure. Approved until 2030
  - If you have to use DES, use triple DES; it has not been broken yet

DES to AES

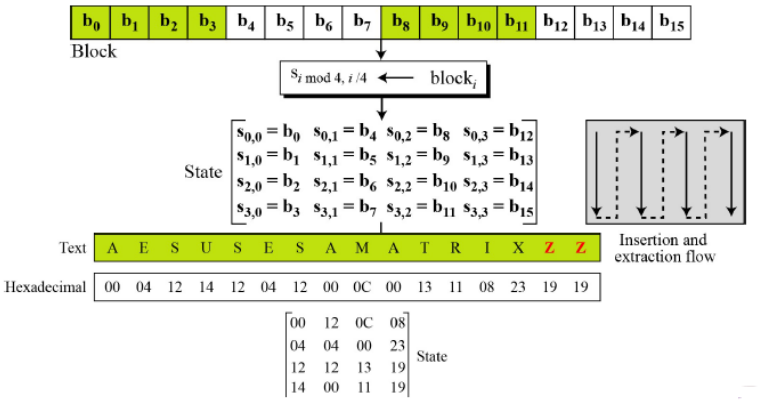
- With future anticipation of technology
  - NIST called for contest in 1997
  - 5 finalists
  - Final selection based on security as well as cost and efficiency
  - NIST selected Rijndael from Dutch Cryptographers as the winner
  - **Advanced Encryption Standard** became the Federal Information Processing Standard, FIPS 197 (2001)

AES

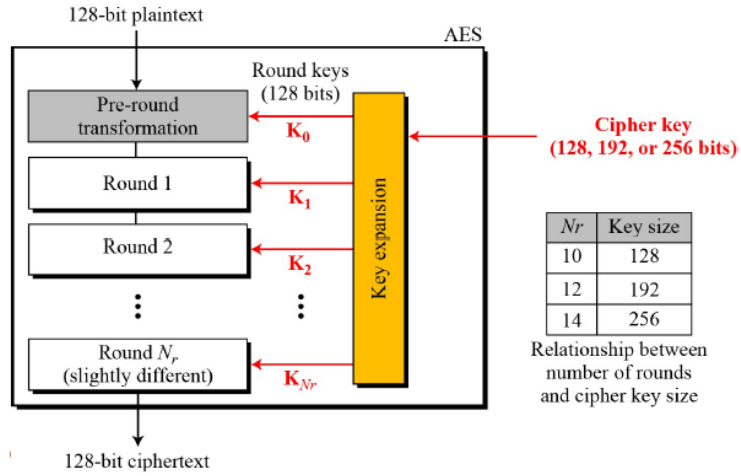
- FIPS-197
- Designed to withstand attacks that were successful on DES
  - Security, cost, implementation
- Characters of AES
  - 128 bit block
  - Key size 128/192/256 bit and Rounds respectively 9/11/13
  - Uses bit level multiplication along with substitution and permutation
  - No Feistel function
    - No secrecy
- AES Data Units



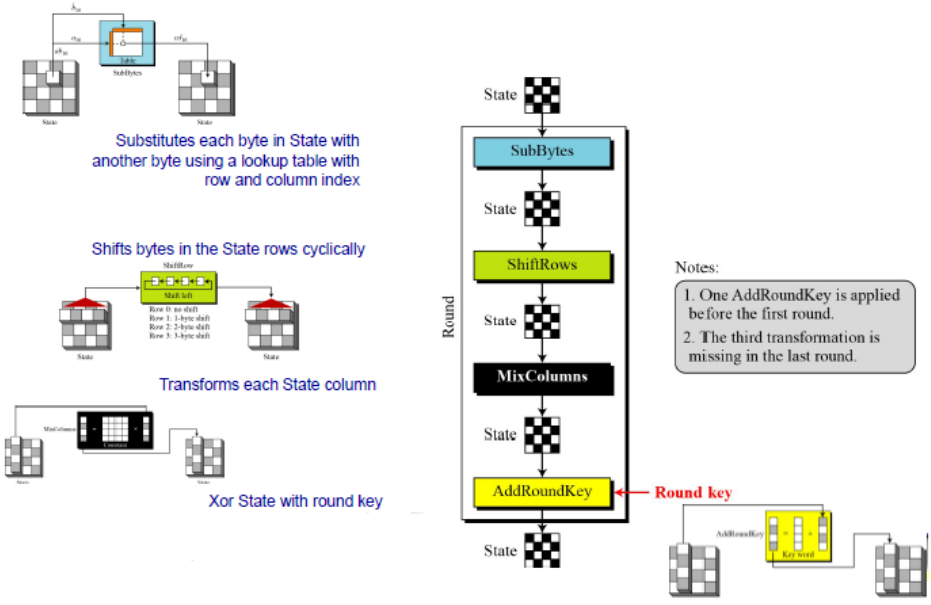
- AES State



- AES Design



- AES Round



- AES Security
    - Has not been broken *yet*
        - No brute force attacks
        - No statistical attacks
        - No linear/differential attacks
    - Efficient and simple implementation
        - Software, hardware, firmware
    - Available royalty free world wide

Public Key Cryptography

- Asymmetric key cryptography
- Proposed by Diffie and Hellman, 1976 (Same people that created DES Cracker)
- Two Keys
    - Private key known only to individual
    - Public key available to anyone

Symmetric and Asymmetric Cryptography

- Symmetric: Same key locks and unlocks
- Asymmetric: One key locks, the other unlocks

Public and Private Key

- Mathematically related **AND** generated together
    - If message encrypted by one, can only be decrypted by the other
- For example, 3 users: each of the 3 has **ONE** pair of keys
    - Public key
        - Known/accessible to anyone who wants to communicate with that user
    - Private Key
        - **ONLY** known to that user
- Each of the three should also have Public key of the other users
    - Public key of the other users

Public Key Cryptography Goal

- Confidentiality
- Data Authentication (Integrity)
- Origin Authentication
    - Non-repudiation

Requirements

1. It must be computationally easy to encipher or decipher a message given the appropriate key
2. It must be computationally infeasible to derive the private key
    - From the public key
    - From a chosen plaintext attack

RSA

- Rivest, Shamir, Adleman, 1977
- Exponentiation cipher
    - Based on exponentiation arithmetic

RSA Background

- One way function properties:
    - Given x, $y = f(x)$ is easily computed
    - Given y, $x = f^{-1}(y)$ is computationally infeasible to calculate
- Trapdoor: One way function with additional property:
    - Given y and a secret (trapdoor), x can be computed easily
- Prime numbers
    - Numbers that have no factor common (or GCD = 1) with any other numbers
- Totient function $\phi(n)$
    - The number of numbers relatively prime/coprime (no factors in common or GCD = 1) to a larger integer n

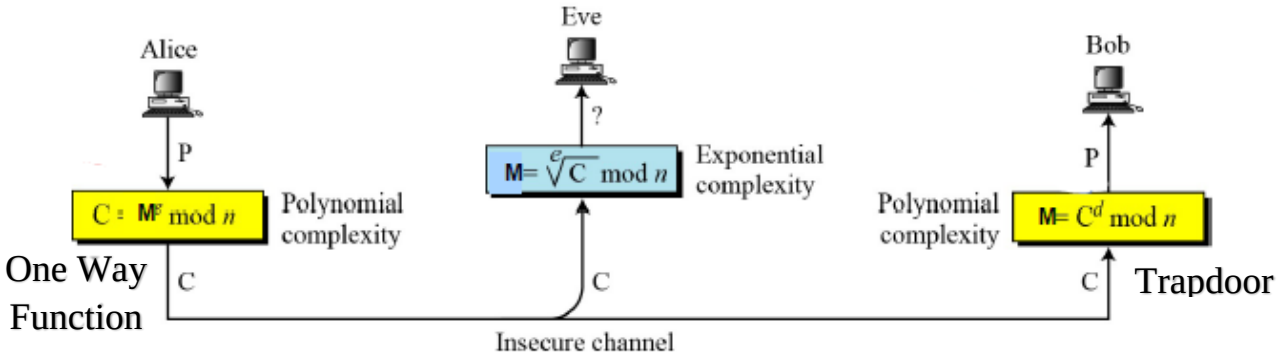
- Example: ϕ(21) = 12
  - 21 = 3 * 7
  - 1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20 are relatively prime to 21
- Example: ϕ(10) = 4
  - 10 = 2 * 5
  - 1, 3, 7, 9 are relatively prime to 10

RSA Algorithm

- Choose two large prime numbers p, q
  - Compute **n = pq**
  - Compute **ϕ(n) = (p – 1)(q – 1)**
  - Choose 1 < e < n such that e is **relatively prime** to ϕ(n)
  - Computed d such that **ed mod ϕ(n) = 1**
- Public key: **(e, n)**
- Private key: **d**
- Encipher: **c = m^e mod n**
- Decipher: **m = c^d mod n**

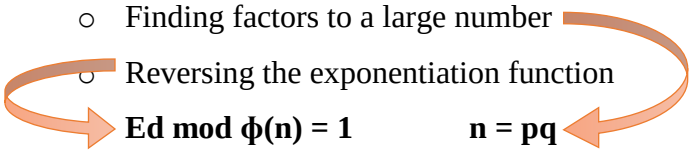
One Way Function and Trapdoor in RSA

- If attackers intercept ciphertext, recovering plaintext without knowing private key, is infeasible



Strength of RSA

- Relies on the difficult of
  - Finding factors to a large number
  - Reversing the exponentiation function



$\frac{Ed - 1}{} = k$ some constant

$\phi(n)$ → is equal to $(p - 1)(q - 1)$

- Knowing e & n does not make deducing d trivial, because attacker
  - Won't know K
  - Won't know P & E pair

March 19th – Crypto

Confidentiality with RSA

- To send message
  - Encipher using the receiver's public key
  - Decipher using the receiver's private key

Example: Confidentiality with RSA

- Take p = 7, q = 11, so n = 77
- Alice is receiver and Bob is sender
- Alice chooses e = 17, making d = 53
- Bob wants to send Alice a secret message HELLO (07 04 11 11 14)

  $07^{17} \bmod 77 = 28$

  $04^{17} \bmod 77 = 16$

  $11^{17} \bmod 77 = 44$

  $11^{17} \bmod 77 = 44$

  $14^{17} \bmod 77 = 42$

  Alice's keys: (17, 77); private: 53

  Bob's keys: public: (37, 77); private: 13
- Alice receives 28 16 44 44 42
- Alice uses private key, d = 53, to decrypt message:

  $28^{53} \bmod 77 = 07$ – H

  $16^{53} \bmod 77 = 04$ – E

  $44^{53} \bmod 77 = 11$ – L

  $44^{53} \bmod 77 = 11$ – L

  $42^{53} \bmod 77 = 14$ – O
- Alice translates message to letters to read HELLO

Integrity with RSA

- To send message
  - Encipher using sender's private key
  - Decipher using sender's public key

Example: Data and Origin Authentication with RSA

- Alice is sender and Bob is receiver
- Alice wants to send Bob the message HELLO (07 04 11 11 14) so Bob knows it is what Alice sent (no changes in transit, and authenticated)

  $07^{53} \bmod 77 = 35$

  $04^{53} \bmod 77 = 09$

  $11^{53} \bmod 77 = 44$

  $11^{53} \bmod 77 = 44$

  $14^{53} \bmod 77 = 49$

  Alice's keys: public (17, 77); private: 53

  Bob's keys: public: (37, 77); private: 13
- Bob receives 35 09 44 44 49
- Bob uses Alice's public key, e = 17, n = 77, to decrypt message:

  $35^{53} \bmod 77 = 07 – H$

  $09^{53} \bmod 77 = 04 – E$

  $44^{53} \bmod 77 = 11 – L$

  $44^{53} \bmod 77 = 11 – L$

  $49^{53} \bmod 77 = 14 – O$
- Bob translates message to letters to read HELLO
  - If (enciphered) message's blocks (letters) altered in transit, it would not decrypt properly

Confidentiality and Integrity with RSA

- Double encipherment on sender's part
  - First encipher using sender's private key and then encipher the result again with the receiver's public key
- Double decipherment on receiver's part
  - First decipher using receiver's private key and then decipher the result again with the sender's public key

Example: Confidentiality, Data, and Origin Authentication with RSA

- Alice is sender and Bob is receiver
- Alice wants to send Bob the message HELLO both confidentiality and integrity-checked
- Alice enciphers HELLO (07 04 11 11 14)

  $(07^{53} \bmod 77)^{37} \bmod 77 = 07$

  $(04^{53} \bmod 77)^{37} \bmod 77 = 37$

  $(11^{53} \bmod 77)^{37} \bmod 77 = 44$

  $(11^{53} \bmod 77)^{37} \bmod 77 = 44$

  $(14^{53} \bmod 77)^{37} \bmod 77 = 62$

  Alice's keys: public: (17, 77); private: 53

  Bob's keys: public: (37, 77); private: 13
- Bob deciphers 07 37 44 44 62

  $(07^{13} \bmod 77)^{17} \bmod 77 = 07$ – H

  $(37^{13} \bmod 77)^{17} \bmod 77 = 04$ – E

  $(44^{13} \bmod 77)^{17} \bmod 77 = 11$ – L

  $(44^{13} \bmod 77)^{17} \bmod 77 = 11$ – L

  $(62^{13} \bmod 77)^{17} \bmod 77 = 14$ – O
- Bob received 07 04 11 11 14 or HELLO

Security Services by RSA

- **Confidentiality**
  - Text enciphered with public key cannot be read by anyone except the owner of the private key
- **Data Integrity**
  - Enciphered letters cannot be changed undetectably without knowing the private key
- Ori**gin Authentication**
  - Only the owner of the private key knows it, so the text enciphered with private key must have been generated by the owner
  - Non-repudiation
    - Message enciphered with private key came from someone who knew it

Attacks against RSA

- Although RSA key is secure enough, susceptible to inference attacks similar to substitution ciphers
- If 1 character per block
    - Statistical attacks to guess plain text
    - Reordering of letters attack to alter message meaning
        - Example: reverse enciphered message of text ON to get NO or text LIVE to get EVIL
        - Padding helps where block identification is sent along
    - If a possible plaintext is known, Forward searching or precomputation attack to guess plaintext sent
    - Example: precompute buy or sell ciphertext and then match with transmission

Use of RSA

- Still widely used in e-commerce
- Secure with
    - Long keys:
        - Typically large public e (65537 – mandated bit NIST)
        - N usually 1024-2048 bits long, p and q usually at least 512 digits
    - p and q not "too close"

Comparison between Symmetric and Asymmetric

- **Symmetric:**
    - Based on shared secret
    - One secret key
    - Less complex computation
    - Easier implementation
    - Faster
    - Harder distribution
    - No non-repudiation
    - Used for general purposes
- **Asymmetric:**
    - Based on personal secret
    - Two keys (one secret and one public)
    - More complex computation
    - More complex implementation
    - Slower
    - Easier distribution
    - Non-repudiation
    - Used for specialized purposes

In-class Problem:

Assume, SpongeBob and Patrick have the following public and private key pairs (in order):
SpongeBob (3, 46)
Patrick (5, 29)

Fill in the blanks with one of these numbers representing keys to answer how would SpongeBob send a message to Patrick such that no one else would be able to read it?

- SpongeBob would use **5** to encrypt it and Patrick would use **29** to decrypt it.
- If Plankton wanted to read the message successfully, will he be able to do it? **NO**
- If Plankton wanted to change the message, will he be able to do it? **YES**
- Would this change be detected? **NO**

Fill in the blanks with one of these numbers representing keys to answer how Patrick would send a message to SpongeBob such that SpongeBob would know it came from Patrick?

- Patrick would use **29** to encrypt it and SpongeBob would use **5** to decrypt it.
- If Plankton wanted to read the message successfully, would he be able to do it? **YES**
- If Plankton wanted to change the message, will he be able to do it? **YES**
- Would this change be detected? **YES**

Fill in the blanks with one of these numbers representing keys to answer how would SpongeBob send a message to Patrick such that no one else would be able to read it AND Patrick would know that it came from SpongeBob?

- SpongeBob would use **46** to encrypt it first and then **5** to encrypt it finally and Patrick would use **29** to decrypt it first and then **3** to decrypt it finally.
- If Plankton wanted to read the message successfully, would he be able to do it? **NO**
- If Plankton wanted to change the message, will he be able to do it? **YES**
- Would this change be detected? **NO**

March 21st – Crypto & Hash

Quiz on Tuesday (3/26) will be over everything after midterm until today's lecture.

- Responsible for Weak properties of DES (slide 67 of Crypto1) until Key Differences (slide 153 of hash)

Why is key distribution easier for public key cryptography than for private key cryptography?

- Private key cryptography requires that only two people can have access to the keys so the problem is sharing the key between both people so that no one else can access it.
- With public key cryptography, anyone can access the public key so it does not matter who sees it. This makes the sharing of the public key easier. Only one person needs access to the private key so there is no problem with sharing it.

Checksum

- Digital checksums are a compact representation of data
- Provides **integrity** service
- Simple checksum uses 1 unit of data to detect errors in the rest
    - Use of ASCII parity bit
        - ASCII has 7 bits; 8th is "parity" (can be 0/1)
            - **Even parity**: parity bot set to get even number of 1 bits
            - **Odd parity**: parity bit set to get odd number of 1 bits
    - Bob receives "10111101" as the bits
        - If sender is using *even parity* the character was received *correctly* (Six 1 bits)
        - If sender is using *odd parity* then the character was received *incorrectly* (Six 1 bits)

Cryptographic Checksum/Hash

- Hash: a unique (fingerprint) representation that cannot be copied
- Functional/method to generate a set of k bits from a set of n bits (where $k \leq n$)
    - For any given message, <u>easy to compute</u> the hash (not vice versa)

        - For a given message, the corresponding hash <u>cannot be predicted</u> (only computed to be seen)
    - Same message will produce the same hash
        - Any change in the message will generate completely different hash (Avalanche effect)
- A.k.a
    - Message digest
    - Message Integrity Code
    - Modification Detection Code

Hash is NOT Encryption

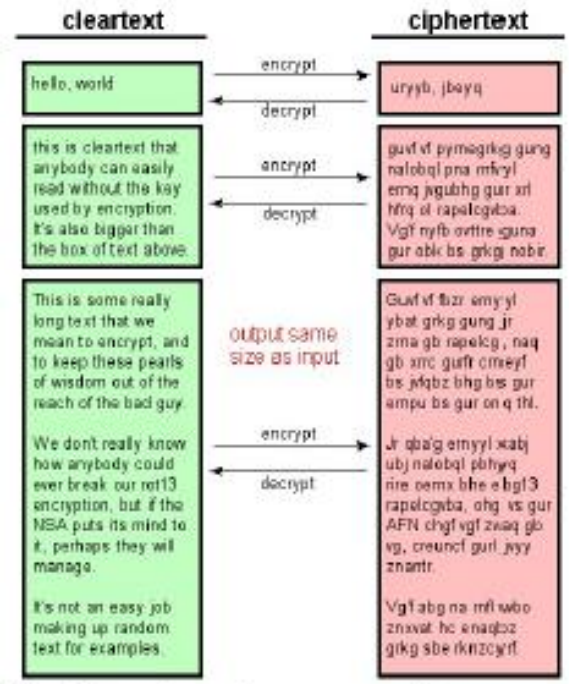


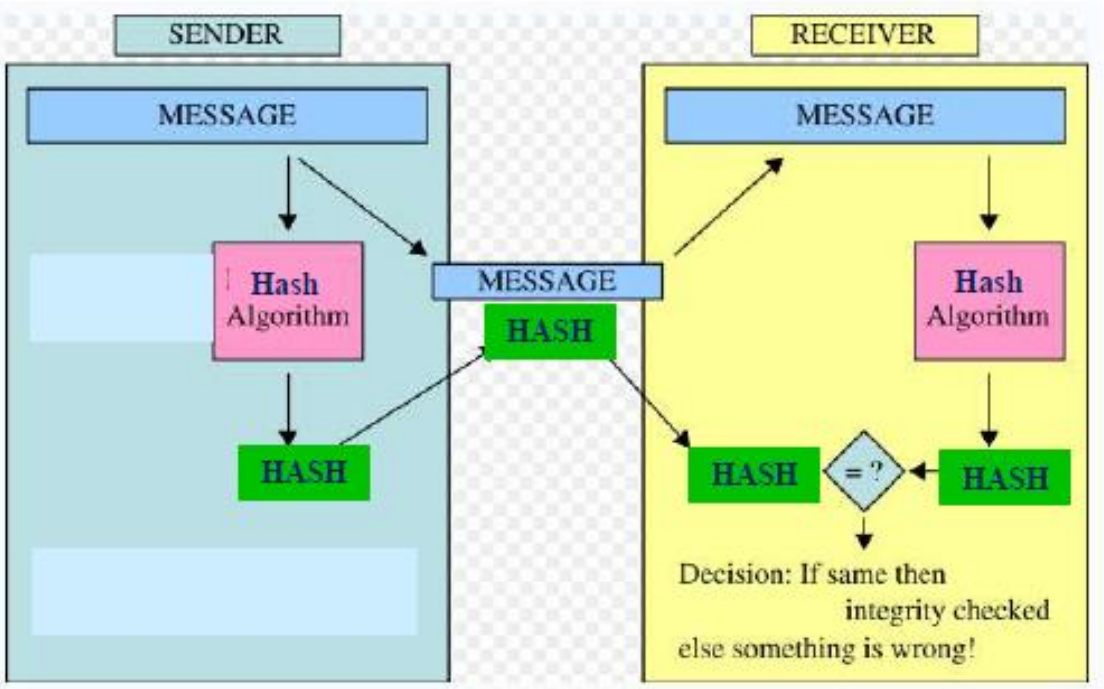
Fig. 1: Encryption – a two-way operation

Fig. 2: Hashing – a one-way operation

Cryptographic Hash Properties

- Impossible to derive original message from digest
- Fixed-length digest regardless of message size
- A digest should be result of the whole message, not a portion of it

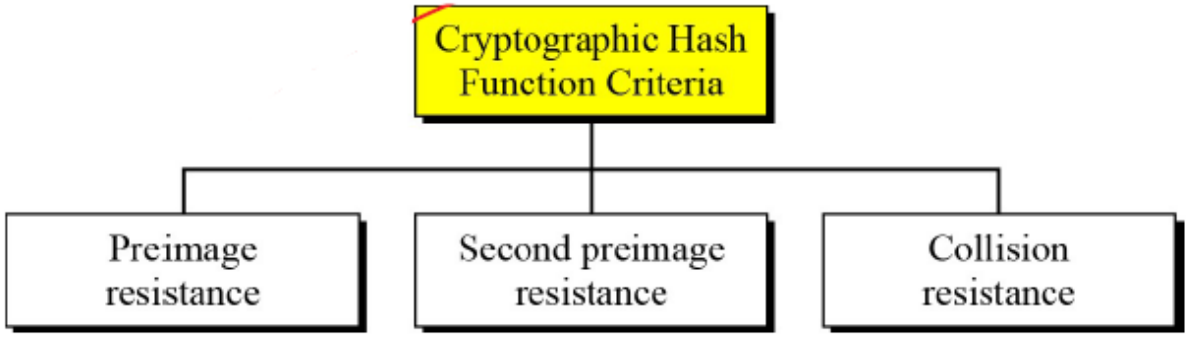
Checksum/Hash



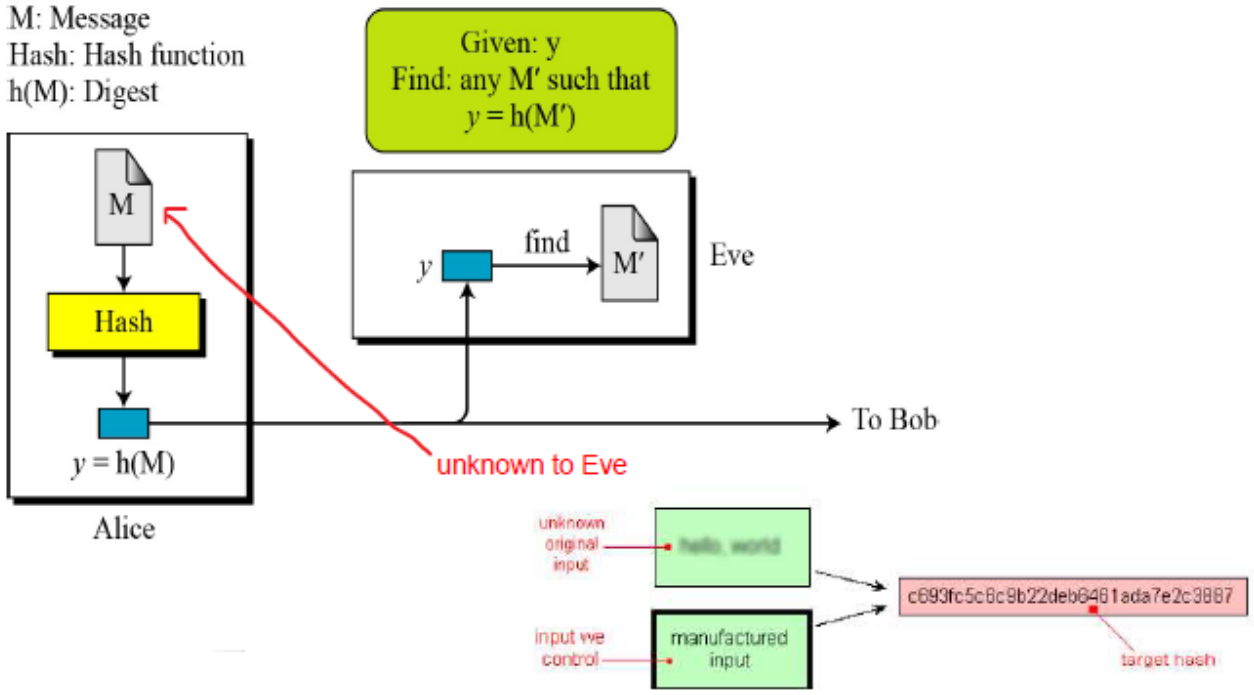
Applications of Hash

- Message (in communication) integrity
- Document/file (in storage) integrity
- Application/system state integrity
- Identifier of entity
- Key generation

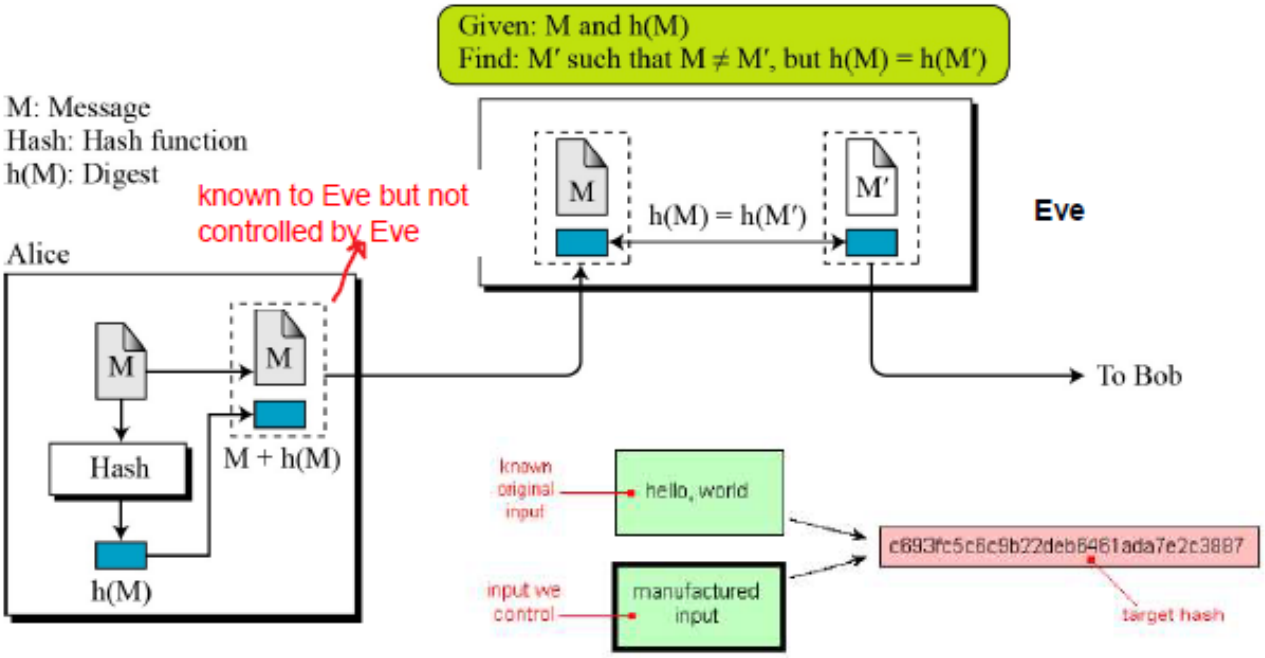
Resistance Criteria of Cryptographic Hash Function



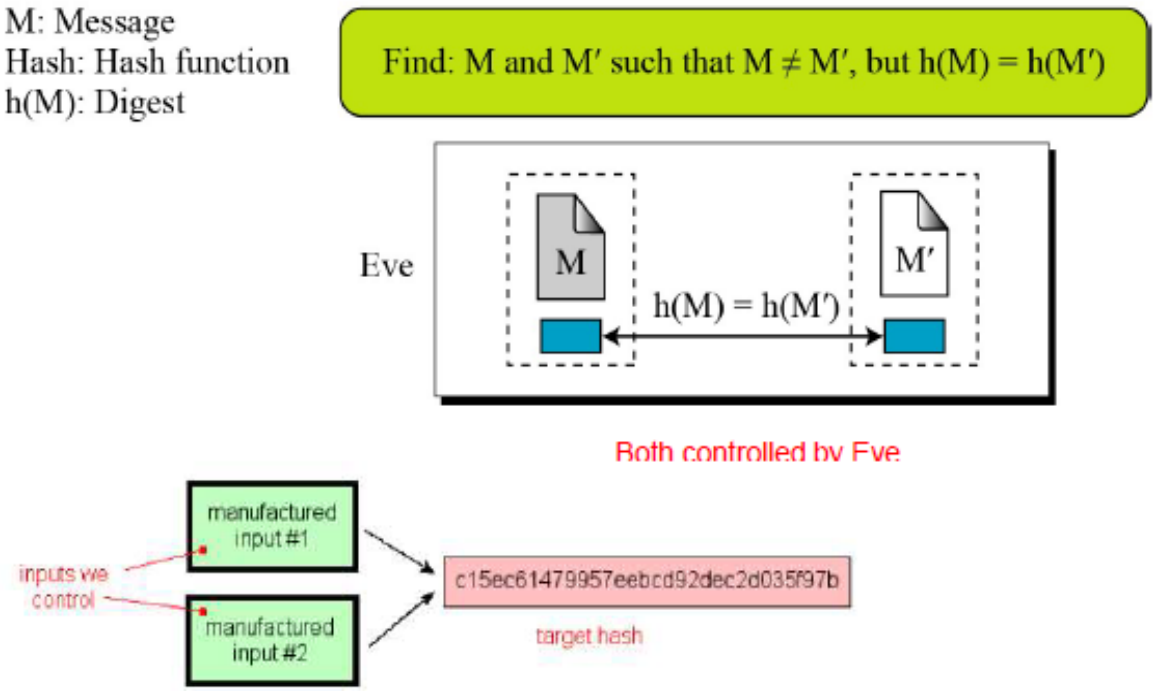
Preimage Resistance



Second Preimage



Collision Resistance



Summary of Hash Resistance

- No other message should produce the same digest
    - Given only hash/digest (**preimage resistance)**
    - Given only hash/digest and original image **(second resistance)**
- No two messages should result in the same hash/digest **(collision resistance)**

Collisions

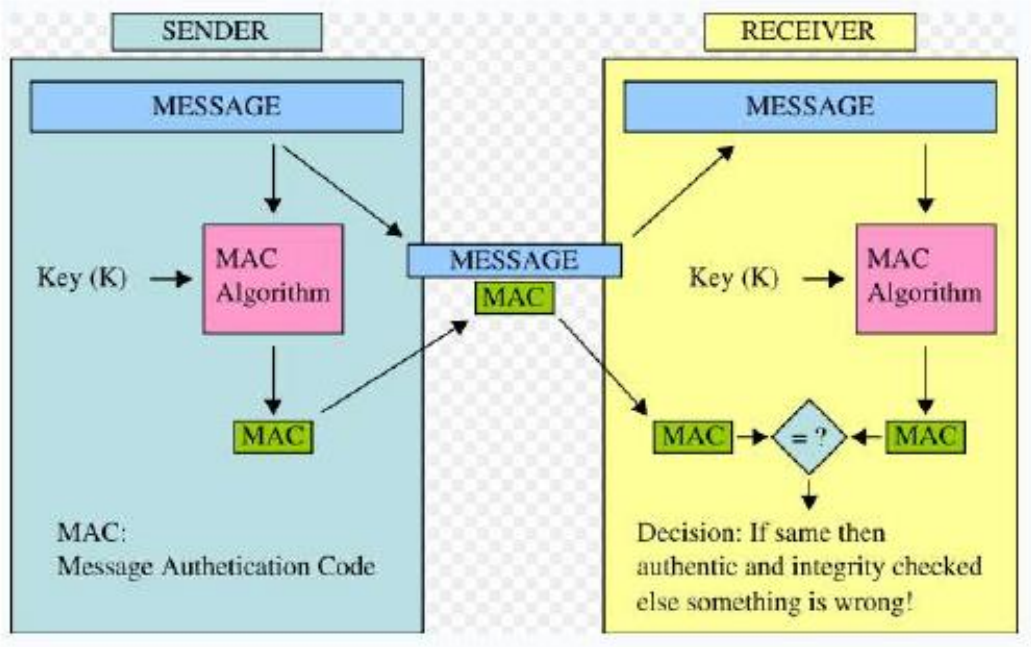
- When two different messages have the same hash, they are called **collisions**
    - If x ≠ x' and h(x) = h(x'), x and x' are a collision
    - **Pigeonhole Principle:** If there are n containers for n + 1 objects, then at least one container will have 2 objects in it
        - Example: if there are 32 files and 8 possible cryptographic checksum values, at least one value corresponds to at least 4 files

- Birthday Attack
    - Used for finding collisions to commit forgery
    - Possible because of **birthday paradox**
        - Probability that at least one pair from a group will have the same birthday
            - The probability increases with pigeonhole principle, for example, this probability will be maximum when there are more than 366 people in the group

Key use in checksums

- **Keyless cryptographic checksum**: requires no cryptographic key
    - MD5 and SHA-1 are best known; others include MD4, HAVAL, and Snefru
    - Susceptible to forgery
- **Keyed cryptographic checksum**: requires cryptographic key
    - Can be used for message authentication (**origin integrity**)
    - Called **Message Authentication Code** (MAC)
        - Not same as *Digital Signatures*
- *Keyed checksum provides more assurance than keyless*

MAC

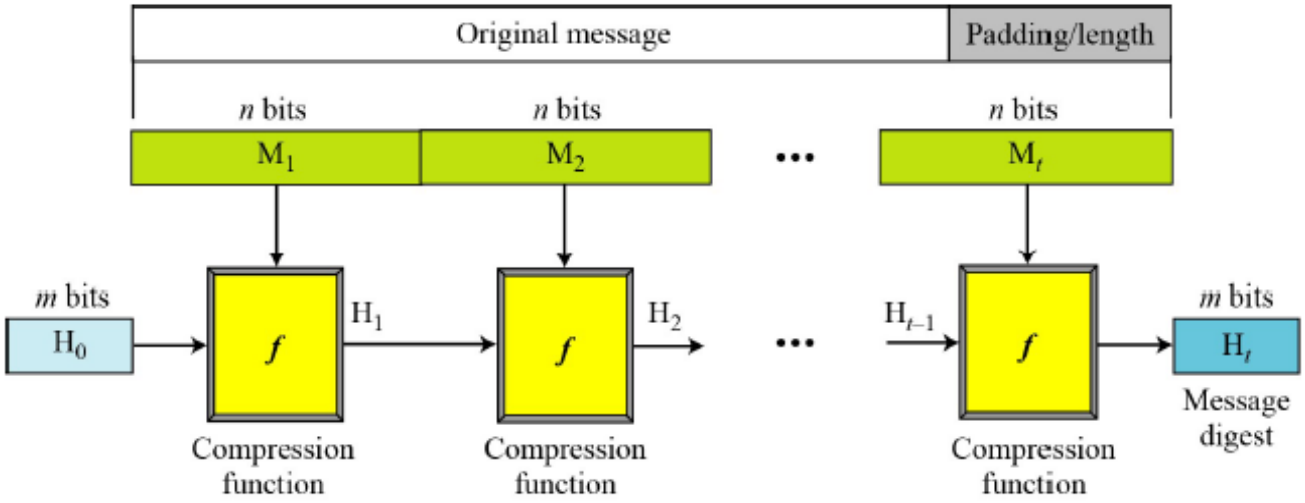


Use of Key in MAC

- Usually the key is concatenated with the message and then hashed to produce MAC
    - Prefixes, postfix or both
- Subject to brute force attack for key, unless size of key makes it difficult
- Strength of MAC depends on strength of hash algorithm used
- To improve security, iterated or nested MACs are used

Iterated Hash with Compression Function

- Merkle-Damgard Scheme



Compression (Hash) Functions

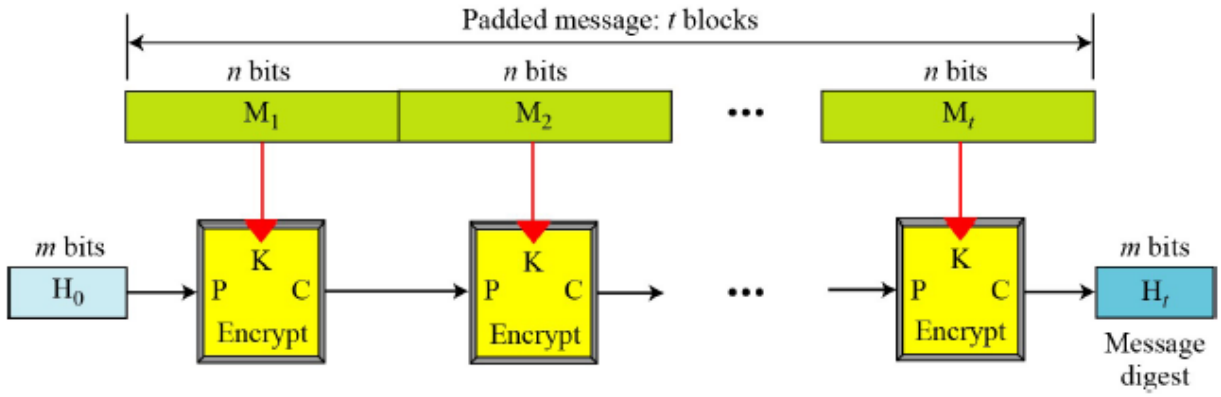
- Generates an m-bit string from a string of n-bit where $m \leq n$
- Two types
    - Made from Scratch
        - MD5, SHA
    - Based on block ciphers
        - Whirlpool

CBC-MAC or CMAC (Hash with block cipher)

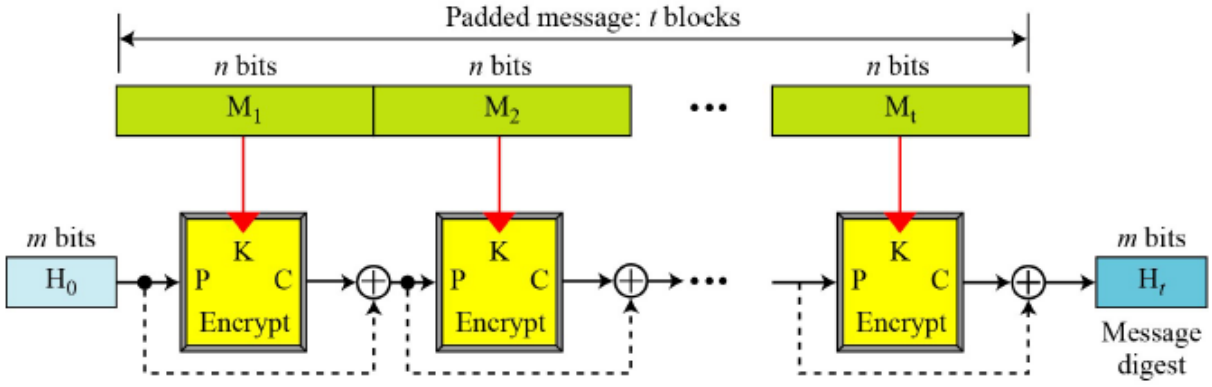
- DES in CBC mode
    - Encipher message, use last n bits as checksum
        - Requires a key to encipher, so it is a keyed cryptographic checksum
        - Integrity with DES

Iterated Hash with Block Ciphers

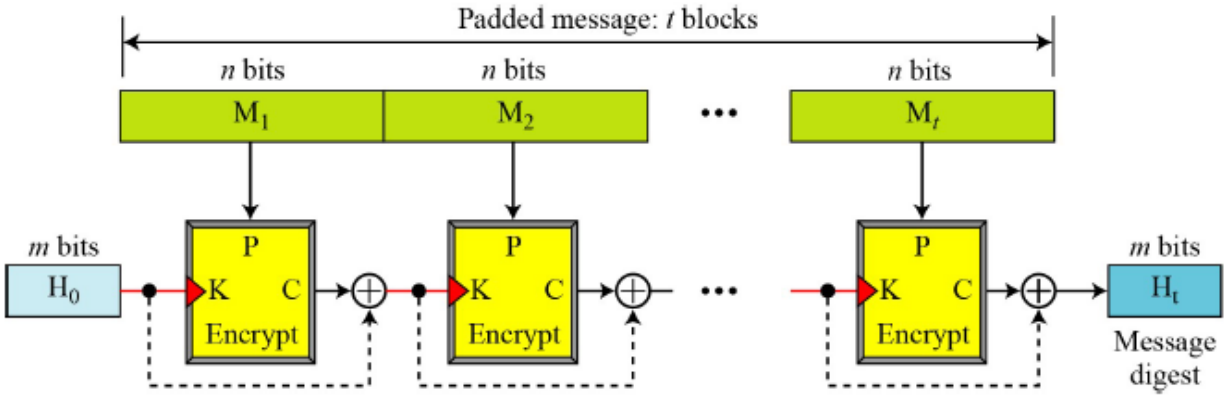
- **Rabin scheme**



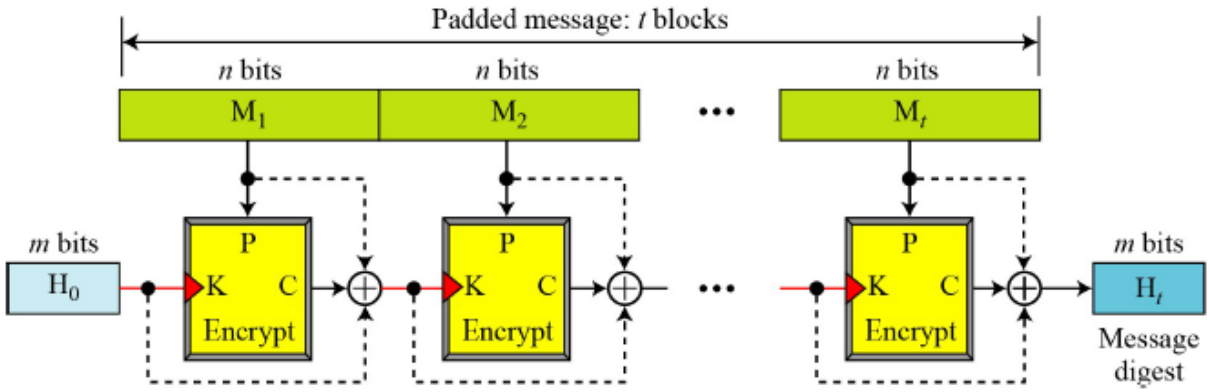
- **Davies-Meyer scheme**



- **Matyas-Meyer-Oseas scheme**



- **Miyaguchi-Preneel scheme**



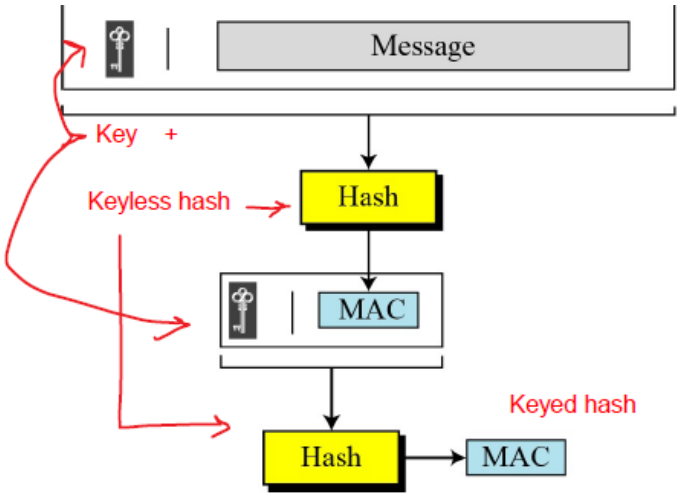
- How are these different? Which is the strongest (look at which has more XOR?
- Questions will be interpreting diagrams that are given.

Nested Hash or HashMAC (HMAC)

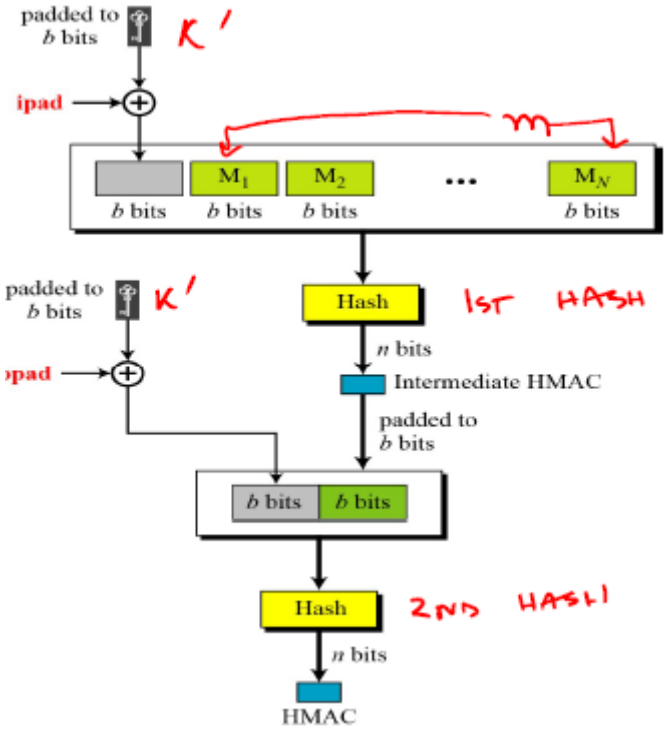
- Purpose
    - Cryptographic algorithms to produce keyed checksums had restriction import/export in many countries BUT keyless checksums don't
    - So, idea is to use key with keyless checksums to produce keyed checksums
- General name for algorithm to *produce keyed checksums* (MAC) *from keyless checksums* (MIC)
    - Example: If hash function is MD5, then the resulting HMAC algorithm is termed HMAC-MD5

- Strength of the HMAC (nested hash/hash-MAC) primarily depends upon
    - Cryptographic strength of the underlying hash function
    - Size and quality of the key
- Not affected by collision problem
- Uses nested checksums

Nested Hash Concept



HMAC



Digital Signature

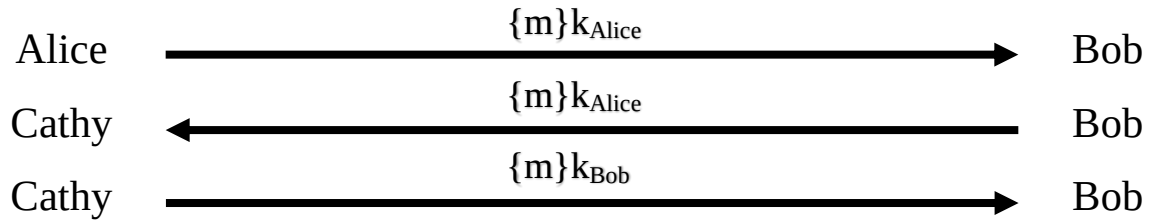
- Construct that authenticates origin of message in a manner ***provable*** *to a disinterested third party* ("judge")
    - Serves as evidence
- Sender cannot deny having sent the message
    - Provides "non-repudiation" service
        - One could claim the cryptographic key was stolen or compromised
- Different from MAC
    - MAC does not offer non-repudiation
    - MAC is always a shared secret key

Example

- Classical: Alice, Bob share key k
    - Alice sends m || {m} k to Bob
    - Is this a digital signature?
        - No, A third party cannot determine whether Alice of Bob generated the message.
        - It is rather a MAC that supports origin integrity

Non-repudiation with Classical/Symmetric Cryptography

- Intervention of trusted third party is required to achieve non-repudiation with classical cryptography
    - Alice, Bob each share keys with TRUSTED 3rd party Cathy (Not with each other)

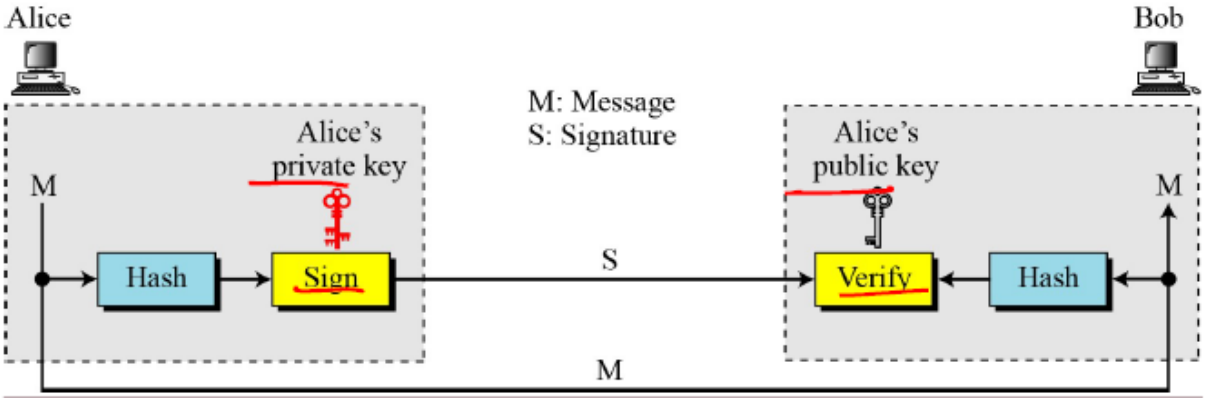


- 3rd party (Cathy) acts as a mediator/representative. During the whole time, the sender and receiver do not share keys, they share keys with Cathy ONLY. This ensures non-repudiation

- To resolve dispute, judge gets $\{m\}k_{Alice}$ and $\{m\}k_{Bob}$, from Bob and Alice respectively, and has Cathy decipher them to check for forgery

Public Key Digital Signatures

- Enciphering message (or hash) with private key produce digital signature



Example

- Alice's keys are d_{Alice}, e_{Alice}
- Alice sends Bob

  $m \parallel \{m\}\ d_{Alice}$
- In case of dispute, judge computes

  $\{\{m\}\ d_{Alice}\}\ e_{Alice}$
  - And if it is m, Alice signed message, she's the only one who knows d_{Alice}

Forgery and Precaution

- Never sign random documents
- Only signing does not help
- Should be both signed (with sender's private key) and enciphered (with receiver's key)
  - Don't encipher and then sign
  - **Sign first and then encipher**

Key Differences

- Differences between MAC and MIC
    - MAC used for authentication, MIC used for integrity
    - MAC requires key, MIC doesn’t
        - Same algorithm on same content = same MIC
        - Same algorithm on same content with different key = different MAC
- Difference between MAC and Digital Signature
    - MAC does not provide non-repudiation like digital signatures because of use of symmetric encryption
- None provide confidentiality (if used in typical way)

March 26th – **Steganography**

Steganography

- Greek: “covered or hidden writing”
- Hidden from unsuspecting user by embedding into message
    - Only intended recipient know of the existence of hidden message
- Differs from cryptography
    - Original message is not disguised (**Covertext**)
    - Secret content is hidden/obscured in Covertext, which becomes **Stegotext**

Use

- Advantage over cryptography
    - *Does not instigate curiosity/suspicion*
- Types of uses
    - Benevolent use
        - Digital watermarking, finger printing
        - **Canary traps** for identifying leaks
    - Malicious use
        - Spying, terrorism, covert communication

Steganography Techniques

- Many exists: Examples
    - Manipulating noisy data
    - Manipulating time/storage
    - Hiding as encryption with some other encrypted data
    - **Chaffing and winnowing**

Chaffing and Winnowing

- Hiding with bogus data (chaffing) and then filtering bogus data (winnowing)
- Unique use of MAC to provide **confidentiality** without encryption of message

Steganography and Countermeasures

- Detection of stegotext with Steganalysis
    - Compare with original
    - Change format before sending
    - Detection tools available (OutGuess…)

March 28th – **Malicious Logic**

Malicious Code/Logic

- Set of instructions that cause site security policy to be violated
    - Usually unintentionally initiated by authorized users
    - Usually assume authorized user's identity
- Example: Key logger

```
int main()
{
   hideConsole();
   while (true)
   {
          Sleep(20); // To make sure this program doesn't steal all
   resources.
          for (int key = 8; key <= 255; key++)
          {
                   if (GetAsyncKeyState(key) == -32767)
                   {
                            processKey(key);
                   }
          }
   }

   return 0;
}
```

Why Malicious Code Triumphs Today

- Increased
    - Connectivity
    - Sophistication of technology
    - Availability of automated tools
- Booming underground economy
- Weakest link (people) is still weakest

Malicious Codes

- Types
    - Trojan Horses
    - Logic/Time Bombs
    - Viruses
    - Worms
    - Rabbits/Bacteria

Trojan Horse

- Program with an **overt** purpose (known to user) and a **covert** purpose (unknown to user)
    - Named by Dan Edwards in Anderson Report

Propagating/Replicating Trojan Horse

- Trojan horse that makes copies of itself
    - Also called propagating Trojan horse
    - Early version of animal game (1975)

Logic Bombs

- A program that performs an action that violates the site security policy when some external event occurs
- Example:
    - Program that deletes company's payroll records when one particular record is deleted

Virus

- Program that inserts itself into one of more files and performs some action
    - Insertion phase
    - Execution phase
    - Incuation/Dormant phase

- Phases also known as:
  - Propagation
  - Mission
    - Trigger
- May be based on conditions
  - Lehigh virus inserted itself into boot file only if boot file not infected

Pseudocode for Virus

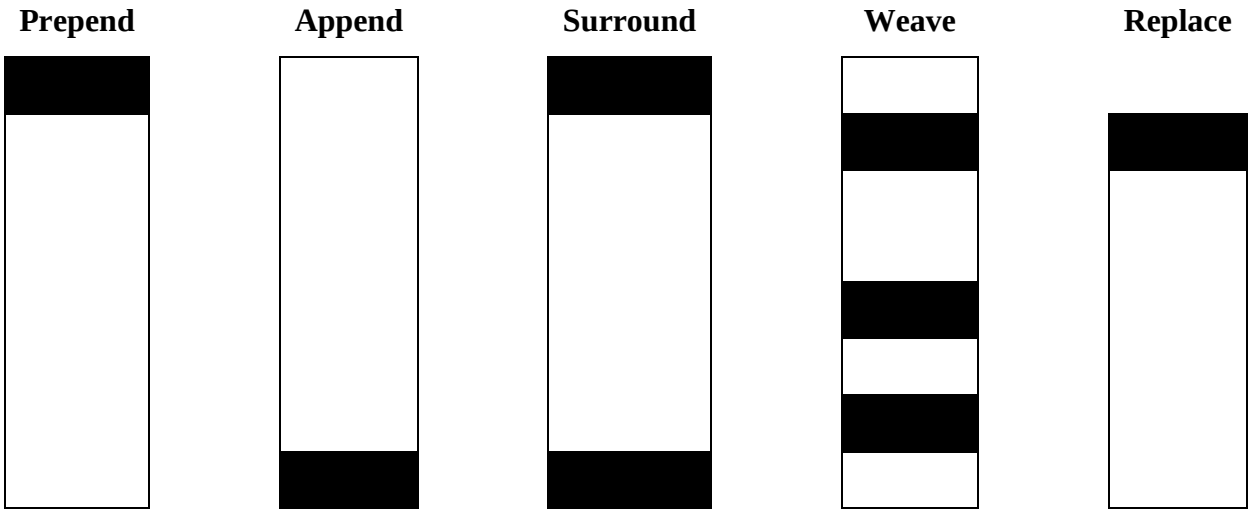
```
beginvirus:
  if spread-condition then begin
    for some set of target files do begin
      if target is not infected then begin
        determine where to place virus instructions
        copy instructions from beginvirus to endvirus
          into target
        alter target to execute added instructions
      end;
    end;
  end;
  perform some action(s)
  goto beginning of infected program
endvirus:
```

Insertion

Execution

Payload

Virus Insertion



Virus: Trojan Horse or Not?

- Yes
    - Overt action = infected program's original actions
    - Covert action = virus' actions (infect, execute)
- No
    - Overt purpose = virus' actions (infect, execute)
    - Covert purpose = none

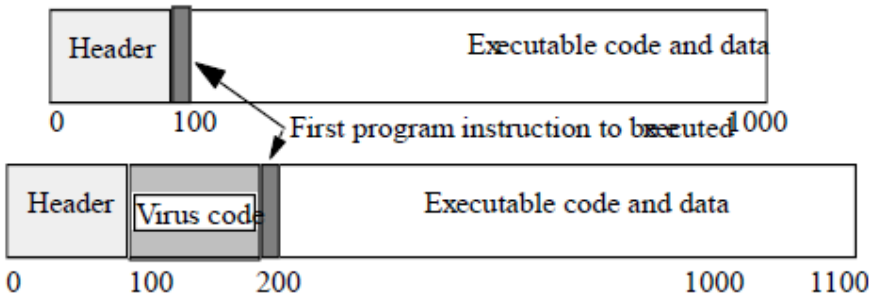
Categories of Viruses

- Based on target platform
    - Boot sector
    - Executable/Parasitic
    - Multipartite (Can spread in multiple ways)
- Based on longevity
    - TSR (Terminate and Stay Resident)
- Based on evasion
    - Cavity
    - Stealth
    - Encrypted
    - Polymorphic
    - Metamorphic
- Based on execution style
    - Binary
    - Macro

Boot Sector Infectors

- A virus that inserts itself into the boot sector of a disk
    - Executed when system first "sees" the disk
- Example
    - Brain/Pakistani virus (1986)

Executable Infectors



- A virus that infects executable programs
  - Can infect .EXE or .COM files
  - May prepend itself or put itself anywhere
  - Example
    - Jerusalem virus (1987)

Multipartite Viruses

- A virus that can infect boot sectors and/or executables
  - There are viruses that infect only files/data
- Typically, two parts:
  - One part boot sector infector
  - Other part executable infector
- Example
  - Ghostball (1989)

TSR Viruses

- TSR is “Terminate and Stay Resident”
- A virus that stays active in memory after the application (or bootstrapping) is completed
  - Fast and slow infectors
- Examples:
  - Brain, Jerusalem viruses
- Counter example (Non-Memory-Resident Virus)
  - Encroacher virus

Cavity Viruses

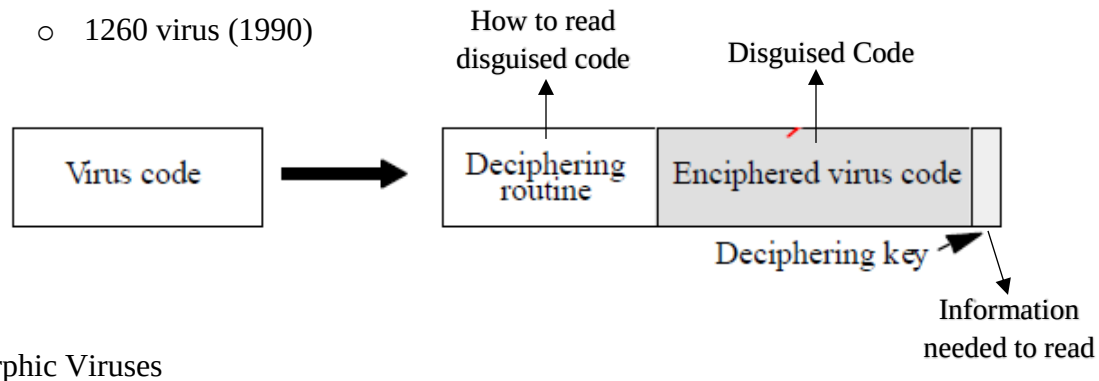
- A virus that does not change the size of the file
    - Overwrites unused areas of executables
- Example:
    - CIH/Chernobyl virus (1998)

Stealth Viruses

- A virus that conceals infection of files
- Can intercept operating system calls to access files
- Example:
    - IDF/4096/4K virus (1989)

Encrypted Viruses

- A virus that is enciphered except for a small deciphering routine
- Example:
    - 1260 virus (1990)



Polymorphic Viruses

- A virus that changes its form each time it inserts itself into another program
- Polymorph at algorithm level
    - Example:
        - Dark Avenger (1988)
- Polymorph at instruction level
    - **Toolkits exist**
        - Trident Polymorphic Engine
        - Mutation Engine

Instruction Level Polymorphic Example

- There are different instructions that have the same effect
    - Add 0 to operand
    - Subtract 0 from operand
    - Exclusive OR 0 with operand
    - No-op
- Polymorphic virus would pick on randomly from these instructions

Metamorphic Viruses

- Completely rewrites itself after each infection
- Behavior change
    - Toolkits exist
        - Metamorphic Engine
- Example:
    - Simile (2002)

Macro Virus

- A virus composed of a sequence of instructions that are interpreted by application rather than executed directly
- Can infect
    - **Executables**
        - Duff's shell virus
    - Or **data files** (a.k.a. Document virus)
        - Lotus 1-2-3 spreadsheet virus
- Independent of machine architecture/platform
    - But their effects may be machine dependent
- Example
    - Melissa (1999)

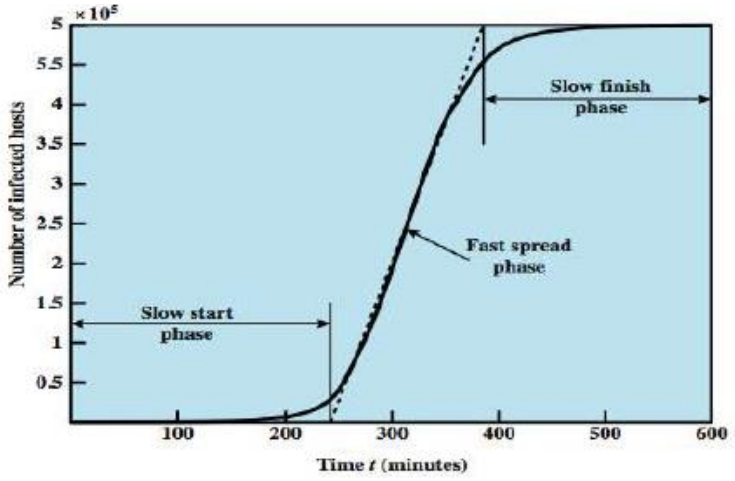
Worms

- A (stand-alone) program that copies itself from one computer to another
    - Does not need host or user intervention
    - Uses network communication medium as transport
        - Email facility, remote access/execution capability, dynamic coding
    - Has phases like a virus
- Origin: Schoch and Hupp (mid ‘70s)
    - To serve the need of distributed computations: animations, broadcast messages
    - Segment (worm) copied into workstation
    - Segment processed data, communicated with worm’s controller
    - Any activity on workstation caused segment to shut down

Morris Worm

- Pioneer worm
- Released by Robert Morris in 1988 through MIT machines
- ~6000 Unix machines infected
- Damage was not “planned”
- He invented the worm and was the first person ever for being prosecuted

Typical Worm Propagation Rate



- The more machines that it is on, the quicker it grows until it runs out of machines

Modern Worm Technology

- Multi-platform
- Multi-exploit
    - Zero-day Exploit
- Multi-tasking
    - Transport vehicle
- Ultrafast spreading with pre-scan
- Dynamic
    - Polymorphic
    - Metamorphic
- Proactive defense

Rabbits/Bacteria

- A program that absorbs all of some class of resources
    - Rabbit is usually a type of **worm**
    - Bacteria is usually a type of **virus**
- Example: for UNIX system, shell commands:

```
while true
do
      mkdir x
      chdir x
done
```

Comparison of Malicious Codes

- All have unauthorized intentions
- **Trojan Horses**
    - Hidden functionality with open legit usage
    - Do not infect
    - Some can reproduce/copy
- **Logic/Time Bomb**
    - Lies dormant until logic/time condition triggers
    - Do not infect

- **Virus**
    - Requires host to reproduce/copy
        - Infection hosts are data or program files
    - Requires user intervention to spread
- **Worm**
    - Independent (Do not require host or user intervention)
    - Automatically spread through network medium

Malware (or not!)

- **Rootkits**
    - Support tool for Hacker
    - Used for seizing control over OS and for concealment
- **Trap Doors/Back Doors**
    - Secret undocumented entry that bypass access control
        - Not always malicious
    - Can be used for unauthorized unauthenticated access
- **Easter Eggs** (Software Based)
    - Secret behavior of code in response to certain stimuli
    - Kind of like a logic bomb

Grayware

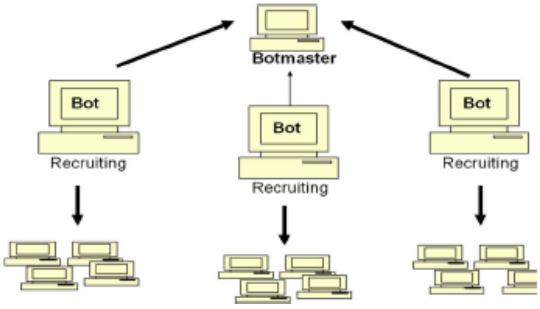
- Annoying
- Undesired
- Negatively affect performance
    - Spyware
        - Profiles users
            - Mostly used for marketing
            - Can be malware when collecting sensitive information
    - Adware
        - Delivers advertisement without user discretion
        - Mostly used for marketing

Scareware

- Software with little or no benefit sold to naïve user with unethical marketing
    - Exploiting people's anxiety, shock, and fear
        - Example: Registry Cleaners

Then there are Bots!

- A botnet is a network of infected end-hosts (bot machines) under the command of a bot master (Bot herder, mothership with human operative behind)
- 1st bot – 1999
- Bots (Zombies, Drones) claimed by:
    - Exploiting vulnerabilities
    - Social Engineering
- Largest since Nov 2012 is Bredolab with 30 million Zombies and 143 control servers



Botnet Characteristics

- Lifecycle
    - Recruit
    - Communicate
    - Execute mission
- Bots commands
    - Update
    - Attack
        - Recruit

Uses of Botnets

- Distributed Denial-of-Service Attacks
- Spamming
- Sniffing Traffic
- Keylogging
- Spreading new malware
- Installing Advertisement Addons and Browser Helper Objects
    - Google AdSense abuse
- Manipulating online polls/games
- Mass identity theft

~not responsible for history of malcode

Defense against Malicious Logic

- **Hard to contain**
    - Uses legitimate means to propagate
- **Hard to detect**
    - Any level of scrutiny can be defeated

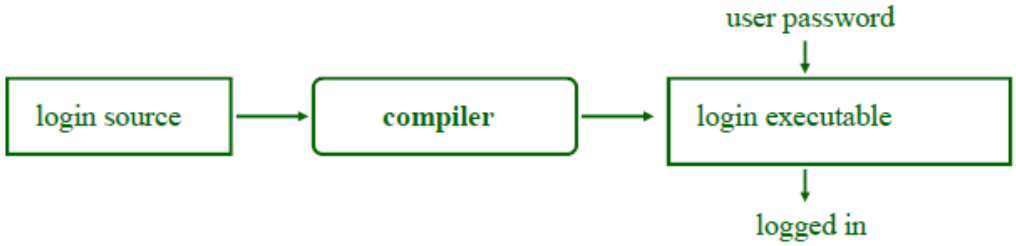
Detecting Trojan Horse

- 1976: Karger and Schell suggested modifying compiler to include Trojan horse that copied itself into specific programs including later versions of the comples
- 1980s: Thompson implements this

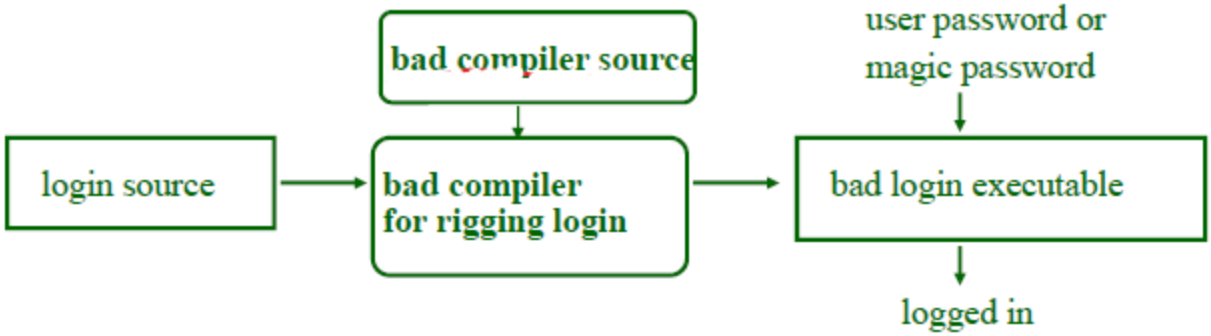
Thompson's Compiler

- Modify *login* source to accept the user's correct password or a fixed password (the same for all users)
    - Code visible in login source code
- Instead, modify the compiler so that when it compiles *login*, *login* accepts the magic password
    - Code visible in compiler source code
- Then modify the compiler again, so when it compiles a new version of itself, the extra code to do the 2nd step is automatically inserted
- Recompile the compiler
    - Doctored compiler in place
- Delete the source containing the modification and put the undoctored source back
    - Doctored compiler still in place

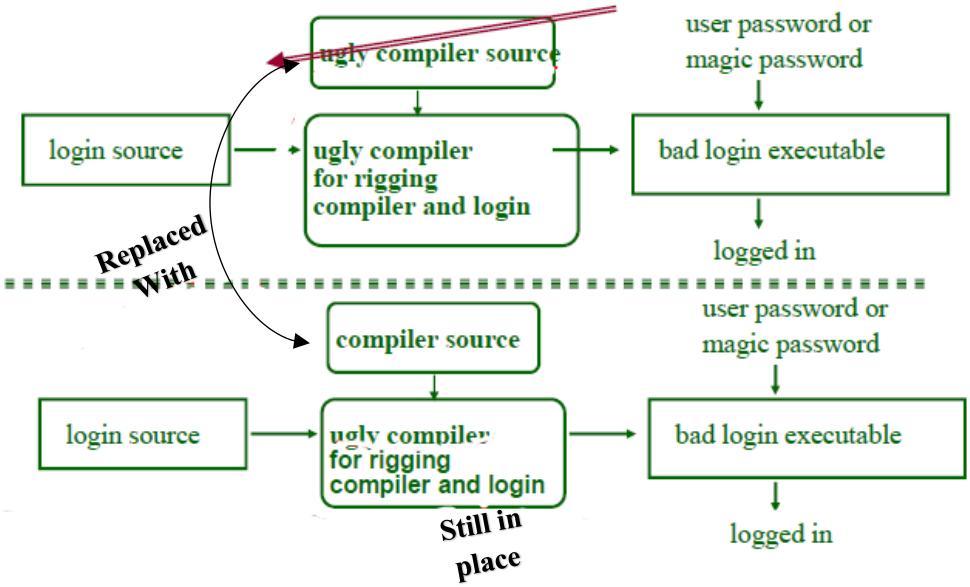
The Login Program



The BAD Compiler



The Ugly Compiler



Point is…

- No amount of source-level verification or scrutiny will protect you from using untrusted code
    - No matter what, you are dependent and have to trust/rely on something
    - You are forced to have assumptions and these may be wrong

Login Program & Bad Compiler

- Has trapdoor and if you use the magic word then you are let in
    - You must program it this way but that means someone might see it…
- Instead of writing it in the code itself, tell the compiler to insert certain lines in the source code that identify the "magic word"
    - Now someone can see that the compiler is doing this and can detect the trapdoor…
- Now, make another compiler source code – the compiler itself has to be created from source code
- You write a program that reminds the compiler to insert code when it is running "login" and check for the magic code then

Still Defend!

- Prevent
- Detect
- Recover

Prevent

- Distinguish between data, instructions
    - Require certifying authority to convert "data" label to "instruction" label
- Limit objects accessible to processes
    - Implement **least privilege principle**
        - Enable reducing rights
- Inhibit sharing
    - Implement **least common mechanism**

- Inhibit single point of failure
    - Implement **separation of privilege**
    - Introduce redundancy and diversity
- Filter suspicious content/activity
    - Use **Firewall and/or Intrusion Prevention Systems** to detect abnormal/malicious content/activity and filter/prevent accordingly
    - Use **watchdogs/guardians** to check file system events or access requests
    - Use **Sandbox** to contain untrusted code
        - Tightly controlled environment with limited resource for untrusted code to run

Detect

- Detect altering of files
    - Compute and verify with **manipulation detection code**
        - Example: Integrity scanner Tripwire's signature block
- Detect actions beyond specifications
    - Look for unknown signatures
        - In code and in AntiVirus software
            - **Bait files**
    - Look for variations of code
        - Verify code sequences with **checksum**
        - Verify with **proof carrying code**
            - Use a theoretical problem to take the program in a producer-consumer approach
            - The client give the program a safety requirement to create a proof that it conforms to the requirement
            - The client takes the program and runs it through another theoretical model to see if the two proofs match to check if something happened to it
    - Look for anomalies in code or activity
        - Analyze characteristics to learn profile

Anti-Virus Programs

- First generation
    - File attribute checking, Simple pattern matching
- Second Generation
    - File integrity checking
    - Encryption checking
- Third Generation
    - Action analysis rather than structure analysis
- Fourth Generation
    - Containment facility with all of the above and more

Recover

- Malcode removal
    - System Restore
- OS reinstallation
- Backup

Countermeasure Success Factors

- Timeliness
- Minimal false positives and false negatives
- Generality
- Resiliency to
    - Invasion techniques
    - Evasion techniques
- Global and local coverage
- Minimal operational overhead
- Transparency

Good Virus

- Viruses that are benevolent/useful
- Called viruses because they need a host to do these things
    - Can conduct virus detection, encryption, and maintenance
        - "Anti-virus" virus, File Compressor virus, Disk Encryptor virus, Maintenance virus
- Cons
    - **Technical**
        - Lack of Control
        - Recognition Difficulty
        - Resource Occupation
    - **Psychological Reasons**
        - Trust problems:
            - Negative Meaning
    - **Ethical and Legal Reasons**
        - Unauthorized Data Modification
            - Copyright and Ownership Problems
        - Possible Misuse

Key Points

- A difficult problem
    - How do you tell wat the user did is *not* what the user intended?
- Trust underlies everything
    - Policy, implementation, certification, usage

April 2nd - Authentication

Authentication

- Binding of an identity to a subject/principle
    - Needed for access control and accountability
        - Real time need
- One or more of the following
    - What entity **knows**
        - Username, password, etc.
    - What entity **has**
        - Phone, private key, device, etc.
    - What entity **is**
        - Physical & behavioral
        - Fingerprints, facial recognition, typing, mouse movement, etc.
    - **Where** entity is
        - **Location**

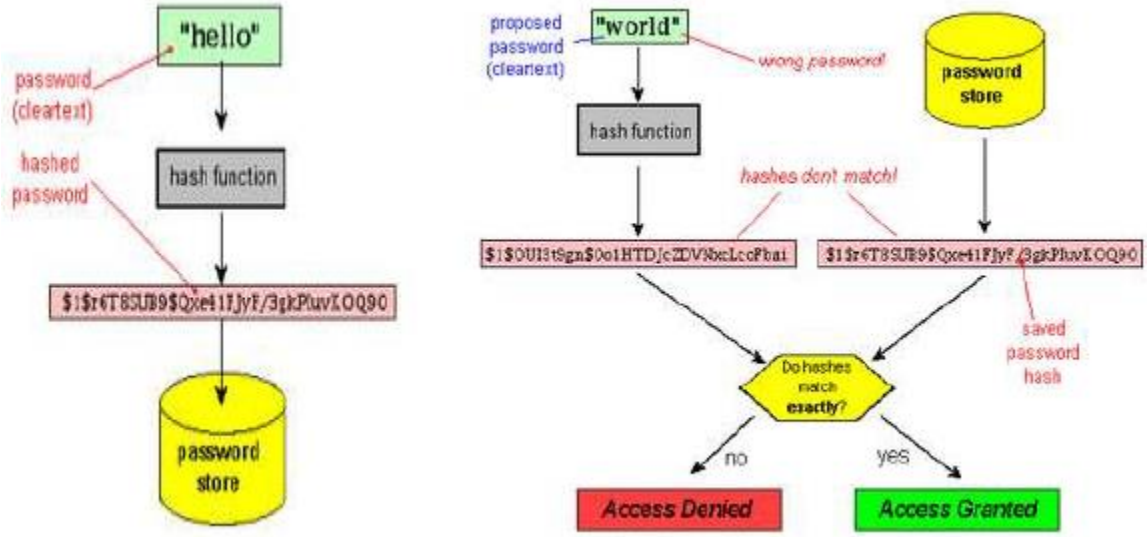
Passwords

- Sequence of characters
    - Example: 10 digits, a string of letters, etc
- Sequence of words
    - Example: pass-phrases
- Algorithms
    - Example: challenge-response

Password Storage

- Store as clear text in file
    - If password file compromised, all passwords revealed
- Hide text in file
    - Encipher file
        - Need to have cipher keys in memory (Goes back to previous problem)
    - Store one-way hashes of passwords
        - If file read, attacker must invert the hash or guess the password

Passwords and Hashes



Authentication System

- System that **obtains** authentication information and **analyzes** to **verify** if identity is associated with entity
- (A, C, F, L, S)
    - A – information the proves identity
    - C – information stored on computer and used to validate authentication information
    - F – complementation function; f: A→ C
    - L – function that verifies identity
    - S – function that enables entity to create, alter information in A or C

Example (A, C, F, L, S) for Password system, with passwords stored as hash

- A – set of strings making up passwords
- C – the hash corresponding to A
- F – the hash function
- L – verify identity by hash matching
- S – function to set/change password

Unix Example

- Unix system standard hash function
  - Hashes password into 11 char string using one of 4096 hash functions
- As authentication system:
  - A = {strings of 8 chars or less}
  - C = {2 char hash id || 11 char hash}
  - F = {4096 different versions of function based on DES}
  - L = {login, su, …}
  - S = {passwd, passwd+, …}

Anatomy of Attacking

- Goal
  - Get authenticated
    - Using a ϵ A
- **Direct approach**
  - Get verified directly through C
    - With unknown f ϵ F and C find an "a" such that f(a) = c ϵ C
- **Indirect approach**
  - Get verified through L (via C)
    - Get to F through l(a) (…succeeds iff f(a)

Preventing Attacks

- How to prevent this:
  - Hide a, f, c
    - Example: UNIX/Linux
      - Hide c's
  - Block access to all l ϵ L or result of l(a)
    - Prevent from using l(a) to guess (trial and error)
    - Example:
      - Preventing *any* logins to an account from a certain network
      - Preventing knowing results of login

Typical Password Attacks

- Try all possible passwords
    - Brute Force Attack
- Try possible passwords using short cuts
    - Optimized Brute Force
- Try probable passwords
    - Dictionary Attack

******QUIZ QUESTIONS AND HOMEWORK OVER THE FOLLOWING INFORMATION******

Brute Force: Finding a, given f and C

- **Brute force**: knowing c, hash each possible a until you find a c that matches (time intensive)
- **Pre-compute**: Store all possible hash or c for all possible a for future look up (space intensive)
- Compromise (time memory trade off)
    - Rainbow table

Time Memory Trade Off Analysis with Rainbow Tables

- Goal: Reduce time and memory requirements for Brute Force Attacks with pre-computed hashes
- Allows faster lookup with manageable memory
- Rainbow table
    - Compact representation of related plaintext **passwords and hashes**

Password	MD5 Hash
123456	e10adc3949ba59abbe56e057f20f883e
password	5f4dcc3b5aa765d61d8327deb882cf99
12345	827ccb0eea8a706c4c34a16891f84e7b
12345678	25d55ad283aa400af464c76d713c07ad
qwerty	d8578edf8458ce06fbc5bb76a58c5ca4
123456789	25f9e794323b453885f5181f1b624d0b
1234	81dc9bdb52d04dc20036dbd8313ed055
baseball	276f8db0b86edaa7fc805516c852c889
dragon	8621ffdbc5698829397d97767ac13db3
football	37b4e2d82900d5e94b8da524fbeb33c0

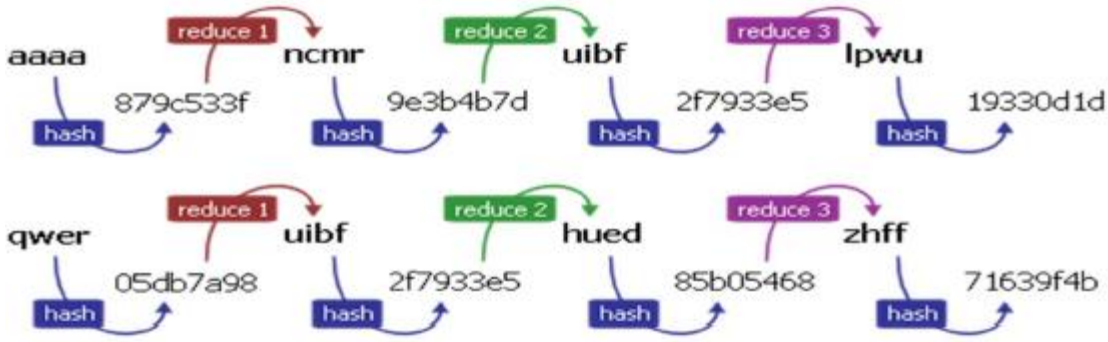
Rainbow Tables

- Stores fraction of total number of password hashes **pre-computed for certain algorithms** (e.g. LM, MD5)
- Table generation uses series of **"reduction" functions** that map to different passwords (also *irreversible* like hashes)

Creating Rainbow Chains

- Start with the password: password
- Next element is reduce(hash(password))
- Next one is reduce(hash(reduce(hash(password))))
- …
- Do this many many times and only store password and last element in chain
    - Last element can be hash or pwd

Creating Rainbow Chains



Rainbow Table

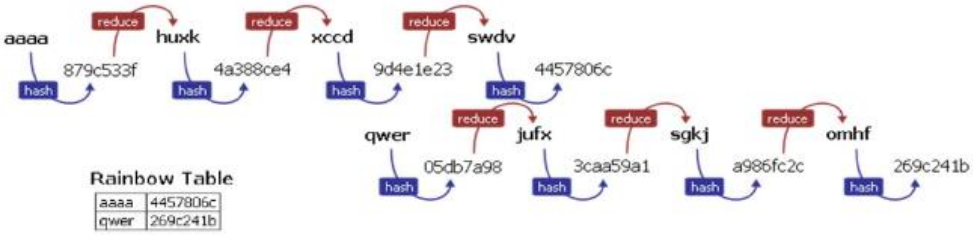
aaaa	19330d1d
qwer	71639f4b

- If you are given the beginning plaintext and you know what hash and reduces are used, you can recreate the chain and come up with the result
- All you need to do is keep track of password and hash – not the middle computations

Using Rainbow Chains

- Look up table entries to match the hash h under test
    - Search hash in 2nd column
        - If found,
            - Take the paired password P in 1st column for the matched entry
            - Apply hash-reduce iterations until the hash h is found
            - The immediate password P' that directly generated hash h is the answer
        - If not found, calculate hash (reduce (h))
    - Repeat (until a match is found)

Using Rainbow Chains Example



- If hash is: 269c241b, entry found in table
    - (a)start with qwer and then the sought password is recomputed and found to be omhf
    - Look up the hash and recreate the chain until you found the password that created the hash
- If hash is: 4a388ce4, entry NOT found in the table
    - Keep reducing and hashing until one entry is found (445806c) in table
    - Repeat as (a) until password is recomputed and found to be huxk
- Not a guaranteed success (probabilistic)

Rainbow Table Defense

- **Collision** – In this case, having collisions is a good thind because that rainbow chains merge with ambiguous results
- **Salting** – Makes pre-computation useless

Dictionary Attacks

- **Off-line (Type 1):**
    - Attacker knows f and c's and repeatedly tries different guesses g ϵ A until and matches a c
    - Examples: crack, john-the-ripper
        - Usually does not have a specific target
        - Needs to have access to f and c
        - Can cause potentially greater damage
- **On-line (Type 2):**
- Attacker has access to functions in L and try guesses g until some l(g) succeeds
- Examples: trying to log in by guessing a password
    - Usually specific target
    - Needs to have access to l(g)
    - Damage limited (unless root)

Password Guessing Steps

- No password
- User ID
- User name
- User names' variations
- User information
- Common word list
- Common patterns
- Short college dictionary
- Complete English word list
- Common non-English language dictionaries
- Short college dictionary with capitalizations
- Complete English with capitalizations and substitutions
- Common non-English language dictionaries with capitalizations and substitutions
- Brute-force

Password guessing is possible because…

- People choose easy to guess passwords
    - Based on account names, user names, computer names, place names
    - Dictionary words
    - Too short, digits only, letters only
    - License plates, acronyms, SSN's
    - Personal characteristics (names, nicknames, job characteristics, etc.)

Countering Password Guessing

- Anderson's Formula
    - P – probability of guessing a password in a specified period of time
    - G – number of guesses tested in 1time unit
    - T – number of time units
    - N – number of possible passwords
    - Then $P \geq TG/N$

Example

- Goal
    - Passwords drawn from a 96-char alphabet
    - Can test 10^4 guesses per second
    - Probability of a success to be 0.5 over a 365 day period
    - What is the minimum password length?
- Solution
    - $N \geq TG/P = (365x24x60x60)x10^4/0.5 = 6.31x10^{11}$
    - Choose s such that $\Sigma^s_{j=0} 96^j \geq N$
    - $A^s = N$ (when A = Alphabet size & S = Password size)
    - So $s \geq 6$, meaning passwords must be at least 6 chars long

April 4th – Authentication & Key Management

Quiz next Thursday over everything after last quiz

- ***Steganography***
- ***Malicious Logic***
- ***Authentication***
- ***Key Management***

Protection of Type (1) Dictionary Attacks: **Salting**

- Goal: mitigate dictionary attacks
- Types:
  - Randomly select a hash function out of multiple choices
  - Concatenate random data with password as input to the hash algorithm
- Increases work for attacker
- Example:
  - Vanilla UNIX method
  - Perturb hash function in one of 4096 ways
    - C = {2 char hash id || 11 char hash}
    - F = {4096 different versions of modified DES}

Exam Question: Give me an example where collision is a good/bad thing.

Protection of Type (2) **Dictionary Attacks**

- Cannot prevent
  - Otherwise, legitimate users cannot log in
- **Deter**
  - *Backoff* – Delay next attempt
  - *Disconnection* – Disconnect after threshold
  - *Disabling* – Disable after threshold
    - Extra caution with administrative accounts!
    - Locked after a certain amount of attempts (forced to use 2FV)

- **Deflect**
    - *Jailing* – Allow with restriction
- Other Defenses
    - Against password cracking
        - Proactive password checking
    - Against password replay
        - Password aging
        - Challenge response

Proactive Password Checking

- Analyze proposed password for "*goodness*"
- Criteria
    - Always invoked
    - Easy to set up and integrate into password selection system
    - Detect, reject bad passwords for an appropriate definition of "bad"
        - Discriminate per-user, per-site basis
        - Conduct pattern matching on words
        - Execute subprograms and use results
            - For example, Spell checker

Example: passwd+

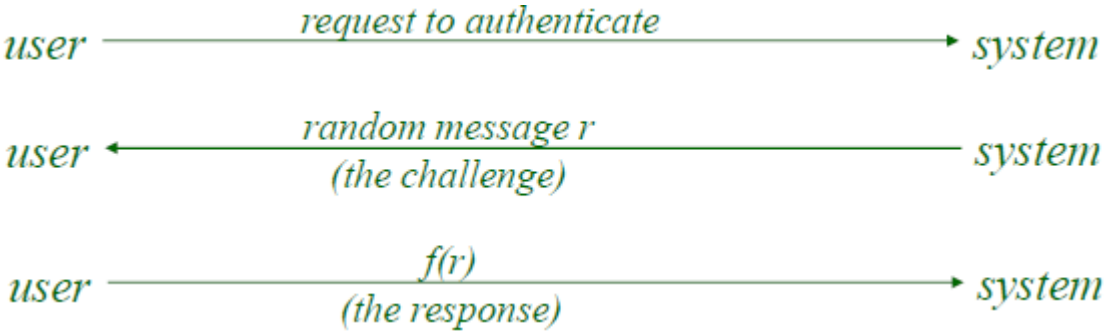
- Provides mechanism for proactive checking
    - Test length("$p) < 6
        - If password under 6 characters, reject it
    - Test infile("/user/dict/words", "$p")
        - If password in file/usr/dict/words, reject it
    - Test !inprog("spell", "$p", "$p")
        - If password not in the output from program spell, given the password as input, reject it (because it's a properly spelled word)

Password Aging

- A requirement to force users to change passwords after some time has expired
    - How often should change occur?
        - Depends on policy, practice and mechanism used
    - How do you force users not to re-use passwords?
        - Record previous passwords
        - Give users time to think of good passwords
        - Warn them of expiration days in advance

Challenge-Response

- User, system share a secret function f (f maybe a secret function or a known function with unknown parameters, such as a cryptographic key)



- If system's computation of f(r) matches with user sent f(r), authentication success

Pass Algorithms

- Challenge-response with the function f itself a secret
    - Example:
        - Challenge is a random string of characters such as "abcdefg", "ageksido"
        - Response is some function of that string such as "bdf" and "gkio"

One-Time Passwords

- Ultimate password aging
- Password that can be used exactly once
    - After use, it is immediately invalidated
- Can also be used in challenge-response manner

    - Challenge is number of authentications identifying the authentication session; response is the password for that particular number
- Problems
    - Generation of good random passwords
    - Synchronization of user, system

One-time Password Scheme: S/Key

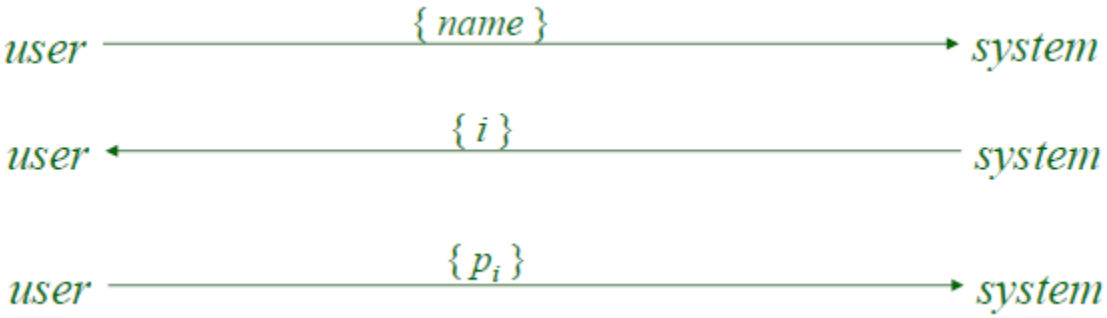
- Based on the idea of Lamport
- Used in Unix-like operating systems
- h one-way hash function (MD5 or SHA-1, for example)
- User chooses initial seed k
- System calculates:

  $h(k) = k_1$, $h(k_1) = k_2$, …, $h(k_{n-1}) = k_n$
- Passwords are used in reverse order:

  P_n, P_{n-1}, … P_1
- System only stores last password (P_n)
- User is given the list of passwords in reverse order and starts with the last one (P_{n-1})

S/Key Protocol – Another View

- System stores maximum number of authentications n, number of next authentications I, last correctly supplied password P_{i-1}



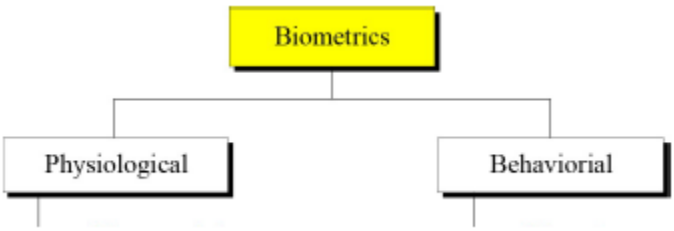
- System computes $h(P_i) = P_{i-1}$. If it matches with what is stored, system replaces P_{i-1} with P_i and updates i

Hardware Supported Authentication

- Work entity has:
    - **Token-based**
        - User owned device used to compute response to challenge sent by computer
            - Computer knows what response to expect
            - May encipher or hash challenge with user supplied PIN
    - **Temporally-based**
        - Every minute (or so) a different number is shown in user owned device
            - Computer knows what number to expect when:
        - User enters number and fixed password
        - Example: RSA SecureID

Biometrics

- **What entity is**
- Automated measurement of biological, behavioral features that identify a person by comparison



Characteristics

- **Fingerprints**: optical or electrical techniques
    - Maps fingerprint into a graph, then compare with database
- **Voices**: speaker verification or recognition
    - Uses statistical/AL techniques to test hypothesis that speaker is who is claimed
    - Checks content of answers
- **Eyes**:
    - Patterns in irises unique

- **Faces**:
    - Image, or specific characteristics like distance from nose to chin
- **Keystroke and mouse dynamics**:
    - Keystroke intervals, pressure, duration of stroke, where key is struck, mouse movements

Cautions

- These can be *fooled*!
    - Assumes biometric device accurate in the environment it is being used in
    - Transmission of data to validator is tamperproof, correct
- These can be *erroneous*!

Location

- **Where entity is**
- Validate identity by verifying location
    - Where you are and what you have
    - Requires special-purpose hardware to locate user
        - GPS (Global Positioning System) provide location signature of user (signature includes time and place)
            - Computer gets location signature for user from GPS
        - User uses LSS (Location Signature Sensor) to get location signature for his/herself
            - Send to computer
        - If match, validated
        - Problem if users' location signature generating device is stolen or compromised
            - Unless policy in place to check further

Multiple Methods

- Authentication methods can be combined
    - **Something you know + something you possess**
        - Smart cards in combination with passwords
    - **Something you are + something you know**
        - Biometrics in combination with passwords
    - **Something you possess + somewhere you are**
        - LSS in combination with location

Good Password Practice

- Use "strong" password
    - Long passwords (length at least 8)
    - At least 1 digit, 1 letter, 1 punctuation symbol, 1 control character
    - Be creative
- Change password
- Don't reuse password for different systems
- Don't write them down
- Don't tell anyone
    - Social Engineering

Social Engineering

- Art of deception
- "Using manipulation, influence, and deception to get a person, a trusted insider within an organization, to comply with a request, and the request is usually to release information or to perform some sort of action item that benefits the attacker."

Techniques

- Dumpster Diving – carelessness
    - Using carelessness in disposing of sensitive information
- Pretexting

    - Using invented scenario (pre-text) to persuade a target to release sensitive information or perform an action that violates privacy
- Phishing
    - Email appearing to come from a legitimate business and resonding to it may result in disclosing sensitive information through a fraudulent web site
    - Vishing (VOIP), Smishing (SMS), Spear Phishing, Whaling
- Gimmes
    - Exploiting curiosity/carelessness to deliver malware
- Quid pro Quo – fear
    - Offering technical help/goods for information
- Shoulder surfing
- Smoking Zone
- Piggybacking - helpfulness
- Reverse Social Engineering
    - Sabotage + Advertising + Assisting

Dealing with Social Engineering

- Easier for attackers to use than technical means
- Very hard to prevent as it exploits human characteristics as:
    - Carelessness, helpfulness, helplessness, comfort zone, fear, curiosity, greed, etc.
- No single solution
    - Need awareness/education
    - Needs training
    - Needs policy placement and enforcement
    - Risk management and contingency plans
    - Legal actions
- "You could spend a fortune purchasing technology and services… and your network infrastructure could still remain vulnerable to old-fashioned manipulation." Mitnick

Key Management

- Distribution of cryptographic keys
- Mechanisms to bind an identity to a key
- Generation, maintenance, and revocation of keys

Notation

- X → Y: {Z||W} $K_{X,Y}$
    - X sends Y the message produced by concatenating Z and W enciphered by key $k_{X,Y}$, which is shared by users X and Y
- A→T:{Z} k_A || {W} $k_{A,T}$
    - A sends T the message produced

Types of keys

- **Interchange key**
    - Vouches for a user/principal
    - Associated with principal(s) in communication
        - Principle is user communicating
        - In case of classical cryptography
            - Interchange key is shared secret key
        - In case of public key cryptography
            - Interchange key is public, private pair
- **Session key**
    - Associated with a single communication
    - Discarded afterwards
    - Limits amount of traffic enciphered with single key
        - Used per communication session
    - Helps with
        - Forward search/precomputation attack
        - Sniffing attack
        - Replay attack

Need for Key Exchange Protocol

- Goal: Alice, Bob need to get shared session key to communicate messages
    - Session key cannot be sent in clear
    - Session key can be sent enciphered with interchange key
- Classical and public key exchange protocols are different

Classical (Session) Key Exchange

- Assume trusted third party, Cathy
    - Alice and Cathy share secret key k_A
    - Bob and Cathy share secret key k_B
- Use interchange keys to exchange shared session key k_s

Concern

- Replay attack
    - Eve intercepts message from Alice to Bob, later replays it; Bob may think he's talking to Alice but isn't
- Protocols must provide defense against replay

****Look at the PDFs and book for the 3 protocols before next class****

Emily Lofts

35.5

Class Quiz 3 **Date: 03/23/19**

CSC 4575/5575

Answer All. Time: 40 minutes Total points: 55

1. Pair left column with ALL that applies on right columns by writing the appropriate right column numbers at the beginning of the left column words [13].

1 MIC	1. Data Integrity
3 MAC	2. Non-repudiation
2 Digital Signatures	3. Origin Integrity

4 MIC	~~4.~~ Keyed Asymmetric
5 MAC	~~5.~~ Keyed Symmetric
6 Digital Signatures	~~6.~~ Keyless

-6.5

9 DES	~~7.~~ Non-Repudiation
8 DES-MAC	~~8.~~ Confidentiality
7 DES with secret shared with trusted 3rd party	~~9.~~ Integrity

DES 11 12	~~10.~~ Shared secret
RSA 10 13	~~11.~~ Personal secret
	~~12.~~ 2 keys
	~~13.~~ 1 key

Weakness 14	~~14.~~ Complementation property
Strength 16 17 15	~~15.~~ Self-healing property
	~~16.~~ Avalanche effect
	~~17.~~ Completeness effect

18 20

Computationally feasible 18	18. One way function
Computationally infeasible 19 20	19. Inverse of one way function
	20. One way function with trap door

Preimage Resistance 21	~~21.~~ Given only hash/digest, no other message should produce the same digest.
Secondary Preimage Resistance 22	~~22.~~ Given only hash/digest and original message, no other message should produce the same digest
23 Collision Resistance	~~23.~~ No two messages should result in the same hash/digest

2. What is the totient function of 24? [2]

$\phi(24) = ?$ $\phi = 8$

Show how you came up with your answer.

$24 = 2 \times 12 = 3 \times 8 = 4 \times 6$

1 ~~2~~ ~~3~~ ~~4~~ 5 ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~ 11 ~~12~~

13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~ 19 ~~20~~ ~~21~~ ~~22~~ 23

3. With p=22 and q=4, if private key is 5 and constant used is 3, then use RSA equation to calculate the public key. [3] Show your work (with equation and values) that leads to the answer. $ed \bmod \phi n = 1$ $n = pq$

public key = 218

$k = \frac{ed-1}{\phi n} = \frac{ed-1}{(p-1)(q-1)}$

-0.5

$3 = \frac{5x-1}{(22-1)(4-1)} = \frac{5x-1}{(21)(3)} = \frac{5x-1}{63} \rightarrow 63 \cdot 3 = 5x - 1$

$= 189 + 1 = 5x$

$190 = 5x$

?

4. "Sign first and then encipher" – what does it mean? [1]

You want to encipher with your private key and then encipher with the receiver's public key

5. What is a collision? Is collision good or bad for attacker (from what we have learned so far in class)? Justify your answer [2]

When two plaintexts create the same hash

- I would think this would be bad for the attacker because it would be harder for them to tell which plaintext actually matches the hash

can be used to commit forgery (good for attacker)

Okay

6. What is HMAC or Hash-MAC? Why do we use it? [2]

converting plaintext to a digest

- using keyless hash algorithms to make keyed algorithms - it is a nested hash

7. Assume that sender and receiver are using ODD parity for simple checksum. In that case, were these data corrupted in transmission or not? Answer yes or no [2].

10110100
yes, there are an even number of 1s (4)

11010111
yes, there are an even number of 1s (6)

8. What is digital signature? Can you create one with [2]:

proof of sender

a) symmetric cryptography? If so, how?

yes, you would need a trusted 3rd party

b) asymmetric cryptography? If so, how?

yes, you would use your private key

9. What is the difference between iterated hash and nested hash? [2]

w/ iterated hash, the message is split up & passed through block ciphers
- nested hash passes the whole message through a hash function

10. Why is RSA called an exponentiation cipher? [1]

uses exponentiation for encryption & decryption

11. Why in RSA, is it not trivial to derive d, knowing both e and n, where d,e and n have usual meaning? [2]

you can use the equation $ed \bmod \Phi(n) = 1$ and solve for d

- even if you know n, you won't know p & q to solve equation

12. List 1 advantage & 1 disadvantage for EACH: Symmetric and Asymmetric cryptography [4]

Symmetric
- simpler implementation
- difficult to share keys

asymmetric
- non repudiation
- slower

13. What is the "meet in the middle attack"? How does it conduct time-memory trade-off? [2]

-1.5

begin encrypting the message on one end while decrypting on the other end
- applies to double DES, known plaintext attack where you take plaintext & brute force to find K_2 w/ K_1 to get middle text to find match
↳ time-memory tradeoff: you save time but you are sacrificing space b/c you don't have to brute force the whole chains

14. What is the "forward search" attack? How can it be prevented? [2]

~~brute force attack~~

15. Short answer (no explanation needed) [1.5]:

-0.5

a) Which of the cipher modes process the blocks of the message independently?

EBC & CTR

b) Which of the cipher modes will generate different cipher text blocks for the same blocks of the plain text message with the same key?

CTR & EBC

c) Which of the cipher modes can be parallelized for decryption?

CBC (all but OBF)

16. With three employees (SpongeBob, Squidward and Pearl Krabs), how many keys would owner Mr. Krabs minimally need to communicate secret instructions to each of them individually? **Show** how you derived your answer for each case:

-0.5

a) using symmetric key cryptography [1]

$\frac{n(n-1)}{2} = \frac{4(3)}{2} = 6$

~~3~~ 3 for one to communicate w/ all

-1

b) using asymmetric key cryptography [1]

~~n=4 (as many keys as people)~~

2 for one to communicate w/ all

public & private

17. Assume, Sherlock and Watson have the following public and private key pairs (in order): [4.5]

Sherlock (2,11) (public, private)
Watson (9,50)

Fill in the blanks with one of these numbers representing keys to answer how would Watson send a message to Sherlock such that no one else would be able to read it AND Sherlock would know that it came from Watson?

- Watson would use 50 to encrypt it first and then 2 to encrypt it finally and Sherlock would use 11 to decrypt it first and then 9 to decrypt it finally.
- If Moriarty wanted to read the message successfully, will he be able to do it? YES/(NO)
- If Moriarty wanted to change the message, will he be able to do it? (YES)/NO
- Would this change be detected? (YES)/NO

18. True or False: T or F [7]

F 1. Cryptographic checksums are non-reversible. T
I 2. Keyless checksum provides less assurance than keyed ones.
I 3. Same MAC algorithm on same content with different key = different hash
F 4. Self-healing property is observable in DES used in direct mode. -2
F 5. RSA encryption (1 character per block) is susceptible to statistical attacks. T
F 6. It is easier for an attacker to find a collision without any message/document given.
T 7. I have been working hard on my term project.

April 9th – Key Management

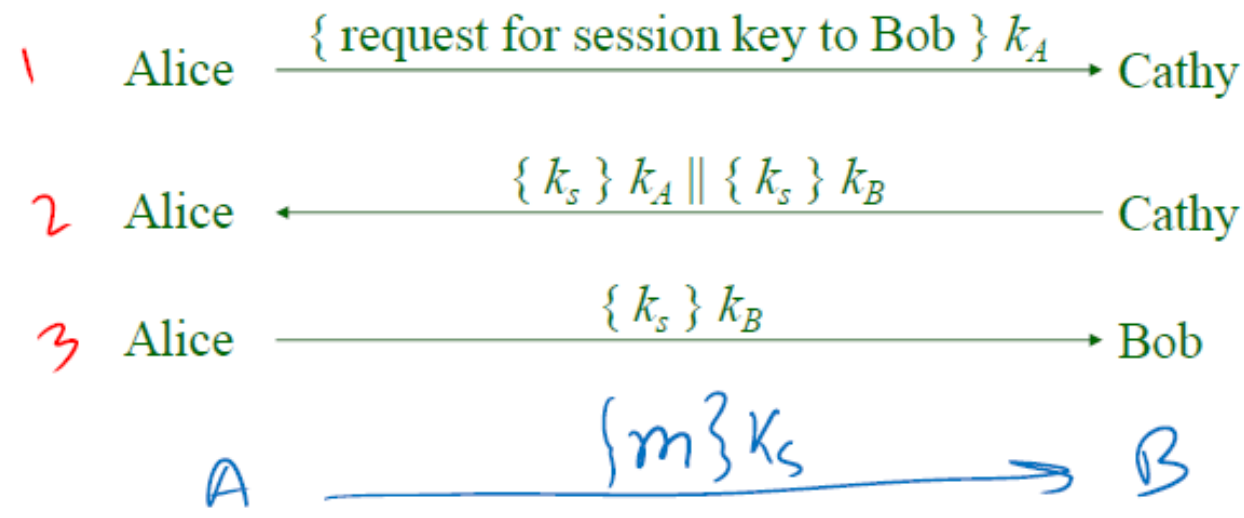
Pre-lecture Discussion

- Shared secret = interchange key
- Why do we need a session key?
    - We want to reduce the attack scope – stops replay & forward search attack (password cracking)
- How do we distribute session keys?
    - Use shared secret that you already have to encrypt the session key and exchange
- How do we share the interchange key?
    - You need Cathy…

Classical (Session) Key Exchange

- Assume trusted third party, Cathy
    - Alice and Cathy share secret key k_a
    - Bob and Cathy share secret key k_b
- Use interchange keys (k_a and k_b) to exchange shared session key k_s

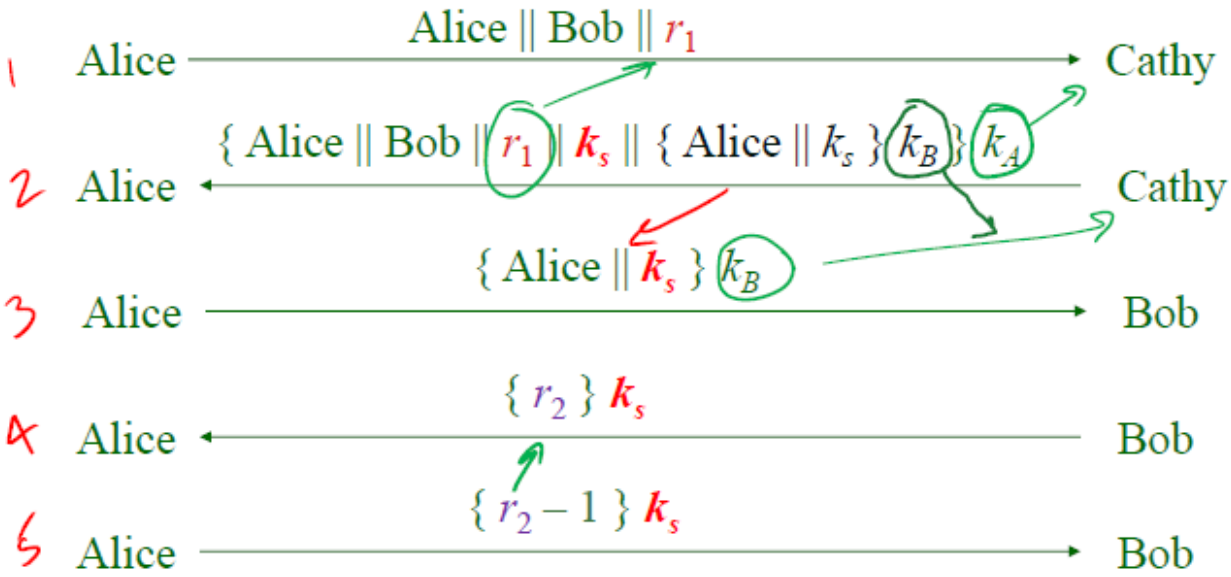
Simple Protocol



Problem with Simple Protocol

- Eve *replays* the messages
    - Eve intercepts the message from Alice to Bob, later replays is; Bob may think he's talking to Alice but he isn't
- Protocols must provide defense against replay
    - To solve this problem, we need Needham-Schroeder Protocol

Needham-Schroeder Protocol



- Bob gets an envelope and sees that it is encrypted with a secret key with Cathy
- He gets the session key from Alice
- Bob comes up with another nonce and sends the message to Alice
- How does Alice know this message came from Bob?
    - It is encrypted with the session key
- Bob asks Alice a question and Alice answers it; Bob is challenging Alice… challenge-response authentication.
- At the end, both Alice and Bob are convinced they are talking to each other through Cathy and that they each have the session key
- **You will not be asked to remember the protocol, but you will be asked questions about it.**

Assumptions

- r_1 and r_2 are two numbers generated randomly that are unique for each protocol exchange
- Both Alice and Bob *trust* Cathy
- Secret keys (k_a and k_b) *are secret*!

Argument: Alice talking to Cathy

- Second message
    - Enciphered using key only she, Cathy, knows
        - Cathy must be the one who enciphered it
    - Response to first message
        - As r_1 in it matches r_1 in the first message

Argument: Alice talking to Bob

- Third and Fourth messages
    - Alice knows only Bob can read it
        - As only Bob can derive session key from message
    - Any messages enciphered with that key are from Bob

Argument: Bob talking to Cathy

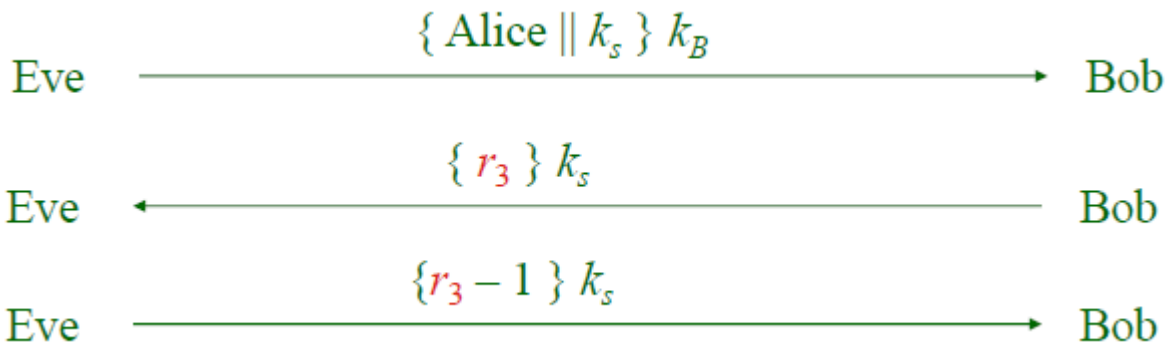
- Third message
    - Enciphered using key only she, Cathy, knows
    - Names Alice, session key
        - Cathy provided session key, says Alice is other party

Argument: Bob talking to Alice

- Fourth message and Fifth message
    - Uses session key and nonce to determine if it is replay from Eve
        - If not, Alice will respond correctly in fifth message
        - If so, Eve can't decipher r_2 and so can't respond, or responds incorrectly

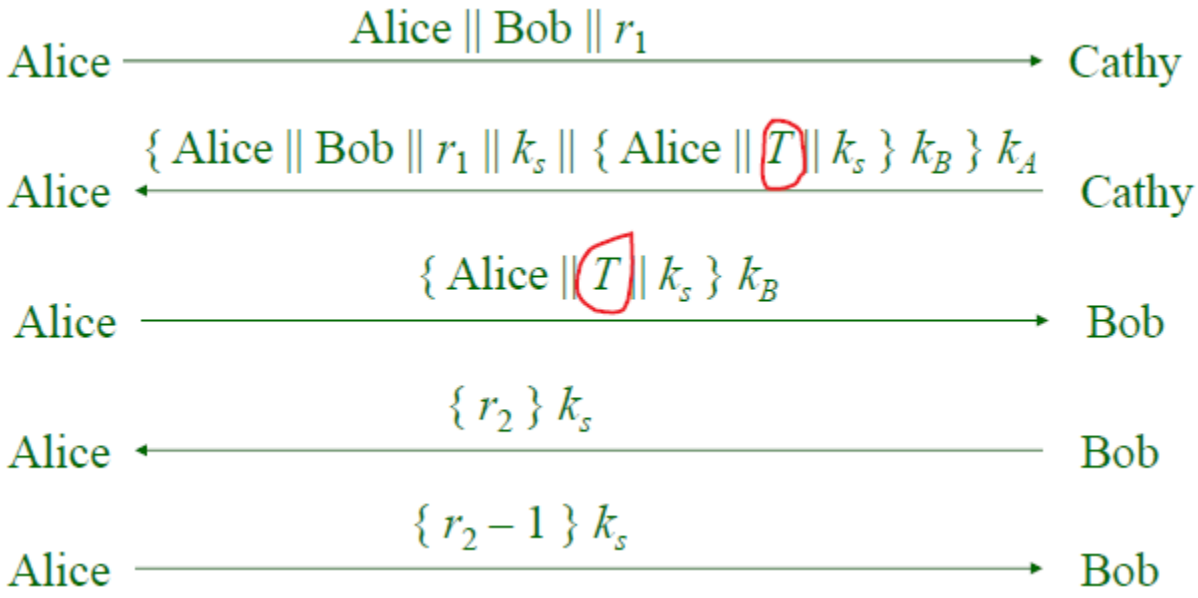
Problem with Needham-Schroeder Protocol

- What is Eve compromised the session key. How does that affect protocol?
    - In what follows, **Eve knows recycled k_s**



Needham-Schroeder with Denning-Sacco Modification

- Solution: **use time stamp to detect replay**



Good News

Won't match the current time

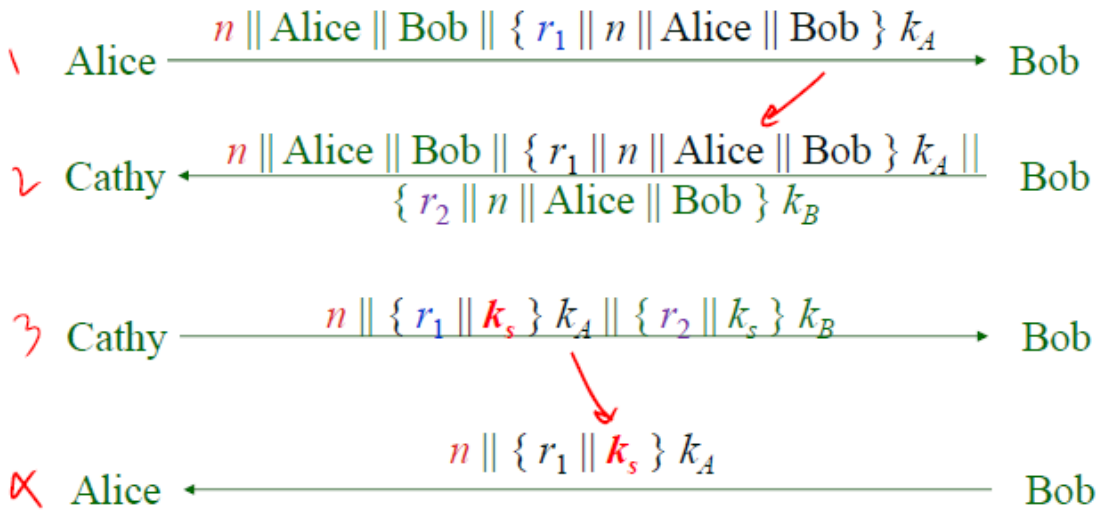
Bad News

- If clock not synchronized for communicating parties, may either *reject valid messages* or *accept replays*
    - Parties with either slow or fast clocks are vulnerable to replay

Otway-Rees (OR) Protocol

- Does not use timestamps to prevent replay
    - Not vulnerable to the problems that Denning-Sacco modification has
- Uses *integer n* to associate all messages with particular exchange

The OR Protocol



- How does Alice know she is talking to Bob?
    - Bob receives message from Cathy, makes his own envelope with another random number and sends it back to Cathy.
        - Cathy receives two messages (from Alice and Bob)
    - What information does she need to decrypt the envelope?
        - The two keys (Alice and Bob is written in clear text – "n||Alice||Bob")
    - When she opens them, she finds the random numbers
        - If the number outside of the envelope is changed, then it will not match the number inside of the envelope

Argument: Alice talking to Bob

- Fourth message:
  - If **n** matches first message, Alice knows it is part of this protocol exchange
  - Cathy generated k_s because only she, Alice, knows $\mathbf{k_A}$
  - Enciphered part belongs to exchange as $\mathbf{r_1}$ matches r_1 in encrypted part of first message

$$\mathbf{n} \,||\, \{\, \mathbf{r_1} \,||\, \mathbf{k_s} \,\}\, \mathbf{k_A}$$

Argument: Bob talking to Alice

- Third message
  - If **n** matches second message, Bob knows it is part of this protocol exchange
  - Cathy generated k_s because only he, Bob, knows $\mathbf{k_B}$
  - Enciphered part belongs to exchange as $\mathbf{r_2}$ matches r_2 in encrypted part of second message

$$n \,||\, \ldots \,||\, \{\, r_2 \,||\, k_s \,\}\, k_B$$

Replay Attack with Otway-Rees Protocol

- Eve acquires old k_s message in third step
  - $\underline{n \,||\, \{\, r_1 \,||\, k_s \,\}\, k_A} \,||\, \{\, r_2 \,||\, k_s \,\}\, k_B$
- Eve forwards appropriate part to Alice (underlined portion)
  - Good News
    - Alice has no ongoing key exchange; n matches nothing, so is rejected
    - Alice has ongoing key exchange (with different n); n does not match, so is again rejected
  - Bad News
    - IF replay is for the current key exchange (while n is in process), AND Eve sent the relevant part BEFORE the other party, only then it would work
      - Not so bad then…

Kerberos

- Network Authentication Protocol for client-server systems
    - Created by MIT
    - Mutual authentication between user and service
    - Single sign-on
        - Once I am authenticated, I can use smaller requests to continue using the server
- Only people authenticated to use server will be allowed on
    - The need is to authenticate if someone is the correct user or service (mutual authentication – not one way, both ways)
- Message encryption uses symmetric key cryptography with DES (version 4)
    - Extended version (version 5) allows AES and public key cryptography for authentication
- Key exchange based on Needham-Schroeder with Denning-Sacco modification

Concepts

- Key Distribution Center (KDC)
    - Trusted third party ("Cathy")
    - Consists of
        - Authentication Server (AS)
            - Identifies user as the correct user
        - Ticket Granting Server (TGS)
            - Vouches for requester of service to the service
        - Works together – Separation of Duties
    - Ex: Providing license to verify age so that you can get into park – receive ticket stating you are of age. Need ticket for each ride, show age validation from earlier rather than being checked at each ride

Kerberos Keys

- **Interchange keys**
    - Used encrypt Tickets containing session keys
        - One shared between User and AS (user key) – $\mathbf{k_u}$
        - One shared between AS and TGS – $\mathbf{k_{TGS}}$
        - One shared between TGS and service S – $\mathbf{k_s}$
- **Session keys**
    - Used to encrypt messages/authenticators
        - One shared between User and TGS – $\mathbf{k_{u,TGS}}$
        - One shared between User and service S – $\mathbf{k_{u.s}}$

Kerberos Authenticator

- Authenticator (Like a license)
    - **Credential** containing identity of submitter of ticket
    - **Encrypted with session key** that vouches for user's identity
    - Generated by user (to show to TGQ and service)
    - Authenticator for TGS
        - $A_{U,TGS} = \{ u \,||\, \text{generation time} \,||\, k_t \}\, k_{U,TGS}$

          where:
            - Session key $k_{U,TGS}$ for user and TGS
            - k_t is alternate session key
            - Generation time is when authenticator generated
                - Note: more fields, not relevant here
    - Authenticator for service s
        - $A_{U,S} = \{ u \,||\, \text{generation time} \,||\, k_t \}\, k_{U,S}$

          where:
            - Session key $k_{U,S}$ for user and service

Kerberos Ticket

- Ticket (Like a plane ticket)
    - Credential to serve as permission and shows that issuer (trusted 3rd party) has authenticated entity
        - Generated by AS (for TGS use) and by TGS (for service use)
    - **Distributes session key**
        - Encrypted with interchange key
    - **Unforgeable** – cannot change it (will be detected)
    - **Authenticated** – can be verified
    - **Non-replayable**

Kerberos Tickets

- Ticket issued to user u **by AS for TGS** (Ticket Granting Ticket/TGT)
    - $T_{U,TGS}$ = TGS || { u || u's address || valid time || $k_{U,TGS}$ } k_{TGS}

      where:
        - $k_{U,TGS}$ is session key for user and TGS
        - k_{TGS} is secret key shared by TGS and Authenticator
        - Valid time is interval for which ticket is valid
        - u's address may be IP address or something else
- Ticket issued to user u **by TGS for service s**
    - $T_{U,S}$ = TGS || { u || u's address || valid time || $k_{U,S}$ } k_S
    - where:
        - $k_{U,S}$ is session key for user and service
        - k_S is secret key shared by TGS and service

April 16th – Confinement Problem

The Generalization of Client-Server Systems

- Client sends request, data to server
- Server performs some function on data
- Server returns result to client
- Access Controls:
    1. Server must ensure that the resources it accesses on behalf of the client must include *only* resources client is authorized to access
    2. Server must ensure it does not reveal client's data to any entity not authorized to see the client's data

The Confinement Problem

- Problem of preventing a server from leaking information that the user of the service considers confidential

Total Isolation

- No process can communicate with any other process
- Process cannot be observed
- Impossible for this process to leak information
    - Not practical as process uses observable resources, such as COU, secondary storage, networks, etc.

Example Problem 1:

- Process p, q not allowed to communicate
    - But they share a file system!
- Communications protocol:
    - P sends a bit by creating a file called 0 or 1, then a second file called *send*
        - P waits until *send* is deleted before repeating to send another bit
    - Q waits until file *send* exists, then looks for file 0 or 1; whichever exists is the bit

        - Q then deletes 0, 1 and *send* and then waits until *send* is recreated before repeating to read another bit
    - P ends by creating a file called *end*
- Covert Channel

Transitive Confinement

- If p is confined to prevent leaking, and it involves q, then q can leak the information that p passes
- Rule:
    - If a confined process invokes a second process, the second process must be as confined as the first
- Confinement should be on the transmission as well

Example Problem 2: Regarding Resource Sharing

- Another type of covert channel
- A process can obtain ("read") rough idea of time
    - Counting the number of instructions executed during a period
        - System clock or wall clock
- A process can manipulate ("write") time
    - Executing a number of instructions and stopping, allowing another process to execute
- This is possible when a process shares the computer with another process
    - Isolation is a remedy

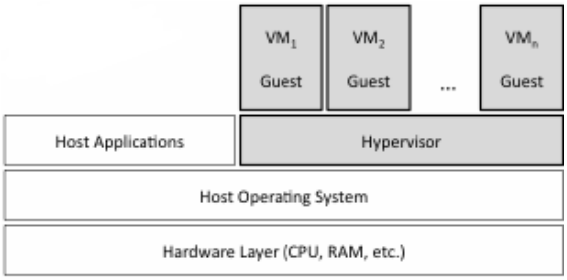
Protection through Isolation

- Virtual Machine
    - Emulates computer
    - Process cannot access underlying computer system, or anything that is not part of the environment

- Sandbox
    - It is not meant to emulate computer
        - Virtual machine is a kind of sandboxing
    - Alters interface between computer, process

Virtual Machine (VM)

- A program that simulates hardware of computer system
- Virtual Machine Monitor (VMM) or Hypervisor provides VM on which conventional OS can run
    - Each VM is one subject – VMM knows nothing about processes running on each VM
    - VMM mediated all interactions of VM with resources, other VMs
    - Enforces rule of transitive confinement



Sandbox

- Environment in which actions of processes are restricted according to security policy
- Two ways
    - **Limiting execution**
        - Adding extra security-checking mechanisms to libraries, kernel
            - Program to be executed is not altered
    - **Intervening**
        - Modifying program or process to be executed
            - Similar to debuggers, profilers that add breakpoints
            - Add code to do extra checks (memory access, etc.) as program runs (*software fault isolation*)

Covert Channels

- A path of communication NOT intended for communication
    - Using shared resources as a communication path
- Covert storage channel uses attribute of shared resource
- Covert timing channel uses temporal or ordering relationship among accesses to shared resource

Example: **Covert Storage Channel**

- Shared file system
    - We have already seen this
- Communications protocol:
    - P sends a bit by creating a file called 0 or 1, then a second file called send
        - P waits until *send* is deleted before repeating to send another bit
    - Q waits until file *send* exists, then looks for file 0 or 1; whichever exists is the bit
        - Q then deletes 0, 1, and *send* and then waits until *send* is recreated before repeating to read another bit
- Covert storage channel: resource is directory, names of files in directory

Example: **Covert Timing Channel**

- Real-Time Clock
    - We have seen this already
- Virtual machines sharing the same physical device (e.g. KVM/370) can have a covert timing channel
    - VM1 wants to send 1 bit to VM2
    - To send 0 bit: VM1 relinquishes CPU as soon as it gets CPU
    - To send 1 bit: VM1 uses CPU for full quantum
    - VM2 determines which bit is sent by seeing how quickly it gets CPU
    - Shared resource is CPU, timing because real-time clock used to measure intervals between accesses

Example: **Ordering of Events**

- A timing covert channel, not by real clock
    - Two VMs
        - Share cylinders 100 – 200 on a disk
        - One is *High*, one is *Low*; process on *High* VM wants to send to process on *Low* VM
    - Disk scheduler uses SCAN algorithm
    - Low processes seeks to cylinder 150 and relinquishes CPU
    - *High* wants to send a bit
        - To send 1 bit, *High* seeks to cylinder 140 and relinquish CPU
        - To send 0 bit, *High* seeks to cylinder 160 and relinquish CPU
    - *Low* issues requests for tracks 139 and 161
        - Seek to 139 first indicates a 1 bit
        - Seek to 161 first indicates a 0 bit
    - Covert timing channel: uses ordering relationship among accesses to transmit information

Noiseless and Noisy Covert Channels

- Noiseless covert channel
    - Uses shared resource available to sender, receiver only
- Noisy covert channel
    - Uses shared resource available to sender, receive, and others
    - Attackers need to minimize interference enough so that message can be read in spite of others' use of channel

Key Properties of Covert Channels

- **Existence**
    - Determining whether covert channel can exist
- **Bandwidth**
    - Determining how much information can be sent over the channel

Defense Against Covert Channels

- Identify Covert Channel
    - Eliminate existence
    - Reduce bandwidth

Detection

- Covert channels require sharing
- Manner of sharing controls
    - Which subjects can send
    - Which subjects can receive
- Ways of identification
    - Shared resource matrix
    - Covert flow trees

Shared Resource Matrix

- Detection approach by Kemmerer
    - Use matrix of resources (rows) and processes that can access them (columns)
    - The entries denote the access rights
    - Analyst examine the matrix to see if information can be leaked
    - Example:
        - Process Low is not allowed to read Resource 2

	Process High	Process Low
Resource 1	Read, Modify	Read, Modify
Resource 2	Read	

        - Process Low CAN read Resource 2 with the help of Process High

	Process High	Process Low
Resource 1	Read, Modify	Read, Modify
Resource 2	Read	Read

Covert Flow Trees

- Porras and Kemmerer
    - Model “flow of information through shared resources” with a tree structure
        - Tree structure representation of sequence of operations that move information from one process to another
    - Analyst examines the flow path to see if security is one violated

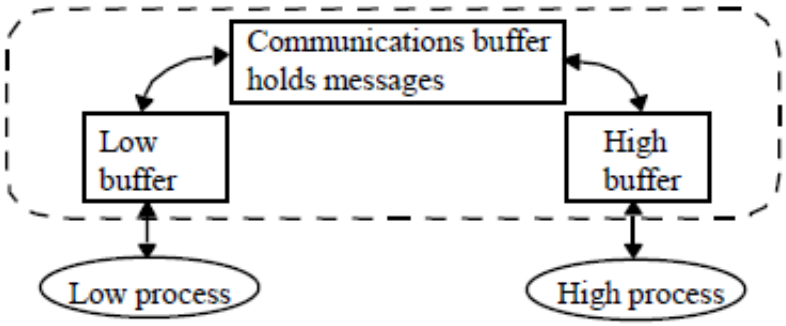
Shared Resource Matrix vs. Covert Flow Tree

- Shared resource matrix can only identify existence of channels
- Cover flow trees can identify existence of (more) channels and sequence of operations that the attacker can use for the covert channels

Covert Channel Mitigation

- **Obvious fix:**
    - Ask processes what resources they need and provide them those ONLY
        - Need must specify the runtime as well
    - Take back after quota finishes
- **Another way:**
    - Obscure amount of resources a process uses
        - Receiver cannot determine what part of sender is using and what not
    - How to do this?
        - Devote uniform, fixed amount of resources to each process
        - Inject randomness into allocation, use of resources
    - Consider tradeoff for loss in efficiency

Another Covert Channel: Pump



- The pump works similarly to the channel but it has a communication buffer which holds the message.
- From the low buffer, information is sent to the communication buffer and the communication buffer sends acknowledgement back to the low buffer

Problem

- Assume: Low process can process messages faster than High process
- Low process sends messages faster than High process can remove them
    - Opens covert channel

Covert Timing Channel with Pump

- High process can control rate at which pump sends it messages
- Initialization: Low sends messages to pump until communications buffer is full
    - Low gets ACK for each message put into the buffer; no ACK for messages when communications buffer full
- Protocol: sequence of trials; for each trial
    - High sends a 1 by reading a message
        - Then Low gets ACK
    - High sends a 0 by not reading a message
        - Then Low doesn't get ACK

How to Fix

- **Closing covert channel**
    - Making High process handle messages more quickly than Low process gets acknowledgements
        - Pump artificially delaying ACKs
        - Low cannot tell whether buffer is full
        - Not optimal (processes may wait even when unnecessary)
- **Decreasing bandwidth**
    - Making pump and processes handle messages at the same rate
        - Decreases bandwidth of covert channel
        - Sub-optimal performance

Key Points

- Confinement Problem: prevent leakage of information
    - Solution: separation and/or isolation
- Shared resources offer paths along which information can be transferred
- Covert channels difficult if not impossible to eliminate
    - Bandwidth can be greatly reduced

April 18th – Key Management (Kerberos)

- Bonus questions on exam from the term projects
- Review session next weekend
- Just one report submission; peer eval due from everyone

Problematic Questions from Quiz 4:

9. (Rainbow table) How did you find it?
   a) I found the hash in the rainbow table and selected the head that belonged to that chain. Once I knew the head, I recreated the chain by reducing and hashing until I found the hash and the text that generated the hash is the password
   b) Hash is not in the second column in the table – reduce and hash until you find the hash. Once you find the hash you grab the head and recreate the reducing and hashing until you find the text that created the hash

Example Questions

- What is the relationship between the interchange key that the user shares with the server and the password?
    - It will pull up the password and create the hash that serves as the key.
    - The authentication server pulls up the stored password and runs it through a hash function to see if it matches the stored key.
    - The user, machine, and AS use the password to calculate the hash but it is not transmitted
    - They now have the interchange key which is shared between them.
- How does the session key get exchanged?
    - Using the protocols
    - INTERCHANGE KEYS DISTRIBUTE SESSION KEYS
    - Interchange key is permanent, session key is one time
- The ticket has something in it that validates the authenticator
- **Watch some tutorials on kerberos**

Overview

- User *u* authenticates to Kerberos authentication server
    - Obtains **ticket** $T_{U,TGS}$ for ticket granting service (TGS)
- User *u* wants to use service (TGS)
    - User generates **authenticator** A_u and sends it with ticket $T_{U,TGS}$ to TGS asking for ticket for service
    - TGS sends ticket $T_{U,S}$ to user
    - User sends $A_{U,}$ $T_{U,S}$ to server as request to use s

Pros

- Cryptographic protection against spoofing/eavesdropping
- Time validity protection against long term attacks
- Timestamp protection to prevent replay attacks
- Session keys
- No password transmission

Cons

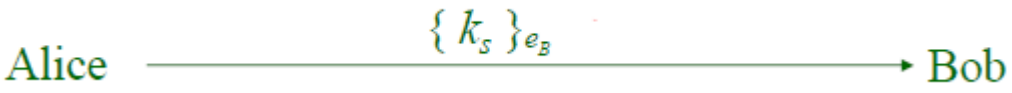
- Single point of failure
- Requires
    - Continuous availability of trusted TGS
    - Trust relationship between TGS and each server
    - Timely transactions
        - **Relies on synchronized clocks**
- User workstation can be compromised and password logged and later replays
    - Password guessing can be a problem
- Does not scale well
- Evolved Kerberos Version 5

Kerberos 5

- Internet standard (specified in RFC1510)
- Supports any symmetric key algorithm (not just DES)
- Longer ticket validity period
- Allows ticket to be renewed
- Supports different types of networking protocols (not just IP)
- Better protection against replay attacks
- Support cross-realm (realm is set of nodes one KDC serves) authentication
- Continued support

(Public Key) (Key Exchange)

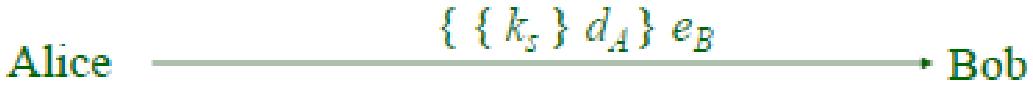
- **Interchange keys**
  - e_A, e_B Alice and Bob's public keys known to all
  - d_A, d_B Alice and Bob's private keys known only to owner
- **Simple Protocol**
  - K_s is desired session key



  - With this ^ Bob can't tell if Alice sent the message

Problem and Solution

- Vulnerable to forgery or replay
  - e_B is known to everyone, Bob has no assurance that Alice sent message
- Simple fix using Alice's private key

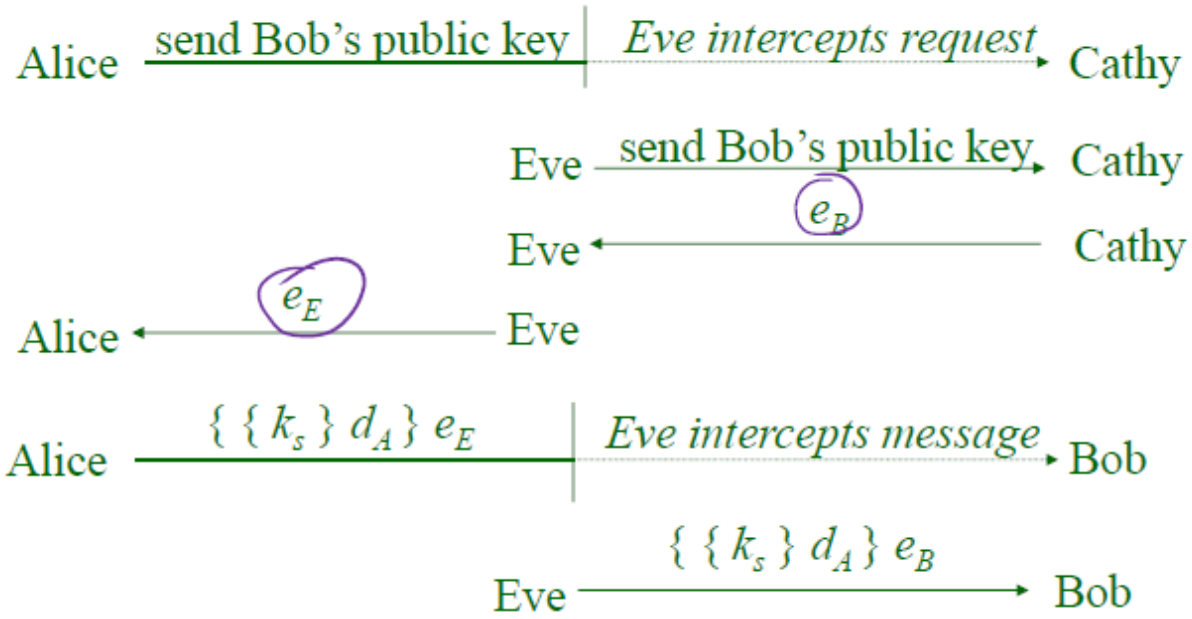


  - Now we can tell that Alice sent the message

But…

- Assume Bob has Alice's public key, and *vice versa*
- If not, each must get it from public server
    - If keys not bound to the identity of the owner, the attacker (eve) can launch a ***man-in-the-middle* attack**
        - An attacker intercepts and relays messages between two parties pretending to be the "other" party to each one

Man-in-the-Middle Attack



- Eve gets the public key from Cathy, she sends Alice her own public key pretending to be Bob
- Now Alice has Eve's public key thinking she is Bob
- Now Eve will receive messages from Alice that she can read with her own private key
- She will now send messages back that are encrypted with Bob's public key
- The problem is there is no authentication of the public key
    - How can you authenticate the public key?
        - Cryptographic Key infrastructure

Cryptographic Key Infrastructure

- Binding of key to owner
  - Classical cryptography: not applicable as all are shared secret keys
    - Use protocols to agree on a shared key
  - Public key cryptography
    - Uses certificates to bind principal/user (identified by an acceptable name) to public key

Certificates

- **Verifiable electronic notarized endorsement**
  - **Verifiable – Hash**
  - **Notarized – Signed by trusted third party**
  - **Endorsement – Cathy is telling you that the key belongs to Alice**
    - Certificates purpose is to give you a public key
    - It associates the public key with the user
  - Vouched/signed by trusted 3rd party
- Token (message) containing
  - Identity of principle (here, Alice)
  - Corresponding public key
  - Timestamp (when issued)
  - Other information (perhaps identity of signer/issuer)
- Protected against alteration
  - Includes checksum/hash
  - Hash should be signed by trusted third party (encrypted with private key of person creating certificate – Cathy) →Signed part is d_c

$$C_A = e_A \,||\, \text{Alice} \,||\, T \,||\, \{h(e_A \,||\, \text{Alice} \,||\, T)\}\ d_c$$

Data Hash Signed

Using Certificates

- Issuer Cathy
  - Creates certificate
    - Generated hash of certificate
    - Signs hash with issuers private key

$$C_A = e_A \,||\, \text{Alice} \,||\, T \,||\, \{h(e_A \,||\, \text{Alice} \,||\, T)\} d_C$$

- Requester Bob of Alice's public key
  - Uses certificate
    - Get data
      - Obtains Alice's public key from certificate
    - Validate data
      - Obtains issuer's (Cathy's) public key
        - **Deciphers hash**
      - Recomputes hash from certificate
      - Compares hash and validates

Problem and Solution

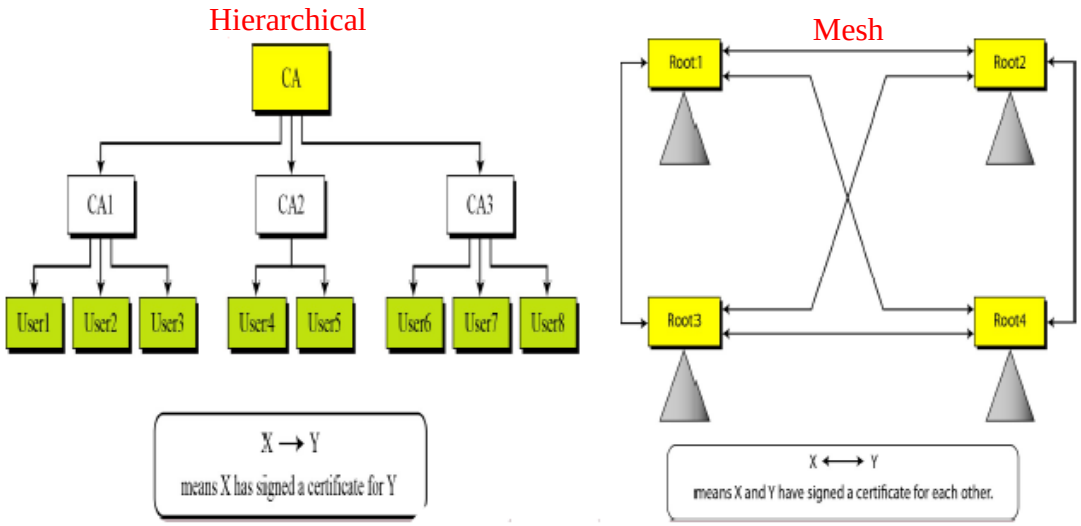
- Bob needs Cathy's public key to validate certificate
  - Problem pushed "up" a level
- Solution:
  - Use of **Certification Authorities (CA)** who generates certificates
- Certificates
  - X.509 and OpenPGP

X.509 Certificate

- A.k.a. Directory authentication framework
- Each certificate has information about 3 cryptographic functions: public key, hash, digital signature
- Primary certificate components in X.509v3
    - Certificate
        - Version
        - Serial Number
        - Signature encryption algorithm info
        - Issuer's name
        - Interval of validity
        - Subject's name
        - Subject's public key info
            - Algorithm and key
    - Signature
        - Hash algorithm
        - Enciphered hash

PKI Models

- Public Key Infrastructure (PKI)
    - Management system that allows the creation, distribution and revocation of X.509 certificates through certificate authorities
- Trust Models
    - **Hierarchical** Cas with root CA who can self sign
    - **Mesh model** with cress-certification (e.g. X.509)
        - Disjoint hierarchical groups with root Cas who cross certify



- You can only be certified by a single user
- Heriarchical
  - Has root that validates subgroups
  - The root is trusted so does not need to be validates
- Mesh
  - Cross-certifies each other

Cross-Certifying in X.509

- Arbitrary length certificate chains rely on individual's knowledge of certifiers
  - Alice's CA is Cathy; Bob's CA is Don
  - How can Alice validate Bob's certificate?
  - Have Cathy and Don cross-certify
- Certificates ("I know a friend who knows a friend…")
  - Cathy<<Alice>>
  - Con<<Bob>>
  - Cathy<<Don>>
  - Don<<Cathy>>
- Alice validates Bob's certificate
  - Alice uses (known) public key of Cathy to validate Cathy<<Don>>
  - Alice uses Cathy<<Don>> to validate Don<<Bob>>

OpenPGP

- Primary certificate standard for PGP (Pretty Good Privacy)
    - Used for secure email communication
    - Multiple users can vouch for you
    - Web Structure
- PGP initially supported only **Web of Trust Model** where…
    - **Anyone** can sign for anyone (even self-sign)
    - **No "authority"**
    - Instead, users certify each other to build a "web of trust"
        - Current version allows hierarchical model with X.509 certificates
- OpenPGP certificates structured into two types of packets
    - One public key packet
    - Zero or more signature packet

OpenPGP Public Key Packet

- Public key packet:
    - Version (3/4)
    - **Creation time** →(Not in X.509)
    - Validity period
    - Public key algorithm, associated parameters
    - Public key

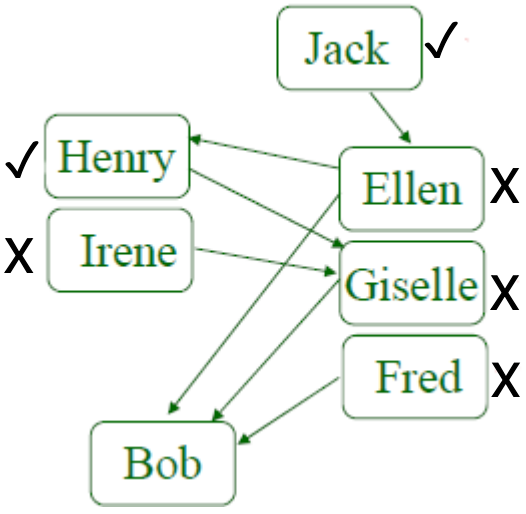
OpenPGP Signature Packet

- Major differences between OPGP and X.509: Mutiple entities (or no one) can vouch and there are levels of trust
- Version 3 signature packet
    - Certificate
        - Version (3)
        - **Signature type** (level of trust)
            - Range from “untrusted” to “ultimate trust”
        - **Creation time**
        - Signer’s signature key identifier
        - Signer’s signature key algorithm
        - Hash algorithm used to generate the hash
    - **Part of signed hash** (used for quick check)
    - Signature (enciphered hash using signer’s private key)
- Version 4 packet more complex

Validating Certificates with PGP

- Alice needs to validate Bob’s OpenPGP certificate
    - Does not know Fred, Giselle, or Ellen
- Alice gets Giselle’s certificate
    - Does not know Irene
    - Knows Henry, but his signature is at “casual” level of trust
- Alice gets Ellen’s certificate
    - Knows Jack who certifies at full trust, so uses his cert to validate Ellen’s, then hers to validate Bob’s

Arrows show signatures
Self signatures not shown



Storing Asymmetric Keys

- For public key
    - IA goal is Integrity
    - Achieved with Certificates
- For private/secret key
    - IA goal is confidentiality
    - Multi-user or networked systems: attackers may defeat access control mechanisms
        - **Encipher file containing key**
            - Attacker can monitor keystrokes to decipher files
            - Key will be resident in memory that attacker may be able to read
        - **Use physical devices like "smart card"**
            - Key never enters system
            - Card can be stolen, so have 2 devices combine bits to make single key

Revoking Keys

- Certificates **invalidated before expiration**
    - Usually due to compromised key/issuer
    - May be due to change in circumstance (someone leaving the company)
- Problems with revoking
    - Circulating revocation information quickly
    - Checking if the entity revoking the certificate is authorized to do so

Certificate Revocation

- X.509: only certificate issuer can revoke certificate
    - Added to CRL
    - ***Certificate revocation list*** (CRL) lists certificated that are revoked
        - Date and Serial

- OpenPGP: issuer, as well as, owner can revoke certificates, or allow others to do so
    - Revocation message placed in OpenPGP packet and signed
    - Expiry dates

Primary Differences Between X.509 and OpenPGP

X.509	**PGP**
• Single path because you can only be certified by one	• Multiple paths because multiple people can endorse you
• Fully trusted because there is no level or "slight" trust	• Partial trust: anything from 0 to 1
• One certificate has one issuer	• A single certificate may have multiple signatures
• No level of trust	• Signature type (level of trust)
• Heirarchical (with root CA) model	• Web of trust model
• Greater impact if issuer (CA) is compromised	• Lesser impact if issues is compromised
• Only issuer can revoke	• Both issuer and owner can revoke
• Use CRL for revocation	• Use info within PGP packets

End Note

- Security is difficult because security is not…
    - Inherent: not default/born in something; must be designed
    - Universal: not just one way to do it; What works for me may not work for you
    - Static: Always changing
    - Absolute: Secure today does not mean secure tomorrow
- Security is a compromise between usability, cost, and peace of mind
- EXPECTATION → PROTECTION → PERCEPTION