



COMP2700: Cyber Security Foundations

Semester 2 - 2024

The Teaching Team

- Lecturer & convener: Michael Purcell
- Tutors:
 - Nicholas Arvanitellis
 - Harry Wu

Course Organization

- Format: 24x1.5h lectures + 10x1h labs.
 - Labs start on Week 1.
 - No Labs on Week 6 or Week 12
- Official course website on Wattle:
 - Course materials: lecture slides, labs/tutorials, assignments, quizzes.
- Discussions forum on EdStem
- Detailed schedule, links, etc., on Wattle

Course Organisation

- **Assessment:**
 - [MQ] Math prerequisite quiz (1%); available Week 1 – 5; multiple attempts allowed.
 - [LQ] 9 lab quizzes (9%); starting Lab 2 (Week 2).
 - [A1] Assignment 1 (25%) -- System and Software security. Due week 7.
 - [A2] Assignment 2 (25%) – Cryptography. Due week 12.
 - [E] Final exam (40%)
- Final exam is a **hurdle**: you must get at least 40% in the final exam.
- Failure at this hurdle may result in failure of the course.

Learning Outcome

- Demonstrate a thorough understanding of the fundamental principles underlying Cyber Security.
- Define and identify cyber security principles and their violation.
- Apply cyber security principles in a practical context.
- Migrate insights from cyber security analysis into new designs.
- Communicate a cyber security threat to a heterogenous team of professionals.

Scope

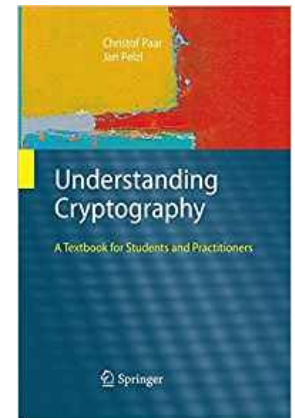
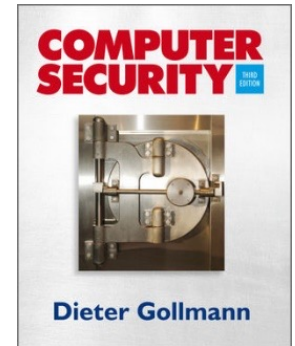
- This is a *technical* foundation course on cyber security.
 - But various non-technical aspects will be treated briefly.
- It covers a range of topics but with emphasis on:
 - Operating system security
 - Software security
 - Access control models
 - Applied cryptography and security protocols

Prerequisites

- COMP1100/COMP1130: familiarity with at least one imperative programming language:
 - Familiarity with C will be helpful, but some basic tutorials will be given in this course.
- COMP1600: basic discrete math
 - Sets, relations, (partial) order
 - Modular arithmetic (from first year math course(s)).
 - Ability to reason (informally) about program behaviours.

References

- Textbooks:
 - [Gollmann] D. Gollmann. Computer Security (3rd ed.), John Wiley & Sons, 2011.
 - [Paar] Christof Paar and Jan Pelzl. Understanding cryptography: a textbook for students and practitioners. Springer, 2010.
- Online copies available for free.
 - See Wattle course site for details.
- Limited physical copies available from the library.



References

Additional sources:

- Wenliang Du. Computer Security: a hands-on approach. CreateSpace, 2017.
- William Stallings. Cryptography and Network Security (7th edition). Pearson, 2017
- Ross Anderson. Security Engineering. Wiley, 2008.
- David Basin, Patrick Schaller, Michael Schlapfer. Applied Information Security: a hands-on approach. Springer 2011.
- Matt Bishop. Computer Security: Art and Science. Addison Wesley, 2018.
- Jon Erickson. Hacking: the art of exploitation (2nd edition). No Starch Press, 2008.
- Stephen G. Kochan and Patrick Wood. Shell Programming in Unix, Linux and OS X (4th edition). Addison-Wesley Professional, 2016.
- Research papers and online resources (to be provided later)

Topics	Week
Intro, security principles, security management	1,2
Identity & Authentication	2
Access Control	3
Reference monitors & hardware-based security	3
Unix security	4
Software security	4, 5
Intro to crypto	6
Stream & block ciphers	7
Encryption mode, hash, MAC	8
Public key crypto	9
Digital signatures	10
Key establishment	10,11
Overview of network security	11
Revision/guest lectures	12

Introduction to Cyber Security

COMP2700 Cyber Security Foundations

Slides prepared based on Chapter 1 and 3 of Gollmann's "Computer Security", Wiley, 2011

What is cyber security?

Cybersecurity: Prevention of damage to, protection of, and restoration of computers, electronic communications systems, electronic communications services, wire communication, and electronic communication, including information contained therein, to ensure its availability, integrity, authentication, confidentiality, and nonrepudiation.

NIST (<https://csrc.nist.gov/glossary>)

- There is a strong emphasis on computer and communication systems.
 - Traditionally this field is called *computer security*.
- But as more and more physical infrastructures are connected to computer and communication systems, the impact of cyber attacks are felt beyond the digital domain.

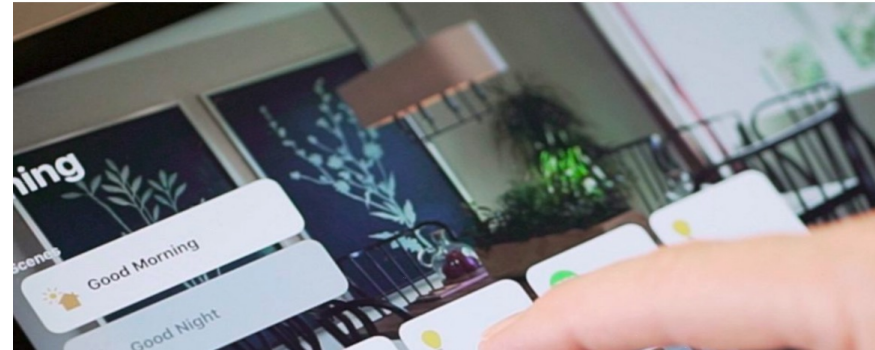
Cyberattacks (Confidentiality & Integrity)

- Cloud-based computing vulnerability:
 - **Spectre** and **Meltdown** (January 2018), potentially allows attackers on a cloud platform to read confidential data
- Bugs in implementation of OpenSSL:
 - 'Heartbleed' bugs (2014): allowing attacker to read memory contents of webserver
- Bugs in operating systems: allowing attacker to take over OS
 - Android OS: Quadrooter (2016), Stagefright (2014), Bluefrag (2020)
 - iOS: Homekit bug (2017) allowing attacker to control smart homes devices

DECEMBER 7, 2017

Zero-day iOS HomeKit vulnerability allowed remote access to smart accessories including locks, fix rolling out

Zac Hall - Dec: 7th 2017 1:03 pm PT [@apollozac](#)



Cyberattacks (Availability)

Smart homes or botnets?

By the Numbers: Are Your Smart Home Devices Being Used as Cryptocurrency Miners?

October 05, 2017



Most people are familiar with malware that encrypt files for ransom or attempts to steal information, but now cybercriminals are devoting resources to directly chasing cryptocurrency. This way, they can bypass any obstacles and directly go for a decentralized—and rapidly appreciating—currency that guarantees anonymity.

Could your 'smart' home be a weapon of web destruction?

By Monty Munford
Technology of Business reporter

Oct 28 October 2016



Do you use a webcam to check on Tiddles the cat or Bonzo the dog while you're at work?

If so, you could be unwittingly turning your internet-connected "smart" home into a weapon of web destruction.

Technology of Business

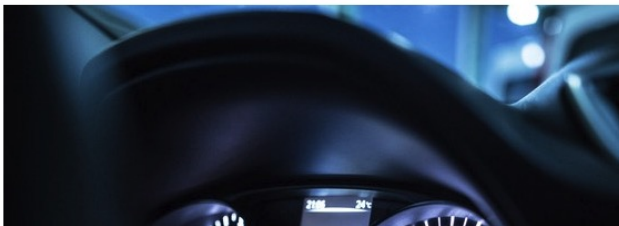
Has the time now come for internet voting?

Source: BBC. Oct 2017

Cyberattacks

ANDY GREENBERG SECURITY 08.16.17 04:55 PM

A DEEP FLAW IN YOUR CAR LET HACKERS SHUT DOWN SAFETY FEATURES



Their work points to a fundamental security issue in the CAN protocol that car components use to communicate and send commands to one another within the car's network, one that would allow a hacker who accesses the car's internals to shut off key automated components, including safety mechanisms.

"You could disable the air bags, the anti-lock brakes, or the door locks, and steal the car," says Federico Maggi, one of the Trend Micro researchers who authored the [paper](#). Maggi says the attack is stealthier than previous attempts, foiling even the few intrusion detection systems some companies like Argus and NNG have promoted as a way to head off car hacking threats. "It's practically impossible to detect at the moment with current technology," he says.¹

Source: Wired. 16/8/2017

Cyberattacks

TECH \ CYBERSECURITY \

UK hospitals hit with massive ransomware attack

Sixteen hospitals shut down as a result of the attack

By [Russell Brandom](#) | May 12, 2017, 11:36am EDT

The result has been a wave of canceled appointments and general disarray, as many hospitals are left unable to access basic medical records. At least one hospital has canceled all non-urgent operations as a result.

According to [a statement from the National Health Service](#), the culprit is a ransomware strain known as Wanna Decryptor (also known as WannaCry). While operations at the

Source: The Verge. 12/5/2017

Cyberattacks

Victorian hospitals across Gippsland, Geelong and Warrnambool hit by ransomware attack

ABC Radio Melbourne

Updated 1 Oct 2019, 7:53pm

The Victorian Government is investigating the scale of a ransomware attack by "sophisticated cyber criminals" on some of the state's major regional hospitals that has forced healthcare providers to go offline.

Hospitals in the Gippsland Health Alliance, in the state's east, and South West Alliance of Rural Health were impacted by the attack.



Source: [ABC News](#) – 1 Oct 2019

Why is security so hard?

“Security involves making sure things work, not in the presence of random faults, but in the face of an intelligent and malicious adversary trying to ensure that things fail in the worst possible way at the worst possible time ... again and again. It is truly programming Satan’s computer.”

-- Bruce Schneier

“Security engineering is about building systems to remain dependable in the face of malice, error, or mischance.”

-- Ross Anderson

Why is security so hard?

Gitlab 2021 Global Developer Report: DevSecOps

<https://about.gitlab.com/developer-survey/>

".. while 83% of security pros agreed at some level that finding bugs is a developer performance metric, nearly the same percentage (81%) complained it was difficult to get devs to make bug fixes a priority. In the end, 77% of security pros agreed at some level that bugs are mostly found by them (and not devs) after code is merged in a test environment."

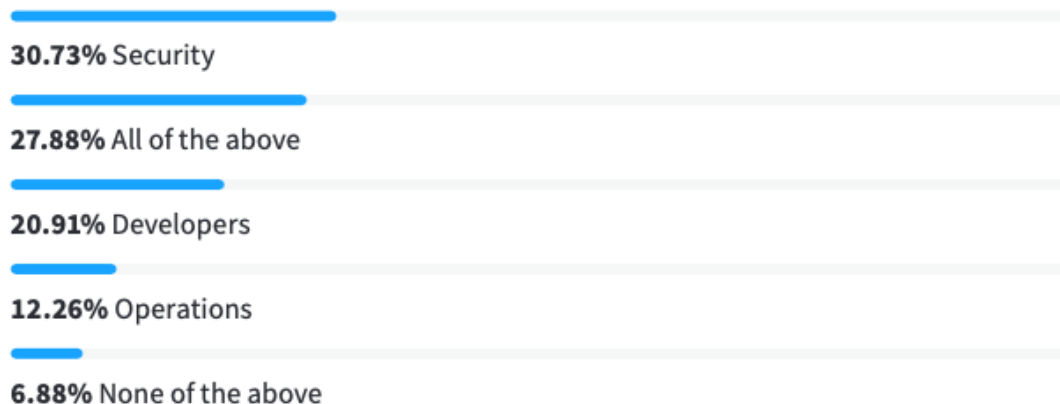
Why is security so hard?

Gitlab 2021 Global Developer Report: DevSecOps

<https://about.gitlab.com/developer-survey/>

- Confusion on who is responsible for security

IN YOUR ORGANIZATION, WHICH GROUP IS PRIMARILY RESPONSIBLE FOR SECURITY?



Cyber Security Skill Gaps

A recent survey conducted by the SANS Institute:

Skill area	Percent of cybersecurity job candidates who were unable to perform even basic tasks	Percent of cybersecurity job candidates who demonstrated hands-on mastery
Common exploitation techniques	66%	4.5%
Computer architectures	47%	12.5%
Networking	46%	4%
Linux	40%	14%
Programming	32%	11.5%
Data and cryptography	30%	2%

Source: [Krebs on Security](#) – 24 July 2020

Outline

- History of computer security: eras of computer security, characterized by:
 - Technology
 - Prevalent security issues
- Definitions of security
- Principles of computer security

Eras of computer security

- 1970s: age of the mainframe
- 1980s: age of the PC
- 1990s: age of the Internet
- 2000s: age of the Web
- 2010 and beyond: it's complicated

1970s: age of the mainframe

- Origin can be traced back to World War II; mainly concerned with cryptography (code breaking).
- Systematic study about *system security* started in the '70s; two prominent studies:
 - The Anderson report, on the design of computer systems.
 - James P. Anderson. "Computer Security Technology Planning Study" (1972)
 - The RAND report by Willis Ware, on the technical foundations of computer security.
 - Willis Ware. "Security Controls for Computer Systems: Report of Defense Science Board Task Force on Computer Security - RAND Report R-609-1" (1979).
- Primary concern was protection of classified information, and driven by military needs.

1970s: technology

- Advance on data storage (IBM 3340 “Winchester” disk) allows processing of ‘large’ amount of data (35-70 MB).
- Application: data crunching in large organisations and government departments.
- Target of protection:
 - classified data in the defence sector;
 - Unclassified but sensitive data, e.g., personal information in government departments.
- Security controls in the system core: operating systems, database management systems
- Computers and computer security managed by professionals.



IBM 3340 direct access storage facility

Source: IBM

1970s: security issues

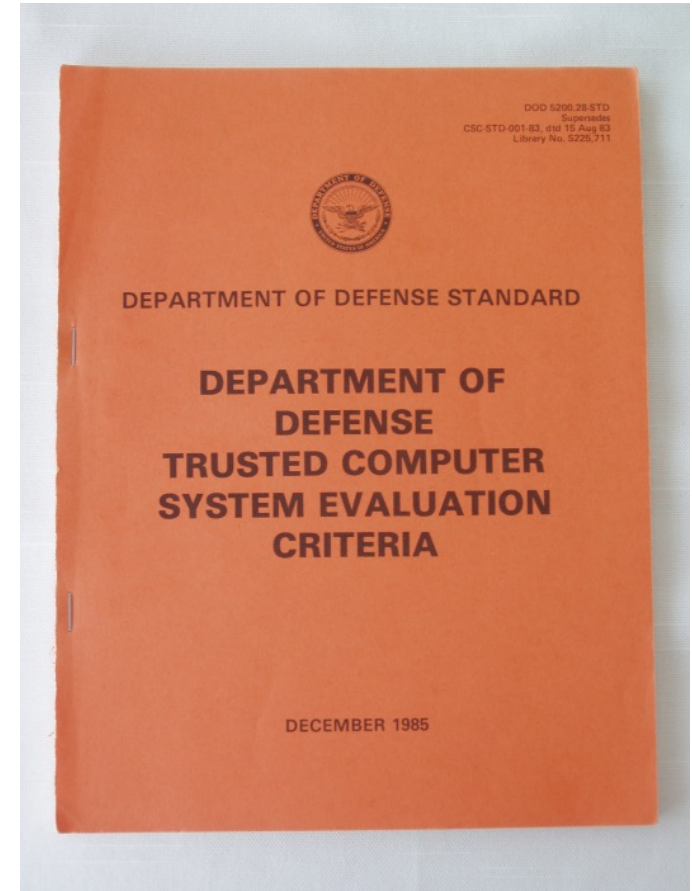
- Multi-level security (MLS) and the Bell LaPadula model of access control.
- Development of high assurance operating system (the Multics project).
- Cryptography: Data Encryption Standard (DES), public research on cryptography.
- Privacy legislation.
- Statistical database security: how to ensure combination of statistical queries (aggregate information) does not leak information.

1980s: age of the PC

- Technology: Personal Computer, GUI, mouse, ...
- Application: word processors, spreadsheets, i.e. software to support office work.
- Single-user machines processing unclassified data:
 - No need for multi-user security or for MLS.
- Emergence of security evaluation standards:
 - Orange Book (TCSEC, 1983/85).

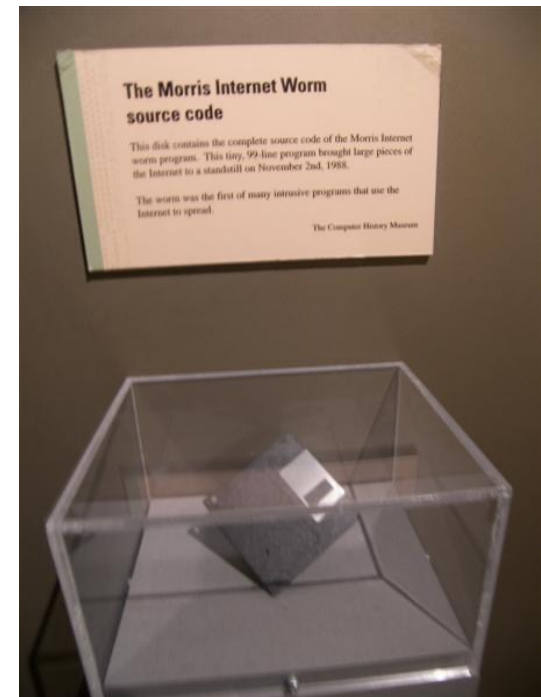
1980s: the Orange book

- Official title: Trusted Computer System Evaluation Criteria.
- Developed by the United States Department of Defense.
- Systematic definitions of security policies, levels of protection.
- Primary concern is protection of classified (defence) information.



1980s: Morris's Worm

- First worm developed by Robert Morris (now professor at MIT).
- Malicious codes that propagate through network, exploiting a buffer-overflow bug in *fingerd* daemon in Unix OS.
- Morris was fined and sentenced to 3 year probation and 400 hours community service.
 - First conviction under US computer Fraud and Abuse act.



1980s: security issues

- MLS is still actively researched; applications to database.
- New security models:
 - Clark-Wilson model: first commercially-driven security research.
 - Chinese Wall model: developed to manage conflicts of interests in financial consultancy business.
- Prevalence of worms and viruses.
- Development of x86 processor architecture, with memory protection support.

1990s: age of the internet

- Technology: Internet, commercially used, Web 1.0.
- Applications: Web surfing, email, entertainment, ...
- **No control on who can send what to a machine on the Internet.**
 - Significant departure from a standalone PC in the previous era, and gives rise to new security challenges.
- “Add-on” security controls: firewalls, SSL, browser sandbox as reference monitor, ...

1990s: security issues

- Development of strong cryptography and its widespread use:
 - Internet security treated as a communications security problem.
 - Endpoints (users' computers) still the weakest link.
- Buffer overrun attacks:
 - Aleph One: Smashing the Stack for Fun and Profit.
- Java security model: the sandbox.
- Trusted Computing; Digital Rights Management (DRM)
- Status today: mature security protocols (IPsec, SSL/TLS), better software security.

2000s:

- Technology: Web services, WLAN, Public Key Infrastructure (PKI)
- Web 2.0: user-generated content, web as collaborative environment.
- E-Commerce: Amazon, eBay, airlines, on-line shops, Google, ...
- Large scale cybercrime; hacking no longer a curiosity, but criminal enterprise.
- Legislation to encourage use of electronic signatures.

2010s: Some trends

Pervasive use of personal mobile devices:

Device Type	2016	2017	2018	2019
Traditional PCs (Desk-Based and Notebook)	220	204	193	187
Ultramobiles (Premium)	50	59	70	80
Total PC Market	270	262	264	267
Ultramobiles (Basic and Utility)	169	160	159	156
Computing Device Market	439	423	423	423
Mobile Phones	1,893	1,855	1,903	1,924
Total Device Market	2,332	2,278	2,326	2,347

Figures in millions of units. Source: www.gartner.com, January 2018.

2010s: Some trends

- Social media on the increase: As of July 2019
 - 3.534 billion active social media users,
 - 3.463 billion active mobile social users

Source: Global digital report 2019, wearesocial.com
- Web access patterns:
 - According Flurry Analytics, in 2015, 90% of users time on mobile is spent in apps; 10% spent on browser.
 - Apps are the preferred option to connect to mobile webservices.

2010s: Some trends (technology)

- Pervasive mobile computing, smart phones and tablets;
- Social media dominates internet usage;
- Apps replace more traditional content services (think of Facebook, WhatsApp, Amazon, TikTok, etc)
- Cloud computing
- Cyber-physical systems: connecting cyber worlds to physical worlds:
 - Smart grids, ‘internet of things’ (IoT), smart homes, etc.
- Blockchains:
 - Decentralised digital currency – enabling anonymised transactions

2010s: Some trends (security issues)

- Old threats still exist (and amplified in some cases).
 - Virus/malwares, web security, ransomware; data theft
- New threats: **privacy issues**
 - Large scale surveillance and data analytics.
 - For example, see the case of Cambridge Analytica.
https://en.wikipedia.org/wiki/Cambridge_Analytica.
 - Contact tracing technologies: apps/devices to track users' movements; public health vs privacy?
- Insecure implementation of communication protocols in apps ¹
- Short cycle of support for OS; large pool of unpatched devices.

¹ Fahl, et. al. Why eve and mallory love android: an analysis of android SSL (in)security. Proceedings of the 2012 ACM Conference on Computer and Communications Security.

Summary

- Distinct eras of security problems.
- Technologies/innovation creates new security problems.
- Intended use vs user innovations.
 - Use of technology doesn't always follow expectation.
- Disruptive technologies:
 - simple and cheap and adopted en masse, but often without adequate security mechanisms.

Foundations of Computer Security

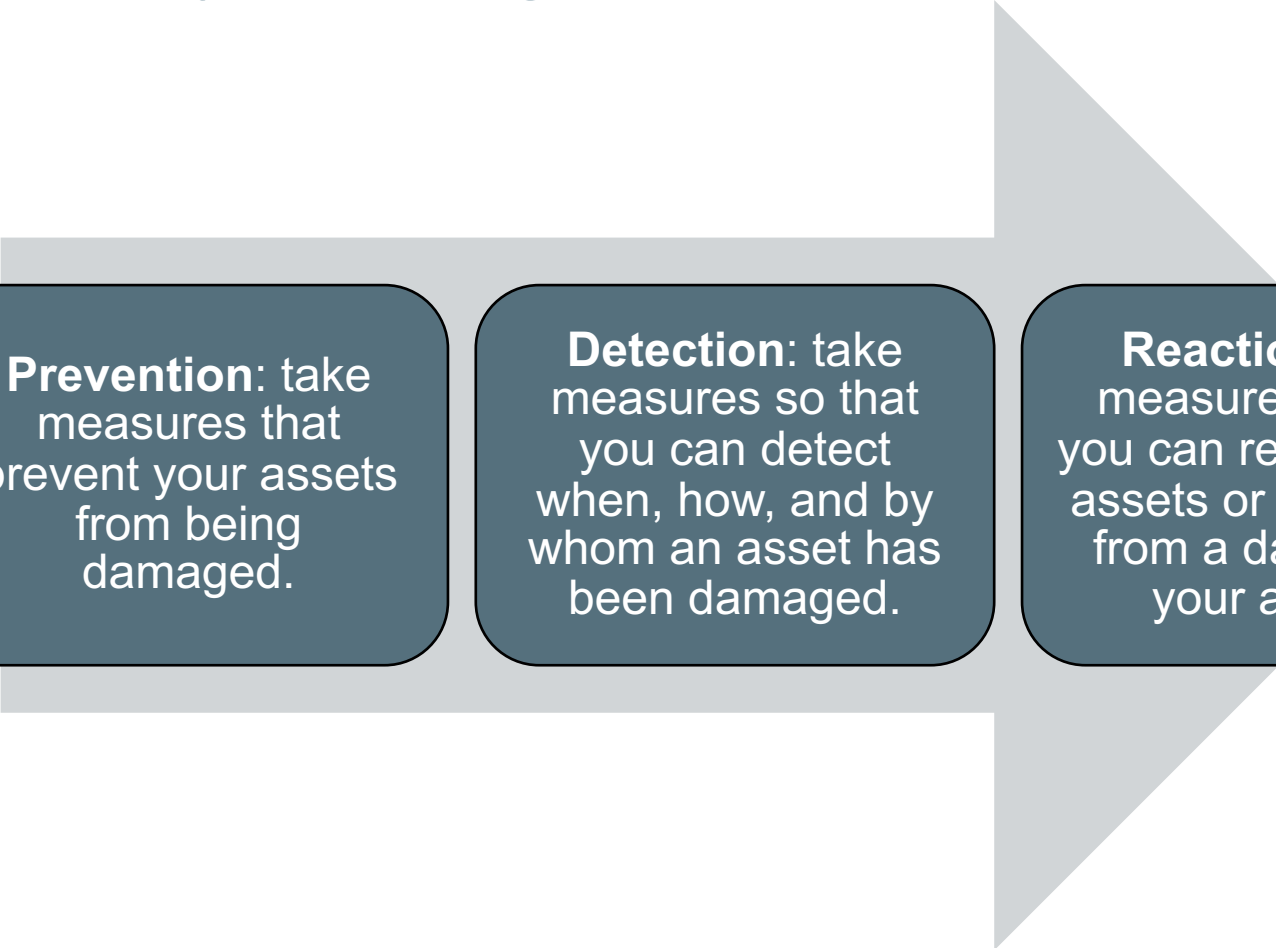
COMP2700 Cyber Security Foundations

Slides prepared based on Chapter 3 of Gollmann's "Computer Security", Wiley, 2011

Outline

- Security strategies: prevention, detection, reaction.
- Security objectives:
 - Confidentiality – integrity – availability
 - Accountability – non-repudiation
- Principles of Computer Security.

Security strategies



Prevention: take measures that prevent your assets from being damaged.

Detection: take measures so that you can detect when, how, and by whom an asset has been damaged.

Reaction: take measures so that you can recover your assets or to recover from a damage to your assets.

Example 1

Prevention:

- locks at doors,
- window bars,
- walls round the property.

Detection:

- stolen items are missing,
- burglar alarms,
- closed circuit TV.

Reaction:

- call the police,
- replace stolen items,
- make an insurance claim ...

Parallels to the physical world can illustrate aspects of computer security but they can also be misleading.

Example 2

Prevention:

- encrypt your orders,
- rely on the merchant to perform checks on the caller,
- two-factor authentication

Detection:

- an unauthorized transaction appears on your credit card statement.

Reaction:

- complain,
- ask for a new card number,
- etc.

Your credit card number has not been stolen; your card can be stolen, but not the card number.

Security Objectives

Confidentiality:

- prevent unauthorised disclosure of information

Integrity:

- prevent unauthorised modification of information

Availability:

- prevent unauthorised withholding of information or resources

Authenticity:

- “know whom you are talking to”

Accountability (non-repudiation):

- prove that an entity was involved in some event

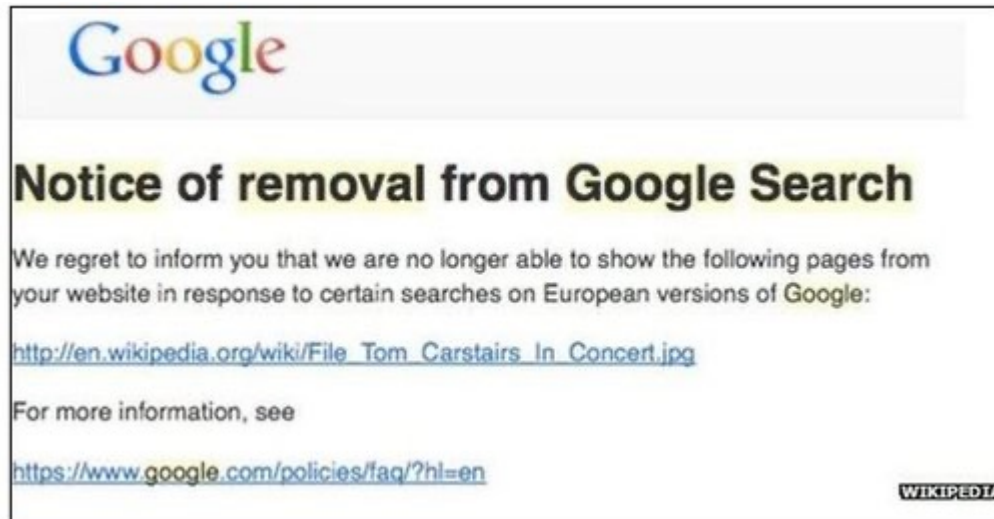
Confidentiality

- Prevent unauthorised disclosure of information (prevent unauthorised **reading**).
- More generally, unauthorized users should not **learn** sensitive information.
- **Secrecy**: protection of data belonging to an organisation/individual.
- Confidentiality can mean more than just secrecy:
 - One may want to hide not just secrets, but also their existence.
 - Traffic analysis, “meta-data”, can reveal sensitive information.
 - Anonymity and unlinkability; in general privacy-related properties.

Privacy

- **Privacy**: protection of personal data (OECD Privacy Guidelines, EU Data Privacy Directive 95/46/EC).
 - “Put the user in control of their personal data and of information about their activities.”
- Taken now more seriously by companies that want to be trusted by their customers.

Privacy



In May 2014, European Court of Justice ruled against Google, for the “right to be forgotten”.

Google was legally required to remove search results related to a certain individual.

Privacy

A major milestone: The EU General Data Protection Regulation (GDPR) 2018

“A consent request needs to be presented in a **clear** and **concise** way, using language that is easy to understand, and be **clearly distinguishable from other pieces of information such as terms and conditions**. The request has to specify what use will be made of your personal data and include contact details of the company processing the data.”

Source: European Commission website ec.europa.eu, 2018.

Integrity

- Prevent unauthorised modification of information (prevent unauthorised **writing**).
- Data Integrity:
 - The state that exists when computerized data is the same as that in the source document and has not been exposed to accidental or malicious alteration or destruction.
- Detection (and correction) of intentional and accidental modifications of transmitted data.
 - Typically enforced via (cryptographic) checksums and other coding techniques.

Integrity

Clark & Wilson: no user of the system, even if authorized, may be permitted to modify data items in such a way that assets or accounting records of the company are lost or corrupted.

Integrity

Integrity is a prerequisite for many other security services;

- In operating systems, integrity of the bootstrap process (kernel, device drivers, system files) is critical to prevent persistent viruses/malwares.
- Windows Vista and above allows only “signed drivers” to be installed.
- Signed OS kernels and drivers are now a standard practice in all major OSes (Windows, Mac OS, Linux, Android, iOS).

Availability

- The property of being accessible and usable upon demand by an authorised entity.
- **Denial of Service (DoS)**: prevention of authorised access of resources or the delaying of time-critical operations.
- Maybe the most important aspect of computer security, but few methods are around.
- Distributed denial of service (DDoS) receives a lot of attention; systems are now designed to be more resilient against these attacks.

Accountability

- Accountability is concerned with attributions of an action (attack) to an entity (user).
- To be effective, one needs:
 - Audit trails: eg, in the OS level, this could be system/authentication logs, etc.
 - A link between a user and a “user identity”, so the user can be held accountable.
- In distributed systems, cryptographic **non-repudiation** mechanisms can be used to achieve the same goal.

Non-repudiation

- Non-repudiation services provide **unforgeable evidence** that a specific action occurred.
- **Non-repudiation of origin**: protects against a sender of data denying that data was sent.
- **Non-repudiation of delivery**: protects against a receiver of data denying that data was received.
- Enforcement relies on public-key crypto techniques.

Reliability and Safety

- Reliability and safety are related to security:
 - Similar engineering methods,
 - Similar efforts in standardisation,
- **Reliability** addresses the consequences of accidental errors.
- **Safety**: measure of the absence of catastrophic influences on the environment, in particular on human life.

Safety vs Security

- Safety requirement: door must unlock when car is flipped.
- Solution: install pressure sensor on the roof of car; unlock when pressure is detected
- Security issue: unauthorized entry by jumping on the roof of the car

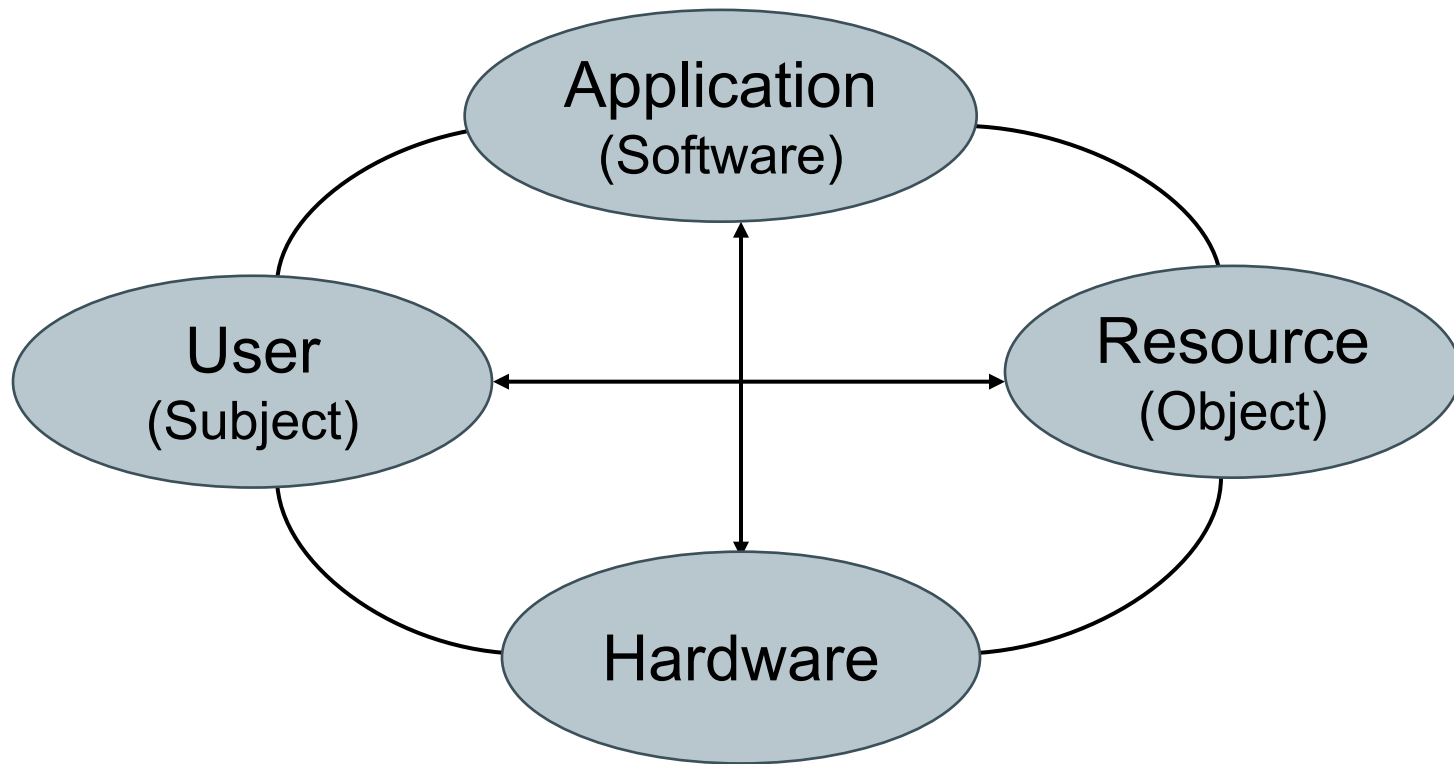


Courtesy of Vivek Nigam (Fortiss), reproduced with permission.

Security and Reliability

- On a PC, you are in control of the software components sending inputs to each other.
- On the Internet, hostile parties provide input.
- To make software more **reliable**, it is tested against typical usage patterns:
 - “It does not matter how many bugs there are, it matters how often they are triggered.”
- To make software more **secure**, it has to be tested against ‘untypical’ usage patterns (but there are typical attack patterns).

Dimension of Computer Security



Design Decisions

- I. Where to focus the protection mechanism?
- II. Where to place the security mechanism?
- III. Complexity (of security properties) vs assurance
- IV. Centralized vs decentralized security control
- V. Protection of the 'layer below'

I. Where to Focus Control?

- Focus on: Data, Operations, or Users?
- Example: integrity requirements may focus on:
 - Format and content of data items (internal consistency): account balance is an integer.
 - Operations that may be performed on a data item: credit, debit, transfer, ...
 - Users who are allowed to access a data item (authorised access): account holder and bank clerk have access to account.

II: Where to Place Security Control?

Application

Services (middleware)

Operating System

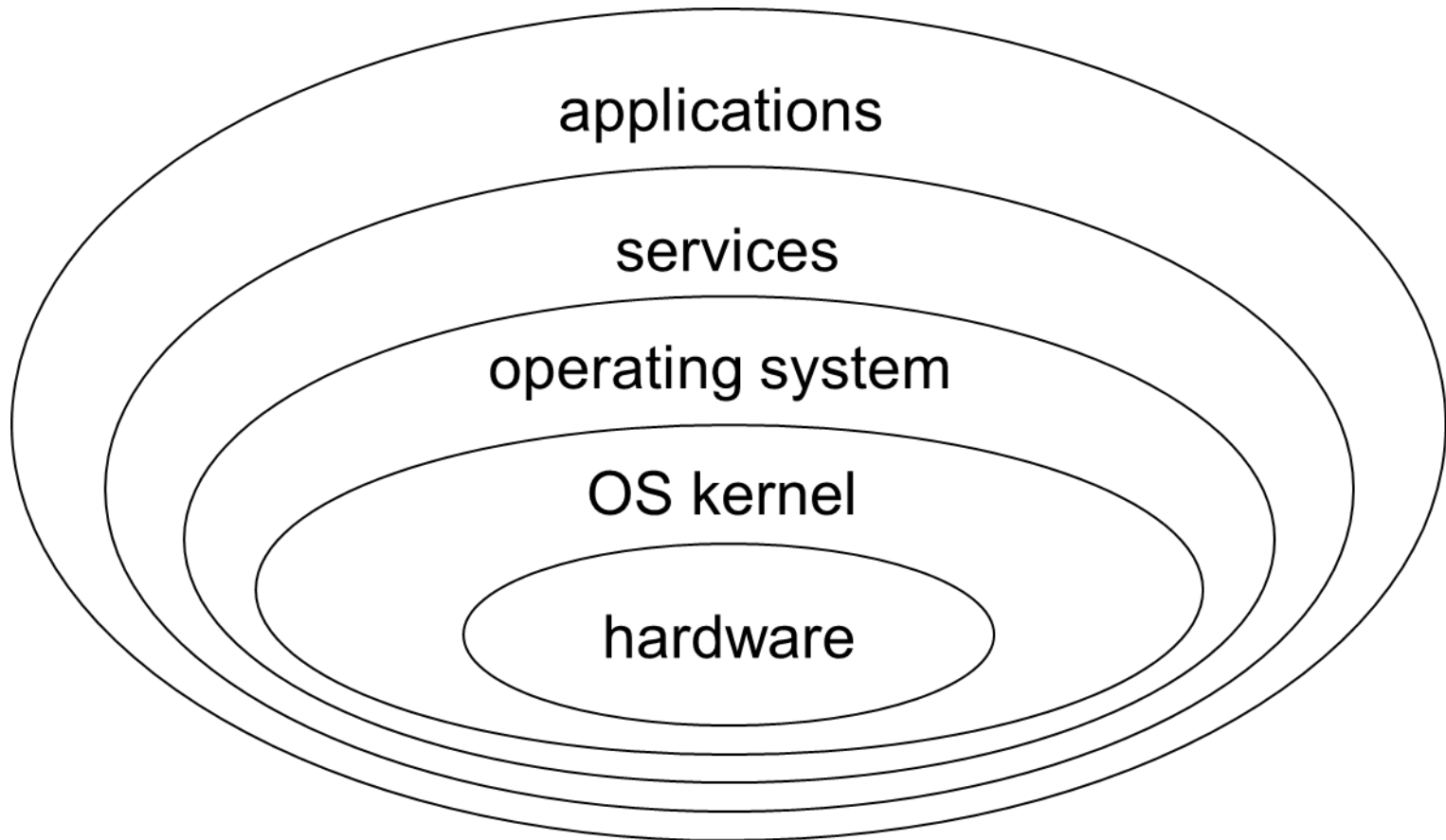
OS kernel

Hardware

Human-machine Scale

- Visualize security mechanisms as concentric **protection rings**, with hardware mechanisms in the centre and application mechanisms at the outside.
- Mechanisms towards the centre tend to be more generic, while mechanisms at the outside are more likely to address individual user requirements.
- The **human-machine scale** for security mechanisms combines our first two design decisions.

Onion Model of Protection



Human-Machine Scale

Specific
Complex
Focus on users



Human
oriented

Generic
Simple
Focus on data



Machine
oriented



Data vs Information

- **Data** are a representation certain aspects of our conceptual and real world.
- The meanings we assign to data are called **information**.
- Information and data lie on the two ends of the man-machine scale.
- The distinction between data and information can be subtle but causes some of the more difficult problems in computer security.

Data vs Information

- Controlling access to **information** may be elusive and need to be replaced by controlling access to **data**.
- But controlling data may not always yields control of information.
- **Side channels**: response time or memory usage may signal information.
- **Inference in statistical databases**: combine statistical queries to get information on individual entries.

Example: data vs information

```
int compute() {  
    int secret;  
    secret = readSecretData(..);  
  
    if (secret == 1) {  
        // do complicated computations  
        // and return  
        ...  
        return 0;  
    }  
    else  
        return 0;  
}
```

Suppose an attacker invokes the function compute().

Is there any secret data read by the attacker?

Does the attacker **learn** any **information** about the secret?

Example: data vs information

```
int compute() {  
    int secret;  
    secret = readSecretData(..);  
  
    if (secret == 1) {  
        // do complicated computations  
        // and return  
        ...  
        return 0;  
    }  
    else  
        return 0;  
}
```

Side channel: the attacker can observe the execution time of the function. Repeated observations would give the attacker some knowledge about secret.

Complex operations, likely require more time than the other branch of if.

Simple operation. Likely requires less time than the other branch.

III: Complexity vs Assurance

- Often, the location of a security mechanism on the man-machine scale is related to its complexity.
- Generic mechanisms are simple, applications clamour for **feature-rich** security functions.
- Do you prefer simplicity – and higher assurance – to a feature-rich security environment?

Centralised vs Decentralised Control

- Within the domain of a security policy, the same controls should be enforced.
- Centralized control:
 - + One entity in charge; easy to achieve uniformity.
 - Performance bottleneck.
- Distributed control:
 - + More efficient but,
 - Need to make sure policy consistency across different components.

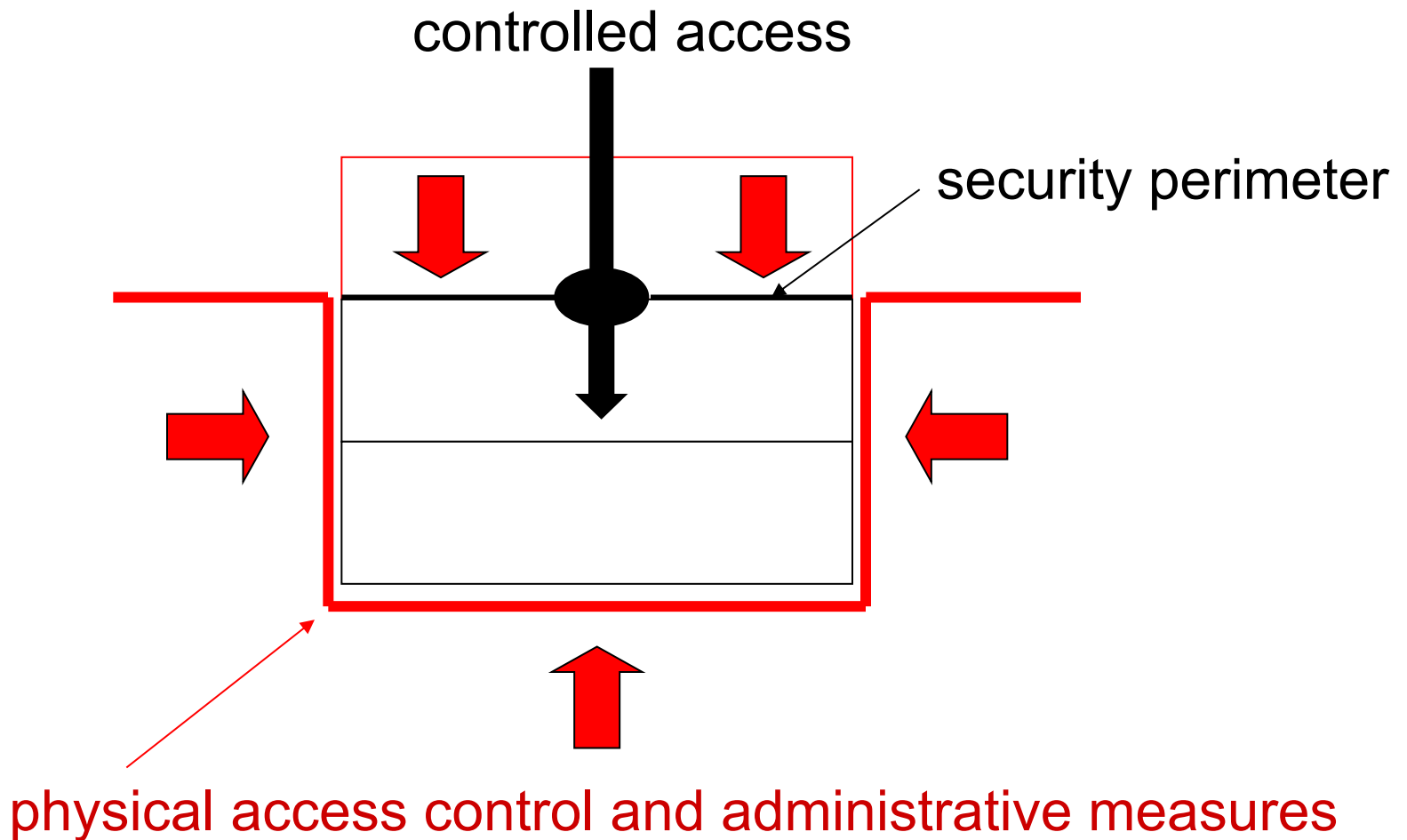
Security Perimeter

- Every protection mechanism defines a security perimeter (security boundary).
- The parts of the system that can malfunction without compromising the mechanism lie outside the perimeter.
- The parts of the system that can disable the mechanism lie within the perimeter.
- Note: Attacks from insiders are a major concern in security considerations.

IV: Protection of the layer below

- Attackers try to bypass protection mechanisms.
- There is an immediate and important corollary to the second design decision:
 - How do you stop an attacker from getting access to a layer below your protection mechanism?

Access to the layer below



Examples: access to the layer below

- **Forensic tools:**
 - restore data by reading memory/disks
 - circumvent logical access control as it does not care for the logical memory structure.
- **File access vs disk access:**
 - In Unix, permissions to disks can be assigned to users.
 - A user who does not have access to file A, but have access to the device the file is hosted can bypass the file protection mechanism (by reading raw bits from the device).
- **Buffer overruns:**
 - a value assigned to a variable is too large for the memory buffer allocated to that variable; memory allocated to other variables is overwritten.

Examples: access to the layer below

- **Object reuse:**
 - in single processor systems, when a new process is activated it gets access to memory positions used by the previous process. Avoid storage residues, i.e. data left behind in the memory area allocated to the new process.
- **Backup:**
 - whoever has access to a backup tape has access to all the data on it.
- **Core dumps and system logs:** may contain sensitive data.

Summary

- Security terminology is ambiguous with many overloaded terms.
 - Need to be clear about security objectives
- Security is not the same as safety
- In designing secure systems, decisions must be made as to which layer the protection should focus on, and protect the 'layer below'.

Security Principles

COMP2700 Cyber Security Foundations

Slides prepared based on Chapter 1 of 'Basin, Schaller, and Schlapfer. Applied Information Security: a hands-on approach. Springer 2011.'

Security Principles

- There are no hard-and-fast rules in security engineering.
- Some patterns of `best practices' are nevertheless useful.
- These principles may be applicable across different system layers and applications.
- We'll see specific instances of some of these principles later in the course.

#1 Simplicity

Keep it simple.

- Simpler systems are less likely to contain flaws than complex ones.
- Simpler systems are easier to analyse and review, thus easier to establish *trustworthiness*.
- Example: the concept of microkernel in operating system design. Its simplicity enables *formal verification* of correctness, e.g., Data61 seL4 microkernel.

#2 Open Design

The security of a system should not depend on the secrecy of its protection mechanisms.

- In cryptography, also known as the Kerckhoff's principle:
 - a cryptosystem should be secure even if all aspects of the system (except the keys being used) are public knowledge.
- Secrets are hard to protect, so need to be reduced to minimum.
- Opposite of *security-by-obscurity*.

#3 Compartmentalisation

Organize resources into isolated groups of similar needs.

- Facilitates simplification: compartments contain resources with similar needs
- Isolation of attacks: reduces the effects of attacks and provides mechanism for tackling it.
- Examples: kernel vs user-mode separation in OS, 'sandboxing' in virtual machines,

#4 Minimum exposure

Minimize the attack surface a system presents to the adversary

- Reduce external interfaces to a minimum
- Limit the amount of information given away
- Minimize the window of opportunity for an adversary, e.g., by limiting the time available for an attack.
- Example: disable unnecessary network services (NFC, Bluetooth, etc), minimize open ports in server, enforces session timeouts in webserver, etc.

#5 Least Privilege

Any component (and user) of a system should operate using the least set of privileges to complete its job.

- “Need-to-know” principle in access control (we’ll come back to this when discussing security models)
- Example:
 - Most office users do not need the privilege to install new software in a corporate computer.
 - A webserver process needs not run with administrative privilege.

#6 Minimum Trust and Maximum Trustworthiness

Minimize trust and maximize trustworthiness

- Trusted system vs trustworthy system:
 - A trusted system/component is one whose failure can break the security policy.
 - A trustworthy system/component is one that is unlikely to fail.
 - A trusted system is not necessarily trustworthy!
- In general, trust should be avoided whenever possible.
- Trust is transitive: a system's behavior can depend on other systems that are related to it by a chain of trust.
 - E.g., chain of trusts in SSL/TLS certificate validation.

#7 Secure, Fail-safe Defaults

The system should start in and return to a secure state in the event of a failure.

- In access control mechanism, the default and the fail-safe state should prevent any access.
 - E.g., a whitelist approach: permission is denied unless explicitly granted.
- Example: most firewall rules follow a whitelist approach; the default is to deny any network packet.

#8 Complete mediation

Access to any object must be monitored and controlled.

- Each access to every security-relevant object within a system must be controlled.
 - This includes all operational states: normal operation, shutdown, maintenance and failure mode.
- Beware of the “layer-below”. Examples:
 - memory management unit (MMU) enforces range checks on memory access requests.
 - ‘Canonicalization’ of data: this deals with issues of different representation of data used in access control. (We’ll come back to this in Software Security lectures).

#9 No Single Point of Failure

Build redundant security mechanisms whenever possible

- Security should not rely on a single mechanism.
- Also known as 'defense in depth'.
- The degree of redundancy must be determined on the basis of a cost-benefit analysis.
- Example: two-factor authentication.

#10 Traceability

Log security-relevant system events

- Traceability requires the system retains traces of activities.
- Need to ensure backup of logs and the integrity of logs, e.g., using tamper-resistant device, or cryptography.

#11 Generating secrets

Maximize the entropy of secrets.

- This helps prevent brute-force attacks, dictionary or guessing attacks.
- Example: use a good pseudorandom number generator to generate secret keys.
 - We'll see specific examples of breaking non-cryptographic random number generators in the second half of the course.

#12 Usability

Design usable security mechanisms

- Security mechanisms should be easy to use.
 - Otherwise (non-malicious) users will be motivated to circumvent it to get the job done.
- Example:
 - Most users don't understand cryptographic mechanisms, so it is possible they ignore warnings related to these mechanisms, e.g., invalid SSL certificates in web browsers.

Discussion

- Most real systems do not adhere to all principles.
- These principles can be contradictory in some situations.
- These principles are not independent; they may intersect and may be in conflict.



Security Management

Slides prepared based on Chapter 2 of Gollmann's "Computer Security", Wiley, 2011

Outline

- Security management
 - Security policies
 - Security metrics
 - Management standards
- Risk & Threat Analysis
 - Vulnerabilities
 - Threats
 - Risk
 - Baseline protection

Security Management

- Security is a people problem.
 - Cannot be solved by technology alone
 - Ultimately a responsibility of the management.
- Security measures may restrict working patterns of workers.
 - Management needs to define clear objectives of security (*security policies*).
 - Security-awareness training needs to be part of the management process.

Security policies

- Security policy: a statement that defines the security objectives of an organization.
- It has to state what needs to be protected; it may also indicate how this is to be done.
- Example: a policy may regulate
 - access to documents, e.g., clearance level in military
 - that certain transactions must be signed off by more than one person
 - Password formats and renewal intervals

Measuring Security

- Security measurement facilitates decision making:
 - Useful for comparing new security products or mechanisms.
- First step: obtain values for security relevant factors (security measurement).
- Second step: consolidate measurements into a single value that is used for comparing the current security state against a baseline or a past state (security metrics).

Security Metrics

- A security metric gives a quantitative statement about the security of a product or system.
 - **Product**: a package of software, firmware and/or hardware,
 - **System**: a specific IT installation, with a particular purpose and operational environment.
- Security metrics for a product: number of bugs detected, or attack surface, number of dangerous instructions in the code.
- Do these measurements really measure security?
- Secure products can be deployed in insecure ways!

Security Metrics

- Security metrics for a system: check configurations of the products deployed; number of privileged users; number of open ports; etc.
- Alternatively measure the cost of mounting attacks:
 - Time an attacker has to invest in the attack.
 - Expenses the attacker has to incur.
 - Knowledge necessary to conduct the attack.
- Another alternative: focus on the assets in the system and measure the risks these assets are exposed to.

Management Standards

- Prescriptive security management standards that stipulate which security measures have to be taken in an organisation; exist for specific industry branches.
 - Example: regulations for the financial sector, or rules for dealing with classified material in government departments.
- For Australian government: Information Security Manual (ISM) produced by the Australian Signal Directorate.
- Other management standards are best described as **code of best practice** for security management.
- The most prominent of these standards is **ISO 27002**.

Risk Analysis

- Many areas of engineering and business have developed their own disciplines and terminology for risk analysis.
- Within IT security, risk analysis is being applied
 - comprehensively for all information assets of an enterprise,
 - specifically for the IT infrastructure of an enterprise,
 - during the development of new products or systems, e.g. in software security.
- Informally, risk is the possibility that some incident or attack can cause damage to your enterprise.

Attacks

- An attack against an IT system is a sequence of actions, exploiting weak points in the system until the attacker's goals have been achieved.
- To assess the risk posed by an attack we have to evaluate the amount of damage being done and the likelihood for the attack to occur.
- This likelihood will depend on the attacker's motivation and on how easy it is to mount the attack.
- In turn, this will further depend on the security configuration of the system under attack.

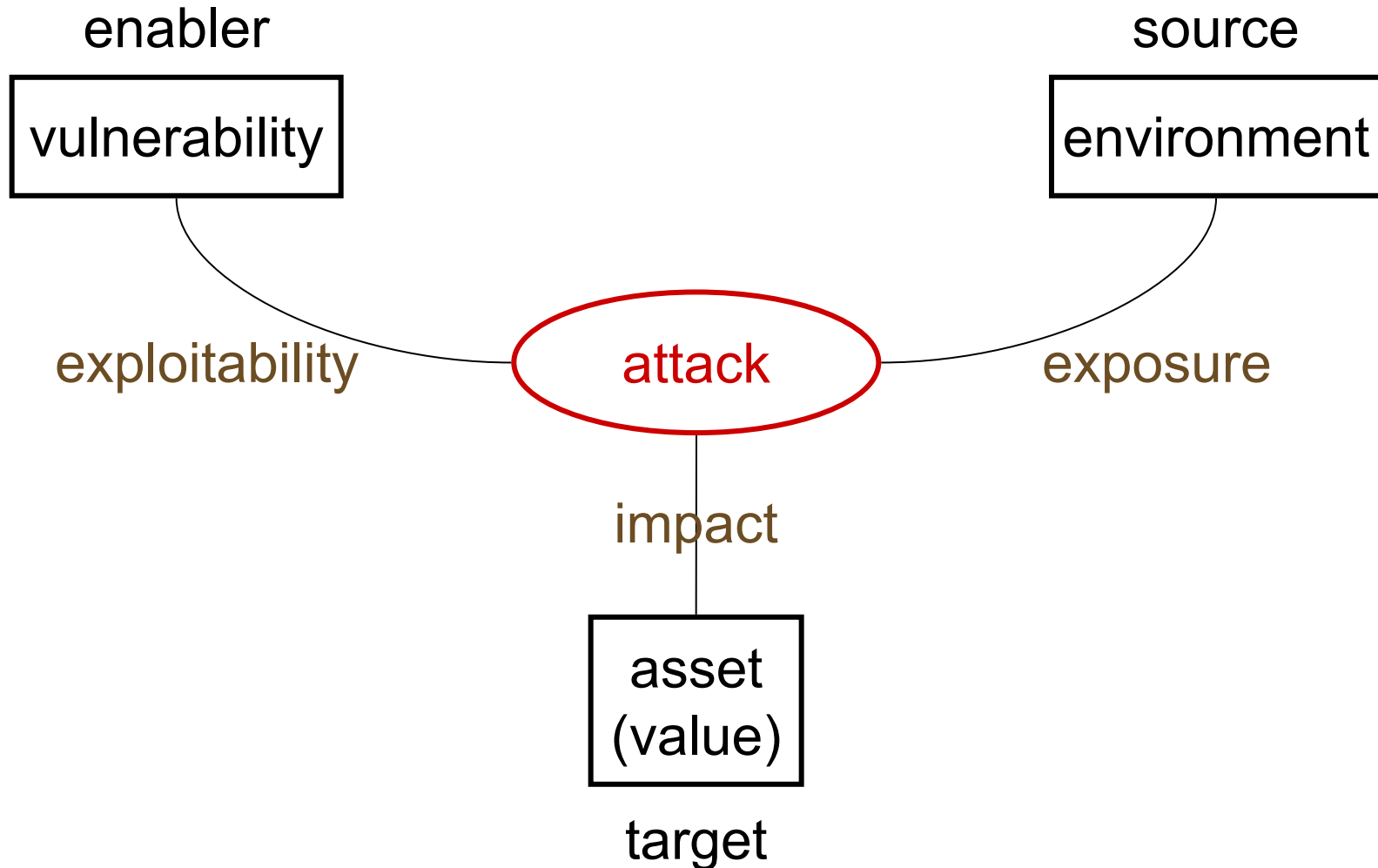
Risk Analysis

- To organize the process of risk analysis, we will look at **assets**, **vulnerabilities**, and **threats**.
- Risk is a function of **assets**, **vulnerabilities**, and **threats**:

$$\text{Risk} = \text{Assets} \times \text{Threats} \times \text{Vulnerabilities}$$

- During risk analysis values are assigned to assets, vulnerabilities, and threats.

Factors in Risk Analysis



Quantitative or Qualitative?

- **Quantitative risk analysis:** values are taken from a mathematical domain like a probability space.
 - For example, we could assign monetary values to assets and probabilities to threats and then calculate the expected loss.
- **Qualitative risk analysis:** values taken from domains that don't have an underlying mathematical structure.
 - Risk calculated based on rules that capture the consolidated advice of security experts.

Assets

- First, assets have to be identified and valued; this step should be relatively easy.
- In an IT system, assets include:
 - Hardware: laptops, servers, routers, PDAs, mobile phones, smart cards, ...
 - Software: applications, operating systems, database systems, source code, object code, ...
 - Data & information: essential data for running and planning your business, design plans, digital content, data about customers, ...
 - Services & revenue
 - Reputation of enterprise, trust, brand name
 - Employees' time

Valuation of Assets

- Assets such as hardware can be valued according to their monetary replacement costs.
- For other assets such as data & information this is more difficult.
 - Leaked business plans or customers' private data may cause (indirect) losses due to lost business opportunities.
 - For lost or stolen equipment you have to consider the value of the data stored on it, and the value of the services that were running on it.
- Value assets according to their **importance**.
- As a good metric for importance: how long your business could survive when a given asset has been damaged?

Vulnerabilities

- Weaknesses of a system that could be accidentally or intentionally exploited to damage assets.
 - Admin account with default passwords
 - Overprivileged programs, or programs with known flaws.
 - Weak access control settings on resources
 - Weak firewall configurations that allow access to vulnerable services.
- Sources for vulnerability updates:
 - CVE (Common Vulnerabilities and Exposure) -- cve.mitre.org
 - NVD (National Vulnerability Database) -- nvd.nist.gov
 - CERTs (Computer Emergency Response Teams), SANS, BugTraq, ...

Rating Vulnerabilities

- Rate vulnerabilities according to their impact (level of criticality)
 - A vulnerability that allows an attacker to take over a systems account is more critical than a vulnerability that gives access to an unprivileged user account.
- **Vulnerability scanners** provide a systematic and automated way of identifying vulnerabilities.
- Some vulnerability scanners also give a rating for the vulnerabilities they detect.

Common Vulnerability Scoring Scheme

Basic metrics		Temporal metrics	Environmental metrics	
Access vector	Confidentiality impact	exploitability	Collateral damage potential	Confidentiality requirement
Access complexity	Integrity impact	Remediation level	Target distribution	Integrity requirement
Authenti-cation	Availability impact	Report confidence		Availability requirement

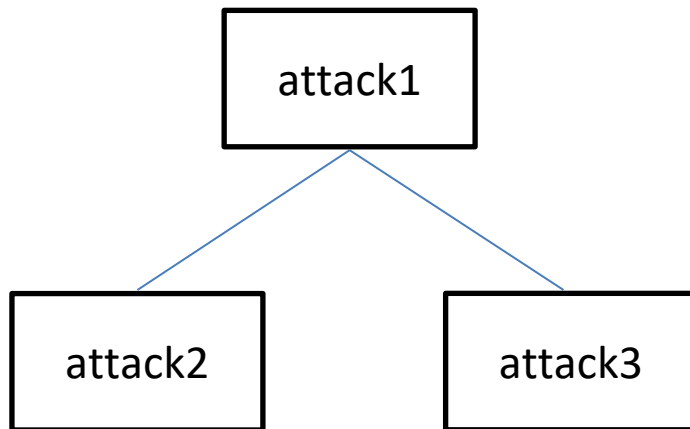
CVSS calculator: <https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator>

Threats

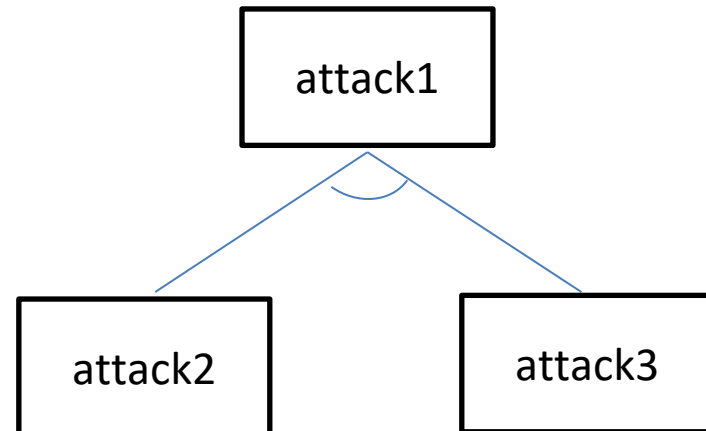
- **Threats:** actions by adversaries who try to exploit vulnerabilities to damage assets.
- Various ways for identifying threats:
 - Categorize threats by the damage done to assets.
 - Identify source of attacks. Insider or outsider, a contractor or a former member? Has the adversary direct access to your systems or is the attack launched remotely?

Attack Trees

- We can analyse how an attack is executed in detail.
- Steps of attacks are decided by experts.
- To get a fuller picture of potential threats, **attack trees** can be constructed.

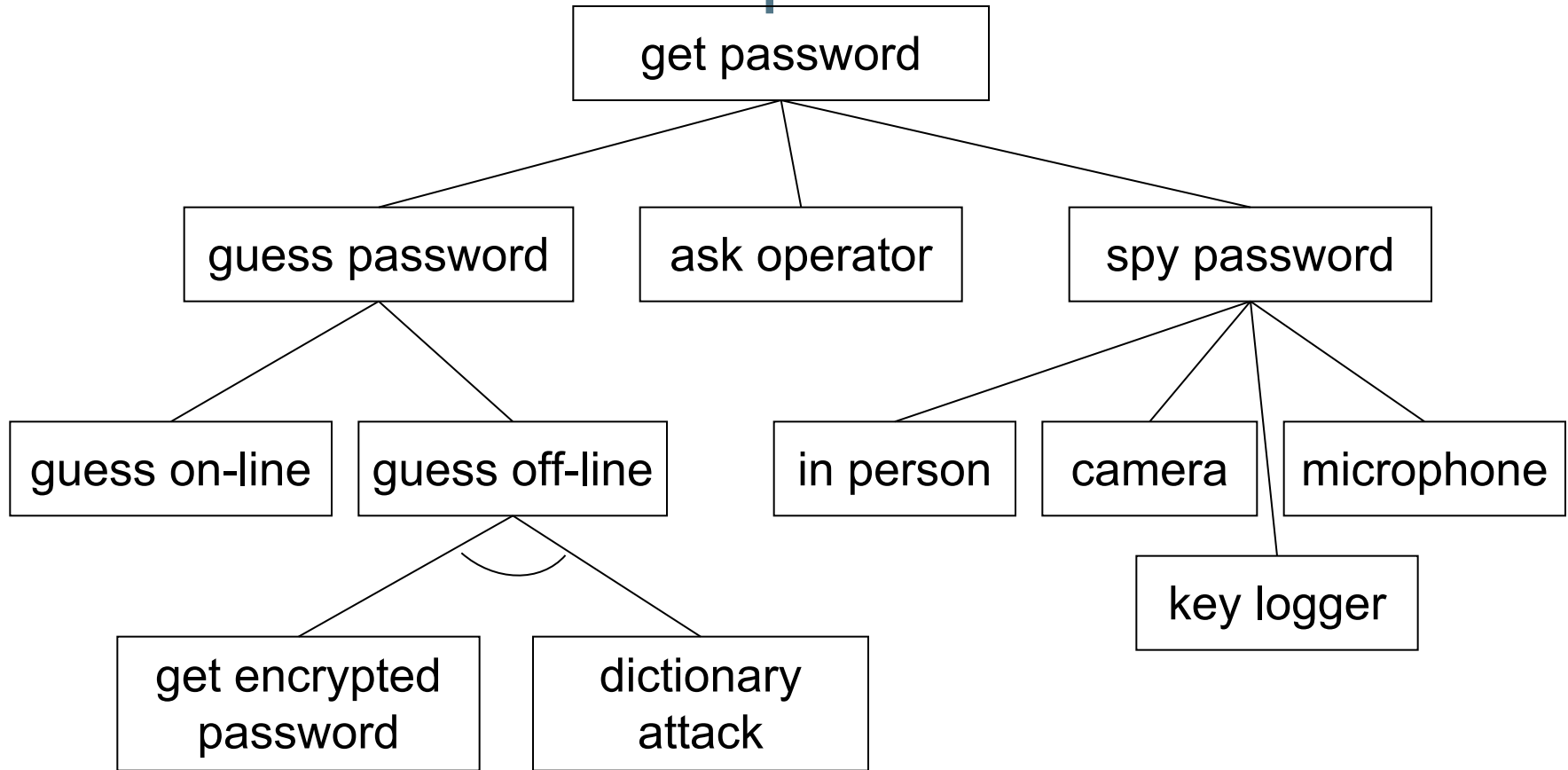


OR-Node: attack1 succeeds if attack2 or attack3 succeeds



AND-Node: attack1 succeeds if attack2 and attack3 succeeds

Attack Tree – example



See also: https://www.schneier.com/academic/archives/1999/12/attack_trees.html

Rating Threats

- Rate threats according to their likelihood.
- The likelihood of a threat depends on
 - difficulty of the attack,
 - motivation of the attacker,
 - number of potential attackers.
- **Attack scripts** automate attacks; they are likely to be available to a larger set of attackers.
- Hence, such attacks would be rated more likely than an individual hand-crafted attack.

Calculating Risk

- In **quantitative risk analysis**, expected losses can be computed based on monetary values for the assets and probabilities for the likelihood of threats.
- Advantage: uses a well established mathematical theory (probability theory)
- Drawback: the ratings obtained are often quite imprecise and based on educated guesses.
- The quality of the results we obtain cannot be better than the quality of the inputs provided.

Calculating Risk

- In **qualitative risk analysis**, rate
 - **assets** on a scale of *critical* – *very important* – *important* – *not important*.
 - **vulnerabilities** on a scale of *has to be fixed immediately* – *has to be fixed soon* – *should be fixed* – *fix if convenient*.
 - **threats** on a scale of *very likely* – *likely* – *unlikely* – *very unlikely*.
- For a finer granularity of scaling you could e.g. use numerical values from 1 to 10.
- Guidance must be given on how to assign ratings.
- The mapping of the ratings for assets, vulnerabilities, and threats to risks is often given by a table that reflects the judgement of security experts.

The MEHARI Approach

$i=4$	2	2	3	4
$i=3$	1	2	2	3
$i=2$	1	1	2	2
$i=1$	1	1	1	2
	$n=1$	$n=2$	$n=3$	$n=4$

Damage potential (d)
as function of **Impact (i)**
and
number of targets (n)

$d=4$	2	3	4	4
$d=3$	2	2	3	3
$d=2$	1	2	2	3
$d=1$	1	1	1	2
	$l=1$	$l=2$	$l=3$	$l=4$

Risk as function of **likelihood (l)**
and **damage potential (d)**

$e=4$	1	2	2	3
$e=3$	1	1	2	3
$e=2$	1	1	2	2
$e=1$	1	1	1	2
	$f=1$	$f=2$	$f=3$	$f=4$

Likelihood (l) as
function of
exploitability (e) and
feeling of impunity (f)

Risk Mitigation

- Risk analysis produces a prioritized list of threats, with recommended **countermeasures** to mitigate risk.
- Analysis tools usually have a knowledge base of countermeasures for the threats they can identify.
- General risk mitigation strategies:
 - **Accept** risk (and live with it); there may be good reasons to do so.
 - **Avoid** risk: eliminate a vulnerability that causes the risk; drop product feature that has a vulnerability.
 - **Limit** risk: use controls to make a threat less likely.
 - **Transfer** risk: buy insurance.

Baseline Protection

- Doing a risk analysis before deciding on which security measures to implement may not work:
 - Conducting a risk analysis for a large organisation takes time, but technology keeps changing; when the analysis is ready, it is already out of date.
 - Costs of a full risk analysis may be difficult to justify to management.
- Organisations may opt for **baseline protection**
 - analyses the security requirements for typical cases and recommends security measures deemed adequate.
 - [Australian Cyber Security Center 'Essential Eight'](#)
 - BSI Baseline Protection Manual

Summary

- Security management creates the context in which individual security mechanisms operate.
- Without good security management, even strong security mechanisms may be ineffective
- Important guidelines: ISO 27000 family of standards and baseline standards.
- Risk analysis gives management information about the risks an organisation faces and the countermeasures that can be taken.