Paper overview and an introduction to security

COSC312 / COSC412

COSC312 / COSC412 paper overview

- **Overall aim** of the paper
  - Explore the modern theoretical bases of cryptography—a central aspect of contemporary computing
  - Investigate security technology in practice

- Since 2014 focus on crypto. & security over complexity
  - (Obviously exam papers pre-2014 thus cover different topics)

- In 2023 we **introduced COSC312**—welcome!

Lecturer

- David Eyers
  - Primary expertise: cryptography in practice; security topics
- (In the past Prof. Michael Albert was co-teacher)
  - Expert in quantum cryptography, theory topics (+much more)
  - Is emeritus professor in SoC and frequently is in his office…

Teaching times: COSC312 / COSC412

- **Two-hour lecture** per week COSC312 & COSC412
  - COSC412 students otherwise carry out self-directed study
- Additional teaching schedule for COSC312:
  - On-demand **one-hour tutorial** per week
    - Tutorials start in week one, but no specific work is set
  - **Two-hour lab** slot per week
    - Labs start in week two
- No assessment linked to labs or tutorials

Assessment

- COSC312 **Two assignments** (40% total)
  - A1, worth 20%, due 25th August—start of week 7
  - A2, worth 20%, due 29th Sept.—start of week 11
- COSC412 **Three assignments** (40% total)
  - A1 and A2 as above, but both worth 10%
  - A3: **Poster and presentation** (20% total)
  - PDFs of posters will be due 10th October—end of week 12;
  - Presentations will be in week 13 (*i.e.*, the last week of term)
- **Exam**: Worth 60%; minimum 40% to pass paper; date TBC

Textbook? Resources?

- We are not setting a particular textbook for the course
  - We expect to provide **online references**
- The COSC412 and COSC312 website resources and lecture notes sections will link to relevant material:
  - https://cosc312.cspages.otago.ac.nz/
  - https://cosc412.cspages.otago.ac.nz/
- We'll present **more than the examinable material**
  - In exam: only what we've been able to discuss

More on posters and presentations (A3)

- COSC412: you will select a **security issue of interest** that you can research in groups

- Groups must **write & design their poster** collaboratively
  - They will be submitted before the presentations
  - Academic posters contain a lot of content—examples later

- **Presentations from groups** must involve all members of the group: during the introduction and/or poster tour

Potential outline of material

- Introduction, security, cryptography theory
  - **L1**: Introduction and administration
  - **L2**: Fundamentals of classical cryptosystems and one-time pads
  - **L3**: Stream ciphers, key agreement & asymmetric cryptography
- Quantum computation and cryptography
  - **L4**: Quantum computation
- More cryptography in practice
  - **L5**: Kerberos and Microsoft Active Directory
  - **L6**: Block ciphers, HTTPS, TLS/SSL and certificates
  - **L7**: Decentralised authorisation and OAuth 2.0

Potential outline of material (cont.)

- Mid-semester break is between L7 and L8
- More cryptography in practice
  - **L8**: Reliability, distributed consensus and bitcoin
  - **L9**: Blockchain and cryptocurrencies
  - **L10**: Homomorphic Encryption
  - **L11**: Programming language security
  - **L12**: Hardware support for software security & emerging tech.
- **L13**: Poster presentations **(412 students)** & exam advice

Learning objectives of lecture one

- Understand **computer security** fundamentals
- Be able to explain **cryptography's** role in security
  - For the 'in practice' parts of the course, we usually employ cryptography as a black box tool
- Appreciate **alternatives** to cryptography
  - Describe the limits of cryptography as a tool
  - Explain threats cryptography cannot protect against

What is cryptography?

- A dictionary definition:
  - cryptography |krip'tägrəfē| noun
  - "*the art of writing or solving codes.*"

- You should aim to be able to define the term more specifically to computing than this!
  - The theory part of this course will help…

What is computer security?

- **Physical security**: protect console / computer
  - Computer can be stolen? Encrypt disks
- **Software security**: authenticity, correctness
  - *e.g.*, code signing; verifying software behaviour
- **Information security** has three main pillars:
  - Confidentiality; Integrity; Availability
- **Network security**: untrustworthy regions

Why is cryptography useful for security?

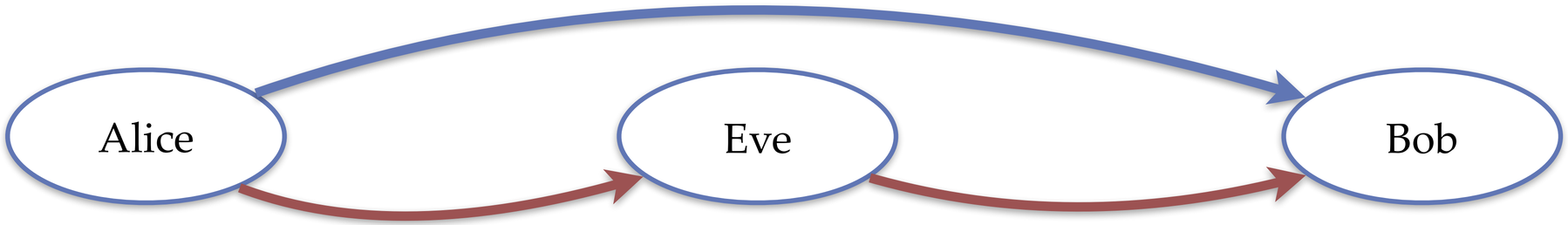
- **Correctness** property: An **untrusted channel** can be used by intercommunicating **trusted principals**

- What about supporting **liveness** properties?
  - *e.g.*, keeping secure communications flowing over a link?

- Attackers don't need full control to **break systems**
  - *e.g.*, DDoS (Distributed Denial of Service):
    - Attackers overwhelm target system without seeing any secrets

Key principle: shared secret

- **Trusted interactions need pre-shared data**
  - Diffie-Hellman key-exchange establishes a shared secret but doesn't authenticate—beware adversary in the middle (AitM)
- Look for where **shared secrets** fit in any given system
  - May not be immediately obvious
- Contrast the shared secret encoding in:
  - HTTPS, SSH, PGP



Some security doesn't need cryptography

- **Physical security**
  - Air gap isolation; walk-in access to data centres
  - Restricting peripheral access (how?)

- **Network security**
  - Separate physical network cabling
  - Separate virtual networks (*e.g.*, VLANs)

- What about **software security**?
  - Compile software from source… but is this enough?

When is cryptography use inappropriate?

- **Performance** used to be an argument—less so, now
- Storage of **life-long sensitive data**?
  - While attackers might not be able to read the data today, you *are* still giving them your data in some form!
  - For **how long** will a given cypher be secure?
  - What application domains have this concern?
- **Managing keys** may be challenging

Cryptography ageing (… badly)

- **Cryptographic strength** diminishes over time
  - *e.g.*, DES
- **Cryptography design** may have bugs
  - MD5—hash collisions can be constructed:
    - http://s3.amazonaws.com/dmk/md5_someday.pdf
- **Implementations of protocols** may contain bugs
  - OAuth; Kerberos 4; NTLM; …

New hardware, new threats to crypto.

- Hardware performance increases allow for **brute-force attacks** that were not previously possible
  - Attacks parallelise easily: e.g., using multicore CPUs, GPUs
    - … tensor processing units (TPUs); programmable hardware (FPGAs)
    - … and many available via large botnets

- **Indexing techniques**: attackers have more storage too
  - Can compute large datasets for attacks (*e.g.*, rainbow tables)

Pillars of information security

- Recall the three main pillars of information security:
  - **Confidentiality**, **Integrity**, **Availability**—CIA (!)
  - We will look at where cryptography fits within each

- Other classifications exist, such as the **IAS Octave**:
  - Adds: privacy, authenticity & trustworthiness, non-repudiation, accountability and auditability
  - CIA principles can help inform these extra ones

Crypto in info. sec.: confidentiality

- **Confidentiality** (AKA secrecy) is probably the most widely appreciated cryptography use
  - Hiding of information
  - Controlling a set of people that have access

- Cryptography supports confidentiality when key distribution is controlled
  - **Asymmetric cryptography**: easier key distribution control
  - (Alternatively just don't give out the data!)

Crypto in info. sec.: integrity

- **Checksums** can check for data being modified
- Go further to create **Message Authentication Codes** (MACs) that include principal's identifying information
  - Uses symmetric cryptography→shared key→no nonrepudiation
- **Digital signatures** go further than MACs
  - Use asymmetric cryptography
  - Include necessary means for nonrepudiation

Crypto in info. sec.: availability

- Can cryptography help **secure availability**?
  - Not directly…
- Resources are used when rebuffing attacks
  - Therefore attacks can affect availability cheaply
- Cryptography **can help indirectly**
  - Validate authenticity of network link usage
  - Effect distributed rate control of malicious use

Cryptography in code executables

- Signing of ‘data’ that is actually executable code
  - *e.g.*, Java Archives (JARs), and
  - macOS and Windows executables
- Linux package repositories include signatures
  - Often of packages rather than the EXEs contained (Debian)
- … also sometimes from the bad guys (how?)

Building effective, secure systems

- Ross Anderson (University of Cambridge) pioneered the field of **Security Engineering**—whole system view
  - Cryptography? Yes, but also:
  - Social science; psychology; economics; *etc*.
- Need to apply **threat modelling**:
  - Identify assets, adversaries, and capabilities
- In any case: best **plan for security failures!**

Social engineering attacks

- **Users** are often the weakest link in secure systems!

- Why would hackers try to break cryptography when they can **access services through users**?
  - **Phishing** attacks are highly profitable
    - We wouldn't fall for '**driftnet**' attacks
    - … but targeted social engineering attacks are a different story (**spear-phishing**): careful research is undertaken by the attacker

Authentication and Authorisation

- Return to how users participate in security
- **Authentication** involves proving identity
  - Generally this should not need to change much
- **Authorisation** checks follow authentication
  - Privileges of user on target system are checked
  - Much more likely to change frequently

… AAA—add Accounting too

- Systems such as RADIUS provide for AAA
  - (Remote Authentication Dial In User Service)
  - RADIUS is often behind corporate Wi-Fi APs

- In addition to managing user identity, and user privileges, RADIUS also manages usage tracking

- How does cryptography link to accounting?

Revocation

- Justifies authorisation / authentication split:
  - May need to **remove the privileges** of a user,
  - but you can't 'remove' **their identity**

- **How quickly** does revocation mechanism take effect?

- Revocation and digitally-signed assertions:
  - Can systems revoke digitally signed statements?
  - *e.g.*, HTTPS CRLs—more on these later

Delegation

- Delegation is a desirable security facility
  - **Temporarily** give another user privileges
  - Needs a clear **revocation protocol**
  - … or an understanding that revocation is impractical
- Most use-cases only **transfer some privileges**
  - Aim not for delegator to be entirely impersonated by target of delegation! e.g., a **helper app** doesn't have all your privileges
  - … so we need rich user privilege representation, which leads onto **access control** policy

Access Control

- … is an enforcement mechanism of some policy
  - **Policy** describes what is allowed
  - **Mechanism** is how that policy is enforced

- Typically **code-based enforcement**, but this risks:
  - Missing access control checks
  - Time of check to time of use (TOCTOU) errors

- Can code access control directly into software, but…
  - Ideally make **policy entirely code independent**
  - Can use libraries such as XACML

Cryptography for access control

- Cryptography supports aspects of access control:
  - **Authentication** of users
    - Password hashes
    - Challenge-response interactions
  - **Controlling access** to data
    - Digital Rights Management
    - Encrypted filesystems
  - **Digital signatures on audit logs**
    - Ensures logs cannot be tampered with in an undetectable manner

Access Control Matrix

- Fundamental representation of **users**, **objects** and **privileges** within a secured system

	/dev/random	Directory 'logs'	File 'report.pdf'
User Jim	read	read, write, execute, own	
User Ned	read	read, execute	read,write,own

- Collect columns? Get Access Control Lists (ACLs)
- Collect rows? Get 'capabilities'
- … but this representation is of static security

Discretionary Access Control—DAC

- DAC is the most common form of access control
- Users are **free to modify access privileges** over objects that they own—think Unix / NTFS filesystem permissions
- No system-wide security policy

Mandatory Access Control—MAC

- Common in military / intelligence services
- Data-linked security: **system-wide policy**
  - Often based on labels
  - Users have labels; processes inherit labels
  - Data items also have labels
- User/data **label policy is enforced**, *e.g.*:
  - No write-down—you can't declassify information
  - No read-up—you can't read more sensitive data

Role-based Access Control—RBAC

- Insert roles as an **abstraction** between users & privileges

- Like user groups, but more expressive
  - Roles have to be activated **within a session**
  - **Role activation** usually under control of the user
  - *e.g.*, RBAC avoids Solaris needing all-powerful 'root' user

- We'll see an RBAC / crypto link much later

Summary

- Introduced **cryptography** and **security**
  - Cryptography is not always needed for security
  - Placed crypto in the context of access control
  - Skimmed over use of crypto in typical software systems and network protocols

- **Security Engineering**: a whole-system view
  - Consider all of the interacting participants
  - **Plan for security failures**—everyone makes mistakes!