AAA

Authentication
Authorization
Access Control

Access Control

- Two parts to access control
- **Authentication:** Are you who you say you are?
  - Determine whether access is allowed
  - Authenticate human to machine
  - Or authenticate machine to machine
- **Authorization:** Are you allowed to do that?
  - Once you have access, what can you do?
  - Enforces limits on actions
- Note: "access control" often used as synonym for authorization

Authentication

Guard: Halt! Who goes there?
Arthur: It is I, Arthur, son of Uther Pendragon, from the castle of Camelot. King of the Britons, defeater of the Saxons, sovereign of all England!
— *Monty Python and the Holy Grail*

Then said they unto him, Say now Shibboleth: and he said Sibboleth: for he could not frame to pronounce it right. Then they took him, and slew him at the passages of Jordan: and there fell at that time of the Ephraimites forty and two thousand.
— *Judges 12:6*

Are You Who You Say You Are?

- How to authenticate human a machine?
- Can be based on…
  - Something you **know**
    - For example, a password
  - Something you **have**
    - For example, a smartcard
  - Something you **are**
    - For example, your fingerprint

Something You Know

- Passwords
- Lots of things act as passwords!
  - PIN
  - Social security number
  - Mother's maiden name
  - Date of birth
  - Name of your pet, etc.

Trouble with Passwords

- "Passwords are one of the biggest practical problems facing security engineers today."
- "Humans are incapable of securely storing high-quality cryptographic keys, and they have unacceptable speed and accuracy when performing cryptographic operations. (They are also large, expensive to maintain, difficult to manage, and they pollute the environment. It is astonishing that these devices continue to be manufactured and deployed.)"

Why Passwords?

- Why is "something you know" more popular than "something you have" and "something you are"?
- **Cost**: passwords are free
- **Convenience**: easier for admin to reset pwd than to issue a new thumb

Keys vs Passwords

- **Crypto keys**
- Spse key is 64 bits
- Then 2^{64} keys
- Choose key at random…
- …then attacker must try about 2^{63} keys

- **Passwords**
- Spse passwords are 8 characters, and 256 different characters
- Then $256^8 = 2^{64}$ pwds
- Users do not select passwords at random
- Attacker has far less than 2^{63} pwds to try (**dictionary attack**)

Good and Bad Passwords

- Bad passwords
  - frank
  - Fido
  - password
  - 4444
  - Pikachu
  - 102560
  - AustinStamp
- Good Passwords?
  - jfIej,43j-EmmL+y
  - 0986437653726 3
  - P0kem0N
  - FSa7Yago
  - OnceuP0nAt1m8
  - PokeGCTall150

Password Experiment

- Three groups of users — each group advised to select passwords as follows
  - **Group A**: At least 6 chars, 1 non-letter
  - winner → **Group B**: Password based on passphrase
  - **Group C**: 8 random characters
- Results
  - **Group A**: About 30% of pwds easy to crack
  - **Group B**: About 10% cracked
    - Passwords easy to remember
  - **Group C**: About 10% cracked
    - Passwords hard to remember

Password Experiment

- User compliance hard to achieve
- In each case, 1/3rd did not comply
  - And about 1/3rd of those easy to crack!
- Assigned passwords sometimes best
- If passwords not assigned, best advice is…
  - Choose passwords based on passphrase
  - Use pwd cracking tool to test for weak pwds
- Require periodic password changes?

Attacks on Passwords

- Attacker could…
  - Target one particular account
  - Target any account on system
  - Target any account on any system
  - Attempt denial of service (DoS) attack
- Common attack path
  - Outsider → normal user → administrator
  - May only require **one** weak password!

Password Retry

- Suppose system locks after 3 bad passwords. How long should it lock?
  - 5 seconds
  - 5 minutes
  - Until SA restores service
- What are +'s and -'s of each?

Password File?

- Bad idea to store passwords in a file
- But we need to verify passwords
- Cryptographic solution: **hash** the pwd
  - Store $y = h(\text{password})$
  - Can verify entered password by hashing
  - If Trudy obtains "password file," she does not obtain passwords
- But Trudy can try a *forward search*
  - Guess x and check whether $y = h(x)$

Dictionary Attack

- Trudy pre-computes $h(x)$ for all x in a **dictionary** of common passwords
- Suppose Trudy gets access to password file containing hashed passwords
  - She only needs to compare hashes to her pre-computed dictionary
  - After one-time work, actual attack is trivial
- Can we prevent this attack? Or at least make attacker's job more difficult?

Salt

- Hash password with **salt**
- Choose random salt s and compute

$$y = h(\text{password}, s)$$

  and store (s,y) in the password file
- Note: The salt s is not secret
- Easy to verify salted password
- But Trudy must re-compute dictionary hashes for each user
  - Lots more work for Trudy!

Password Cracking: Do the Math

- Assumptions:
- Pwds are 8 chars, 128 choices per character
  - Then $128^8 = 2^{56}$ possible passwords
- There is a **password file** with 2^{10} pwds
- Attacker has **dictionary** of 2^{20} common pwds
- **Probability** of 1/4 that a pwd is in dictionary
- Work is measured by number of hashes

Password Cracking: Case I

- Attack 1 password without dictionary
  - Must try $2^{56}/2 = 2^{55}$ on average
  - Like exhaustive key search
- Does **salt** help in this case?

Password Cracking: Case II

- Attack 1 password with dictionary
- With **salt**
  - Expected work: $1/4\ (2^{19}) + 3/4\ (2^{55}) = 2^{54.6}$
  - In practice, try all pwds in dictionary…
  - …then work is at most 2^{20} and probability of success is $1/4$
- What if **no salt** is used?
  - One-time work to compute dictionary: 2^{20}
  - Expected work still same order as above
  - But with precomputed dictionary hashes, the "in practice" attack is free…

Password Cracking: Case III

- Any of 1024 pwds in file, **without** dictionary
  - Assume all 2^{10} passwords are distinct
  - Need 2^{55} **comparisons** before expect to find pwd
- If **no salt** is used
  - Each computed hash yields 2^{10} comparisons
  - So expected work (hashes) is $2^{55}/2^{10} = 2^{45}$
- If **salt** is used
  - Expected work is 2^{55}
  - Each comparison requires a hash computation

Password Cracking: Case IV

- Any of 1024 pwds in file, **with** dictionary
  - Prob. one or more pwd in dict.: $1 - (3/4)^{1024} = 1$
  - So, we ignore case where no pwd is in dictionary
- If **salt** is used, expected work less than 2^{22}
  - See book, or slide notes for details
  - Approximate work: size of dict. / probability
- What if **no salt** is used?
  - If dictionary hashes not precomputed, work is about $2^{19}/2^{10} = 2^9$

Other Password Issues

- Too many passwords to remember
  - Results in password reuse
  - Why is this a problem?
- Who suffers from bad password?
  - Login password vs ATM PIN
- Failure to change default passwords
- Social engineering
- Error logs may contain "almost" passwords
- Bugs, keystroke logging, spyware, etc.

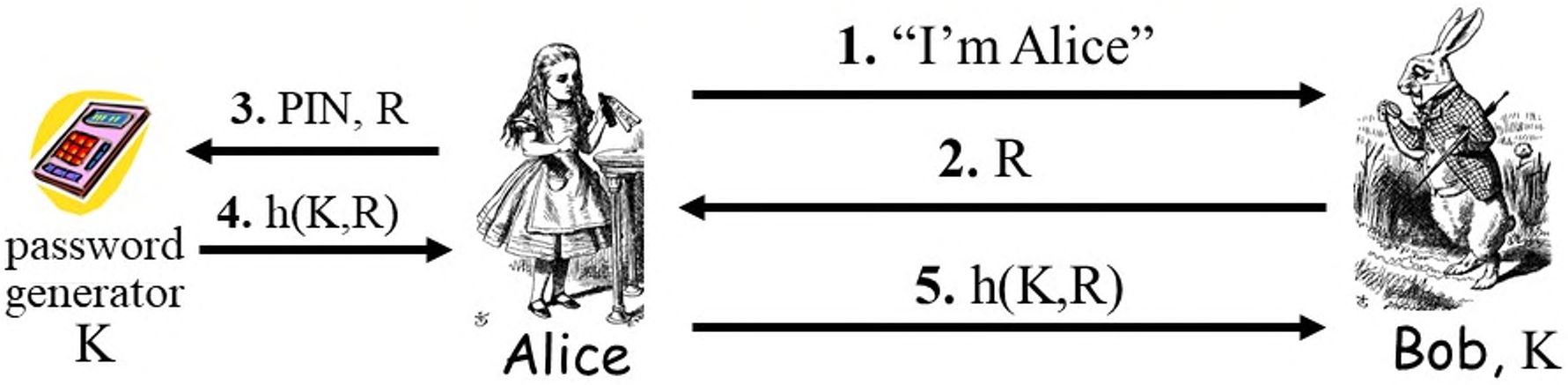
Passwords

- The bottom line…
- **Password cracking is too easy**
  - One weak password may break security
  - Users choose bad passwords
  - Social engineering attacks, etc.
- Trudy has (almost) all of the advantages
- All of the math favors bad guys
- Passwords are a **BIG** security problem
  - And will continue to be a big problem

Password Cracking Tools

- Popular password cracking tools
  - Password Crackers
  - Password Portal
  - LOphtCrack and LC4 (Windows)
  - John the Ripper (Unix)
- Admins should use these tools to test for weak passwords since attackers will
- Good articles on password cracking
  - Passwords - Conerstone of Computer Security
  - Passwords revealed by sweet deal

Password Generator



- Alice receives random "challenge" R from Bob
- Alice enters PIN and R in password generator
- Password generator hashes symmetric key K with R
- Alice sends "response" h(K,R) back to Bob
- Bob verifies response
- Note: Alice **has** pwd generator and **knows** PIN

2-factor Authentication

- Requires any 2 out of 3 of
  - Something you know
  - Something you have
  - Something you are
- Examples
  - ATM: Card and PIN
  - Credit card: Card and signature
  - Password generator: Device and PIN
  - Smartcard with password/PIN

Single Sign-on

- A hassle to enter password(s) repeatedly
  - Alice wants to authenticate only once
  - "Credentials" stay with Alice wherever she goes
  - Subsequent authentications transparent to Alice
- Kerberos --- example single sign-on protocol
- Single sign-on for the Internet?
  - Microsoft: **Passport**
  - Everybody else: **Liberty Alliance**
  - Security Assertion Markup Language (**SAML**)

Web Cookies

- Cookie is provided by a Website and stored on user's machine
- Cookie indexes a database at Website
- Cookies **maintain state** across sessions
  - Web uses a stateless protocol: HTTP
  - Cookies also maintain state within a session
- Sorta like a single sign-on for a website
  - But, a very, very weak form of authentication
- Cookies also create privacy concerns

Authorization

Shynar Mussiraliyeva

Authorization

It is easier to exclude harmful passions than to rule them,
and to deny them admittance
than to control them after they have been admitted.
— Seneca

You can always trust the information given to you
by people who are crazy;
they have an access to truth not available through regular channels.
— Sheila Ballantyne

Authentication vs Authorization

- Authentication — Are you who you say you are?
  - Restrictions on who (or what) can access system
- **Authorization** — Are you allowed to do that?
  - Restrictions on actions of authenticated users
- Authorization is a form of **access control**
- But first, we look at system certification…

System Certification

- Government attempt to certify "security level" of products
- Of historical interest
  - Sorta like a history of authorization
- Still required today if you want to sell your product to the government
  - Tempting to argue it's a failure since government is so insecure, but…

Orange Book

- Trusted Computing System Evaluation Criteria (TCSEC), 1983
  - Universally known as the "orange book"
  - Name is due to color of it's cover
  - About 115 pages
  - Developed by DoD (NSA)
  - Part of the "rainbow series"
- Orange book generated a pseudo-religious fervor among some people
  - Less and less intensity as time goes by

Orange Book Outline

- Goals
  - Provide way to assess security products
  - Provide guidance on how to build more secure products
- Four ***divisions*** labeled D thru *A*
  - D is lowest, *A* is highest
- Divisions split into numbered ***classes***

D and C Divisions

- D --- minimal protection
  - Losers that can't get into higher division
- C --- discretionary protection, i.e., don't force security on users, have means to detect breaches (audit)
  - C1 --- discretionary security protection
  - C2 --- controlled access protection
  - C2 slightly stronger than C1 (both vague)

B Division

- B --- mandatory protection
- B is a huge step up from C
  - In C, can break security, but get caught
  - In B, "mandatory" means can't break it
- B1 --- labeled security protection
  - All data labeled, which restricts what can be done with it
  - This access control cannot be violated

B and A Divisions

- B2 --- structured protection
  - Adds covert channel protection onto B1
- B3 --- security domains
  - On top of B2 protection, adds that code must be tamperproof and "small"
- A --- verified protection
  - Like B3, but proved using formal methods
  - Such methods still impractical (usually)

Orange Book: Last Word

- Also a 2nd part, discusses rationale
- Not very practical or sensible, IMHO
- But some people insist we'd be better off if we'd followed it
- Others think it was a dead end
  - And resulted in lots of wasted effort
  - Aside: people who made the orange book, now set security education standards

Common Criteria

- Successor to the orange book (ca. 1998)
  - Due to inflation, more than 1000 pages
- An international government standard
  - And it reads like it…
  - Won't ever stir same passions as orange book
- CC is relevant in practice, but only if you want to sell to the government
- Evaluation Assurance Levels (EALs)
  - 1 thru 7, from lowest to highest security

EAL

- Note: product with high EAL may not be more secure than one with lower EAL
  - Why?
- Also, because product has EAL doesn't mean it's better than the competition
  - Why?

EAL 1 thru 7

- EAL1 --- functionally tested
- EAL2 --- structurally tested
- EAL3 --- methodically tested, checked
- EAL4 --- designed, tested, reviewed
- EAL5 --- semiformally designed, tested
- EAL6 --- verified, designed, tested
- EAL7 --- formally … (blah blah blah)

Common Criteria

- EAL4 is most commonly sought
  - Minimum needed to sell to government
- EAL7 requires formal proofs
  - Author could only find 2 such products…
- Who performs evaluations?
  - Government accredited labs, of course
  - For a hefty fee (like, at least 6 figures)

Authentication vs Authorization

- Authentication — Are you who you say you are?
  - Restrictions on who (or what) can access system
- **Authorization** — Are you allowed to do that?
  - Restrictions on actions of authenticated users
- Authorization is a form of **access control**
- Classic authorization enforced by
  - Access Control Lists (ACLs)
  - Capabilities (C-lists)

Lampson's Access Control Matrix

- **Subjects** (users) index the rows
- **Objects** (resources) index the columns

	OS	Accounting program	Accounting data	Insurance data	Payroll data
Bob	rx	rx	r	---	---
Alice	rx	rx	r	rw	rw
Sam	rwx	rwx	r	rw	rw
Accounting program	rx	rx	rw	rw	rw

Shynar Mussiraliyeva

Are You Allowed to Do That?

- **Access control matrix** has **all** relevant info
- Could be 1000's of users, 1000's of resources
- Then matrix with 1,000,000's of entries
- How to manage such a large matrix?
- Need to check this matrix before access to any resource is allowed
- How to make this efficient?

Access Control Lists (ACLs)

- ACL: store access control matrix by **column**
- Example: ACL for **insurance data** is in **blue**

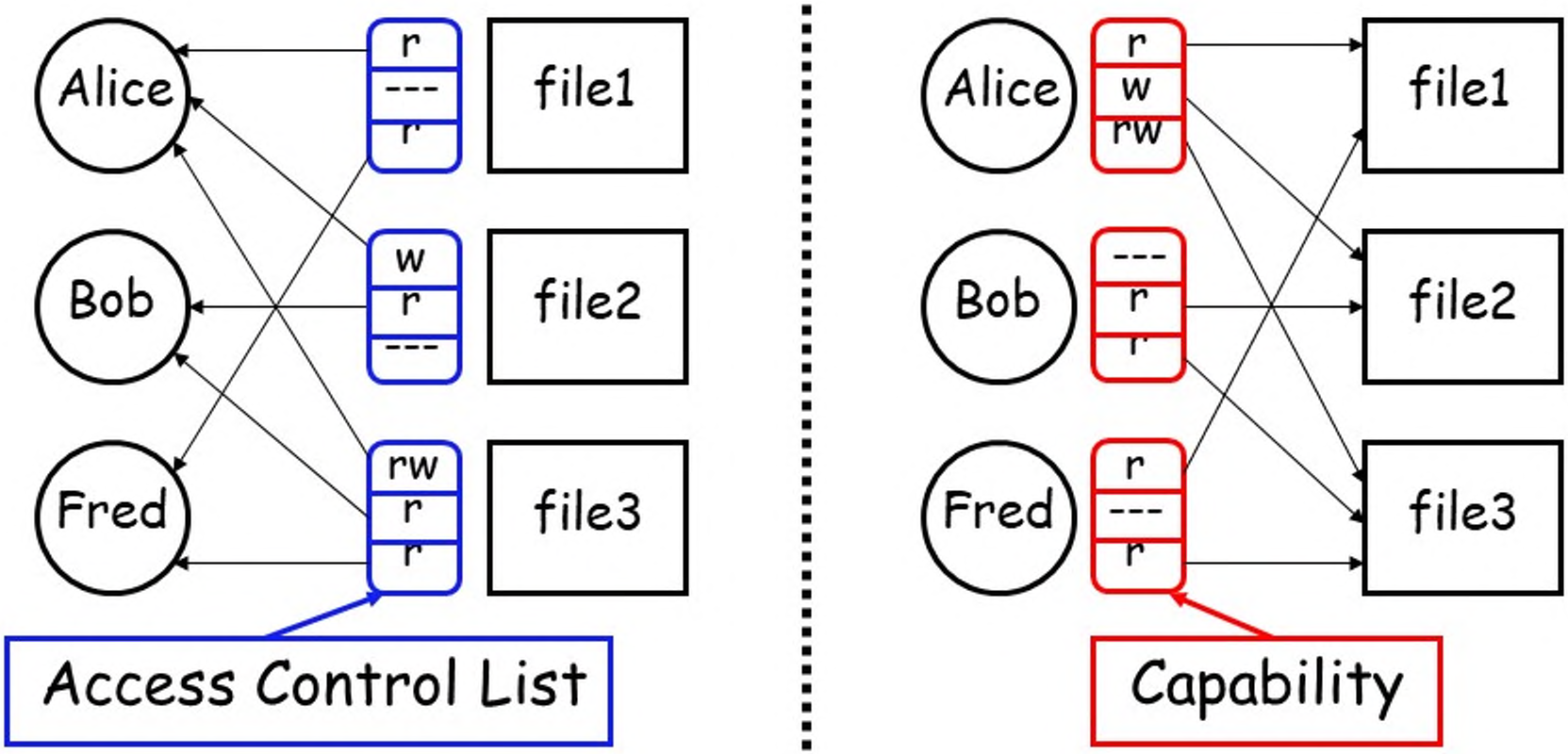
	OS	Accounting program	Accounting data	Insurance data	Payroll data
Bob	rx	rx	r	---	---
Alice	rx	rx	r	rw	rw
Sam	rwx	rwx	r	rw	rw
Accounting program	rx	rx	rw	rw	rw

Capabilities (or C-Lists)

- Store access control matrix by **row**
- Example: Capability for **Alice** is in **red**

	OS	Accounting program	Accounting data	Insurance data	Payroll data
Bob	rx	rx	r	---	---
Alice	rx	rx	r	rw	rw
Sam	rwx	rwx	r	rw	rw
Accounting program	rx	rx	rw	rw	rw

ACLs vs Capabilities



- Note that arrows point in opposite directions…
- With ACLs, still need to associate users to files

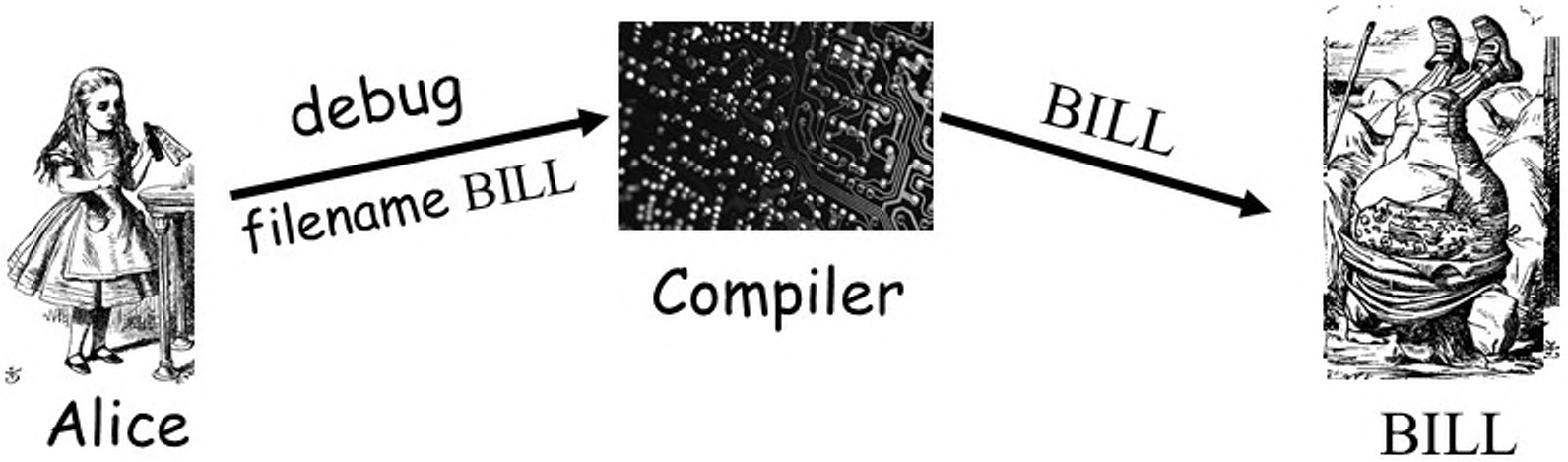
Confused Deputy

- Two resources
  - Compiler and BILL file (billing info)
- Compiler can write file BILL
- Alice can invoke compiler with a debug filename
- Alice not allowed to write to BILL

- Access control matrix

	Compiler	BILL
Alice	x	---
Compiler	rx	rw

ACL's and Confused Deputy



- Compiler is **deputy** acting on behalf of Alice
- Compiler is **confused**
  - Alice is not allowed to write BILL
- Compiler has confused its rights with Alice's

Confused Deputy

- Compiler acting for Alice is confused
- There has been a separation of **authority** from the **purpose** for which it is used
- With ACLs, difficult to avoid this problem
- With Capabilities, easier to prevent problem
  - Must maintain association between authority and intended purpose
  - Capabilities make it easy to **delegate** authority

ACLs vs Capabilities

- ACLs
  - Good when users manage their own files
  - Protection is data-oriented
  - Easy to change rights to a resource
- Capabilities
  - Easy to delegate---avoid the confused deputy
  - Easy to add/delete users
  - More difficult to implement
  - The "Zen of information security"
- Capabilities loved by academics
  - Capability Myths Demolished

Multilevel Security (MLS) Models

Shynar Mussiraliyeva

Classifications and Clearances

- **Classifications** apply to **objects**
- **Clearances** apply to **subjects**
- US Department of Defense (DoD) uses 4 levels:

  TOP SECRET

  SECRET

  CONFIDENTIAL

  UNCLASSIFIED

Clearances and Classification

- To obtain a **SECRET** clearance requires a routine background check
- A **TOP SECRET** clearance requires extensive background check
- Practical classification problems
  - Proper classification not always clear
  - Level of granularity to apply classifications
  - Aggregation — flipside of granularity

Subjects and Objects

- Let O be an **object**, S a **subject**
  - O has a classification
  - S has a clearance
  - Security **level** denoted L(O) and L(S)
- For DoD levels, we have

  TOP SECRET > SECRET >

  CONFIDENTIAL > UNCLASSIFIED

Multilevel Security (MLS)

- MLS needed when subjects/objects at different levels use/on **same system**
- MLS is a form of **Access Control**
- Military and government interest in MLS for many decades
  - Lots of research into MLS
  - Strengths and weaknesses of MLS well understood (almost entirely theoretical)
  - Many possible uses of MLS outside military

MLS Applications

- Classified government/military systems
- **Business example:** info restricted to
  - Senior management only, all management, everyone in company, or general public
- Network firewall
- Confidential medical info, databases, etc.
- Usually, MLS not a viable technical system
  - More of a legal device than technical system

MLS Security Models

- MLS models explain **what** needs to be done
- Models **do not** tell you **how** to implement
- Models are descriptive, not prescriptive
  - That is, high level description, not an algorithm
- There are many MLS models
- We'll discuss simplest MLS model
  - Other models are more realistic
  - Other models also more complex, more difficult to enforce, harder to verify, etc.

Bell-LaPadula

- BLP security model designed to express essential requirements for MLS
- BLP deals with **confidentiality**
  - To prevent unauthorized reading
- Recall that O is an object, S a subject
  - Object O has a classification
  - Subject S has a clearance
  - Security level denoted $L(O)$ and $L(S)$

Bell-LaPadula

- BLP consists of
  - **Simple Security Condition**: S can read O if and only if $L(O) \leq L(S)$
  - ***-Property (Star Property)**: S can write O if and only if $L(S) \leq L(O)$
- **No read up, no write down**

McLean's Criticisms of BLP

- McLean: BLP is "so trivial that it is hard to imagine a realistic security model for which it does not hold"
- McLean's "system Z" allowed administrator to reclassify object, then "write down"
- Is this fair?
- Violates spirit of BLP, but **not** expressly forbidden in statement of BLP
- Raises fundamental questions about the nature of (and limits of) modeling

B and LP's Response

- BLP enhanced with **tranquility property**
  - Strong tranquility: security labels never change
  - Weak tranquility: security label can only change if it does not violate "established security policy"
- Strong tranquility impractical in real world
  - Often want to enforce "least privilege"
  - Give users lowest privilege for current work
  - Then upgrade as needed (and allowed by policy)
  - This is known as the **high water mark** principle
- Weak tranquility allows for **least privilege** (high water mark), but the property is vague

BLP: The Bottom Line

- BLP is simple, probably too simple
- BLP is one of the few security models that can be used to prove things about systems
- BLP has inspired other security models
  - Most other models try to be more realistic
  - Other security models are more complex
  - Models difficult to analyze, apply in practice

Biba's Model

- BLP for confidentiality, Biba for **integrity**
  - Biba is to prevent unauthorized writing
- Biba is (in a sense) the dual of BLP
- Integrity model
  - Spse you trust the integrity of **O** but not **O**
  - If object **O** includes **O** and **O** then you cannot trust the integrity of **O**
- Integrity level of O is minimum of the integrity of any object in O
- **Low water mark** principle for integrity

Biba

- Let $I(O)$ denote the integrity of object O and $I(S)$ denote the integrity of subject S
- Biba can be stated as

  Write Access Rule: S can write O if and only if $I(O) \leq I(S)$

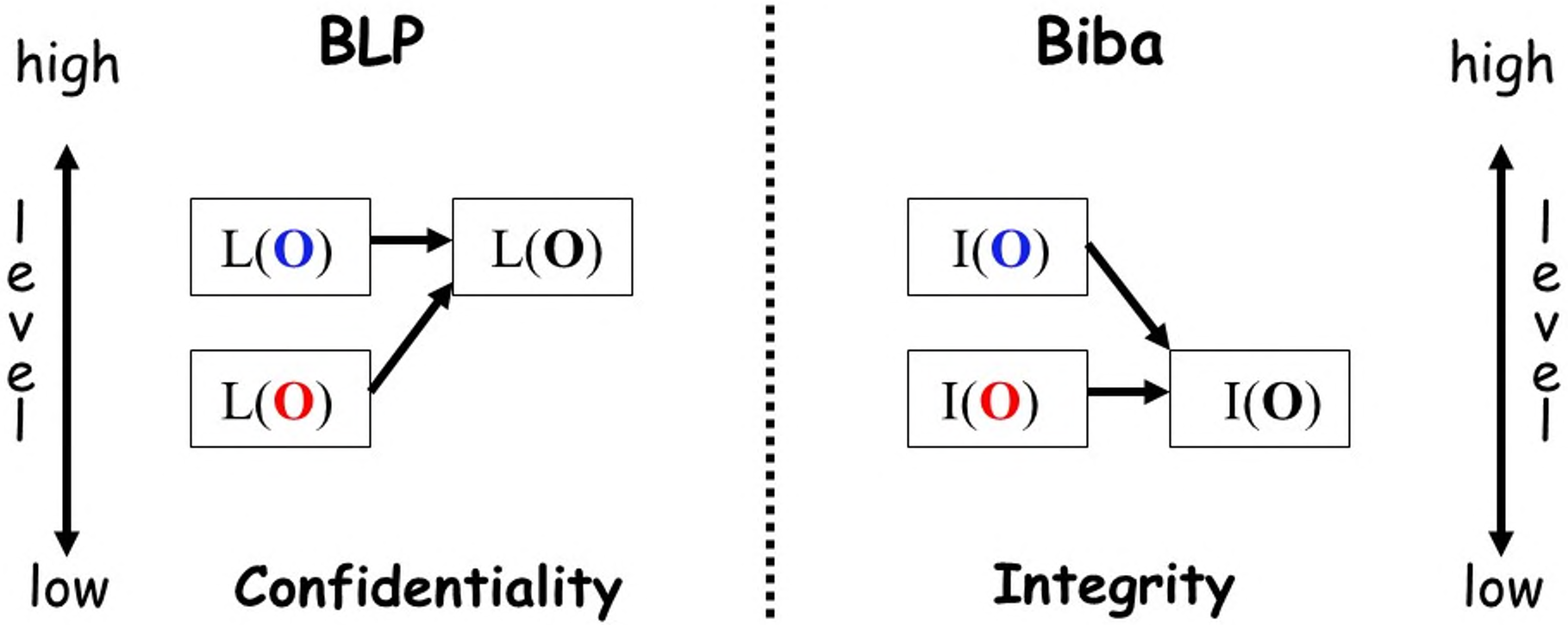
  (if S writes O, the integrity of $O \leq$ that of S)

  Biba's Model: S can read O if and only if $I(S) \leq I(O)$

  (if S reads O, the integrity of $S \leq$ that of O)
- Often, replace Biba's Model with

  Low Water Mark Policy: If S reads O, then $I(S) = \min(I(S), I(O))$

BLP vs Biba


BLP
high
level
low
L(O)
L(O)
L(O)
L(O)
Confidentiality
Biba
high
level
low
I(O)
I(O)
I(O)
I(O)
Integrity

Compartments

Shynar Mussiraliyeva

Compartments

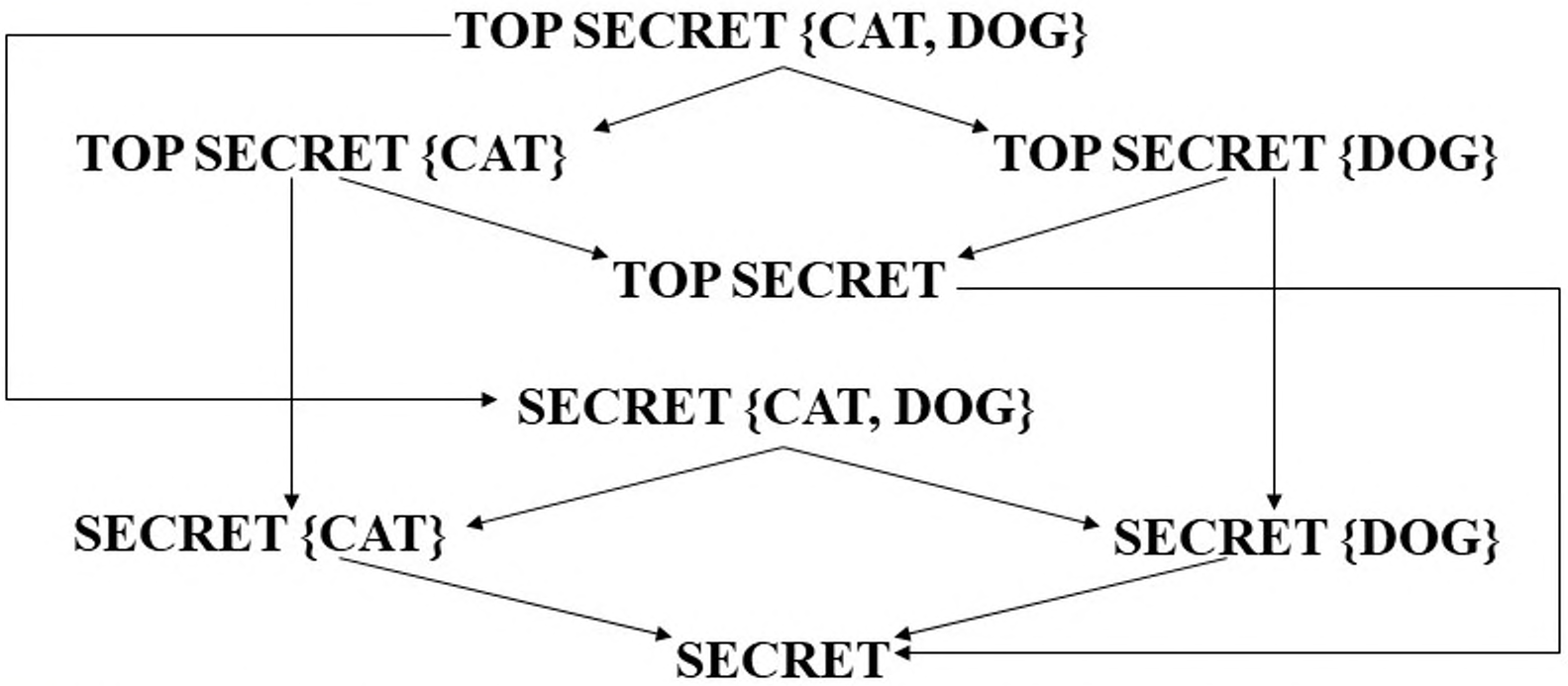
- Multilevel Security (MLS) enforces access control **up and down**
- Simple hierarchy of security labels is generally *not* flexible enough
- Compartments enforces restrictions **across**
- Suppose **TOP SECRET** divided into **TOP SECRET {CAT}** and **TOP SECRET {DOG}**
- Both are **TOP SECRET** but information flow restricted across the **TOP SECRET** level

Compartments

- Why compartments?
  - Why not create a new classification level?
- May not want either of
  - **TOP SECRET {CAT} ≥ TOP SECRET {DOG}**
  - **TOP SECRET {DOG} ≥ TOP SECRET {CAT}**
- Compartments designed to enforce the **need to know** principle
  - Regardless of clearance, you only have access to info that you need to know to do your job

Compartments

- Arrows indicate "≥" relationship



- Not all classifications are comparable, e.g.,

TOP SECRET {CAT} vs SECRET {CAT, DOG}

MLS vs Compartments

- MLS can be used without compartments
  - And vice-versa
- But, MLS almost always uses compartments
- Example
  - MLS mandated for protecting medical records of British Medical Association (BMA)
  - AIDS was **TOP SECRET**, prescriptions **SECRET**
  - What is the classification of an AIDS drug?
  - Everything tends toward **TOP SECRET**
  - Defeats the purpose of the system!
- Compartments-only approach used instead

Covert Channel

Shynar Mussiraliyeva

Covert Channel

- MLS designed to restrict legitimate channels of communication
- May be other ways for information to flow
- For example, resources shared at different levels could be used to "signal" information
- **Covert channel**: a communication path not intended as such by system's designers

Covert Channel Example

- Alice has **TOP SECRET** clearance, Bob has **CONFIDENTIAL** clearance
- Suppose the file space shared by all users
- Alice creates file FileXYzW to signal "1" to Bob, and removes file to signal "0"
- Once per minute Bob lists the files
  - If file FileXYzW does not exist, Alice sent 0
  - If file FileXYzW exists, Alice sent 1
- Alice can leak **TOP SECRET** info to Bob!

Covert Channel Example

Alice: Create file | Delete file | Create file | | Delete file

Bob: Check file | Check file | Check file | Check file | Check file

Data: 1 | 0 | 1 | 1 | 0

Time:

Shynar Mussiraliyeva

Covert Channel

- Other possible covert channels?
  - Print queue
  - ACK messages
  - Network traffic, etc.
- When does covert channel exist?
  1. Sender and receiver have a shared resource
  2. Sender able to vary some property of resource that receiver can observe
  3. "Communication" between sender and receiver can be synchronized

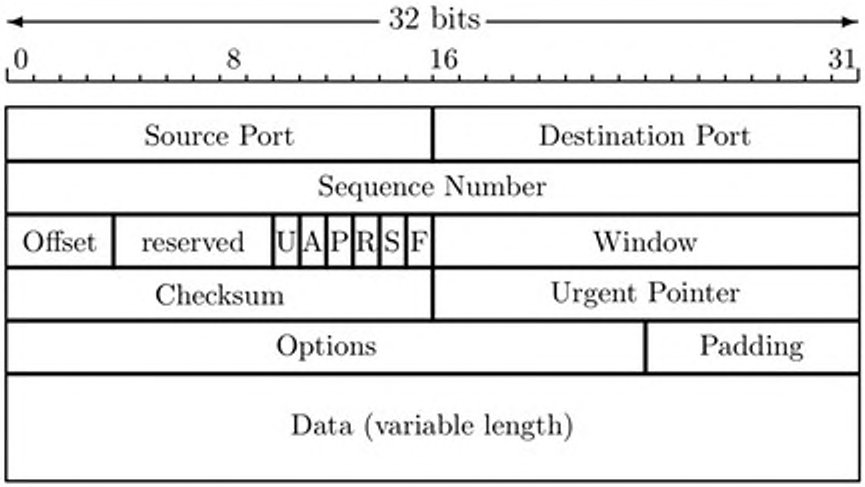
Covert Channel

- So, covert channels are everywhere
- "Easy" to eliminate covert channels:
  - Eliminate all shared resources…
  - …and all communication
- Virtually impossible to eliminate covert channels in any useful system
  - DoD guidelines: **reduce covert channel capacity** to no more than 1 bit/second
  - Implication? DoD has given up on *eliminating* covert channels!

Covert Channel

- Consider 100MB **TOP SECRET** file
  - Plaintext stored in **TOP SECRET** location
  - Ciphertext (encrypted with AES using 256-bit key) stored in **UNCLASSIFIED** location
- Suppose we reduce covert channel capacity to 1 bit per second
- It would take more than 25 years to leak entire document thru a covert channel
- But it would take less than 5 minutes to leak 256-bit AES key thru covert channel!

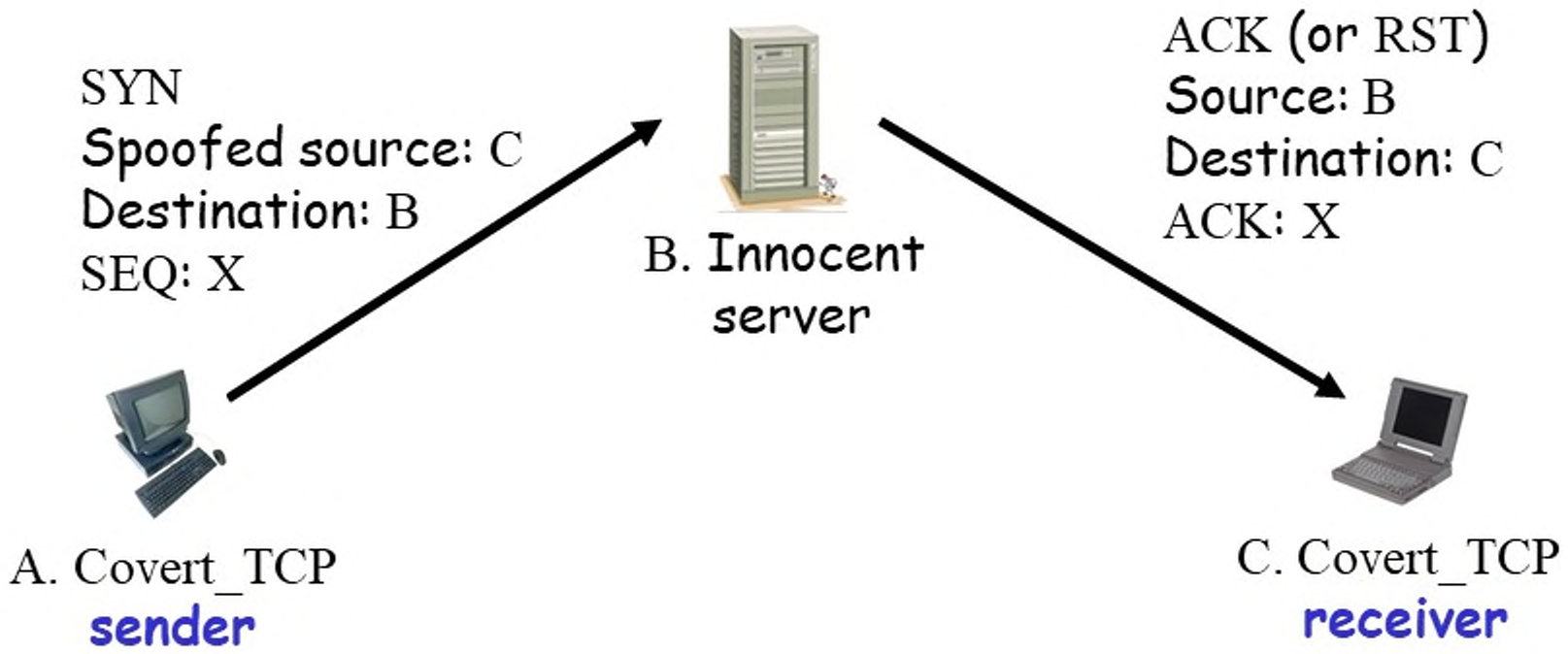
Real-World Covert Channel



- Hide data in TCP header "reserved" field
- Or use covert_TCP, tool to hide data in
  - Sequence number
  - ACK number

Real-World Covert Channel

- Hide data in TCP sequence numbers
- Tool: covert_TCP
- Sequence number X contains covert info



Inference Control

Shynar Mussiraliyeva

Inference Control Example

- Suppose we query a database
  - Question: What is average salary of female CS professors at SJSU?
  - Answer: $95,000
  - Question: How many female CS professors at SJSU?
  - Answer: 1
- Specific information has leaked from responses to general questions!

Inference Control and Research

- For example, medical records are private but valuable for research
- How to make info available for research and protect privacy?
- How to allow access to such data without leaking specific information?

Naïve Inference Control

- Remove names from medical records?
- Still may be easy to get specific info from such "anonymous" data
- Removing names is not enough
  - As seen in previous example
- What more can be done?

Less-naïve Inference Control

- Query set size control
  - Don't return an answer if set size is too small
- N-respondent, k% dominance rule
  - Do not release statistic if k% or more contributed by N or fewer
  - Example: Avg salary in Bill Gates' neighborhood
  - This approach used by US Census Bureau
- Randomization
  - Add small amount of random noise to data
- Many other methods — none satisfactory

Inference Control

- Robust inference control may be impossible
- Is weak inference control better than nothing?
  - **Yes**: Reduces amount of information that leaks
- Is weak covert channel protection better than nothing?
  - **Yes**: Reduces amount of information that leaks
- Is weak crypto better than no crypto?
  - **Probably not:** Encryption indicates important data
  - May be easier to filter encrypted data

Shynar Mussiraliyeva

CAPTCHA

Turing Test

- Proposed by Alan Turing in 1950
- Human asks questions to another human and a computer, without seeing either
- If questioner cannot distinguish human from computer, computer passes the test
- The **gold standard** in artificial intelligence
- No computer can pass this today
  - But some claim to be close to passing

CAPTCHA

- **CAPTCHA**
  - **C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part
- **A**utomated — test is generated and scored by a computer program
- **P**ublic — program and data are public
- **T**uring test to tell… — humans can pass the test, but machines cannot pass
  - Also known as **HIP** == **H**uman **I**nteractive **P**roof
- Like an inverse Turing test (well, sort of…)

CAPTCHA Paradox?

- "…CAPTCHA is a program that can generate and grade tests that it itself cannot pass…"
  - "…much like some professors…"
- Paradox — computer creates and scores test that it cannot pass!
- CAPTCHA used so that only humans can get access (i.e., no bots/computers)
- CAPTCHA is for **access control**

CAPTCHA Uses?

- Original motivation: automated bots stuffed ballot box in vote for best CS grad school
  - SJSU vs Stanford?
- Free email services — spammers like to use bots to sign up for 1000's of email accounts
  - CAPTCHA employed so only humans get accounts
- Sites that do not want to be automatically indexed by search engines
  - CAPTCHA would force human intervention

CAPTCHA: Rules of the Game

- Easy for most humans to pass
- Difficult or impossible for machines to pass
  - **Even with access to CAPTCHA software**
- From Trudy's perspective, the only unknown is a random number
  - Analogous to Kerckhoffs' Principle
- Desirable to have different CAPTCHAs in case some person cannot pass one type
  - Blind person could not pass visual test, etc.

Do CAPTCHAs Exist?

- Test: Find 2 words in the following

- Easy for most humans
- A (difficult?) OCR problem for computer
  - OCR == Optical Character Recognition

CAPTCHAs

- Current types of CAPTCHAs
  - Visual — like previous example
  - Audio — distorted words or music
- No text-based CAPTCHAs
  - Maybe this is impossible…

CAPTCHA's and AI

- OCR is a challenging AI problem
  - Hard part is the **segmentation problem**
  - Humans good at solving this problem
- Distorted sound makes good CAPTCHA
  - Humans also good at solving this
- Hackers who break CAPTCHA have solved a hard AI problem
  - So, putting hacker's effort to good use!
- Other ways to defeat CAPTCHAs???