Trusted hardware and emerging technology

COSC349—Cloud Computing Architecture

David Eyers

Learning objectives

- Appreciate that multiple approaches are emerging that provide **hardware-based security** for the cloud
- Sketch how information flow control and provenance tracking can help manage **data sovereignty needs**
- Understand that **edge computing** and **IoT integrations** are growing rapidly as cloud connected applications
  - Serverless computing is a unifying trend; 5G is also in the mix…

Assisting cloud security using hardware

- **Cloud security** is a significant source of client concern
  - As noted previously, **cloud may be safer** than local security…
  - Additional assurances can come from hardware and software
- We'll skim over three promising **hardware approaches**:
  - Virtual Trusted Platform Modules; Intel SGX; capability machines
- Many computers have a Trusted Platform Module (TPM)
  - ISO/IEC 11889—released 2009
  - Typically implemented as a separate chip or chipset firmware

Virtual TPMs

- TPMs can facilitate **attesting software** for provider
  - … but now virtual TPMs are implemented, too
    - Can be used by tenants to cryptographically check their code
  - IBM pioneered vTPM work—needed to consider tradeoffs:
    - Too much leverage of the real TPM, and lose VM migration
    - Too little use of the real TPM and the vTPM loses strength
- Some security concerns surround (v)TPMs though:
  - Concern that manufacturer has **undue power over machines**
  - Numerous **TPM implementation flaws** have needed repair

Enforcement of security using Intel SGX

- Intel Secure Guard Extensions (SGX)
  - Noted in security lecture: secure software runs in ‘enclaves’
  - All **code and data encryption/decryption** done by CPU
  - Can run signed code on an **untrusted kernel**
- Pragmatic balance between TPM and nothing
  - … however **SGX suffers Spectre** and motherboard problems
- Involved in research ‘porting’ Docker to use SGX
  - Performance boosted by minimising enclave entry/exit

Full VM encryption

- Entire VMs are encrypted, including their OSs
  - Don't get protection between components building the VM
  - … but simpler not needing to port code to use SGX enclaves

- AMD Secure Encrypted Virtualisation (SEV)
  - Protects VMs from hypervisor and other VMs

- Arm TrustZone
  - Splits CPU operation between secure and normal 'world'
  - Can isolate some CPU operations from main OS

Hardware supporting memory capabilities

- Emerging cloud-relevant technology… from the 1970s!
- Privilege separation within x86 CPUs is into four rings
    - As seen earlier, typically use VMM; VM OS kernel; userspace
- Capability machines: **fine-grained privilege separation**
    - Individual processes and threads can be isolated
    - Pointers are replaced with **capabilities**—checked before use
- Prototype **Arm CPUs** adopt a capability architecture

My Information Flow Control research

- Information Flow Control is **mandatory access control**
  - Principals have compulsory policy applied to them
  - In contrast, discretionary access control (DAC) allows resource owners to specify who can access their data

- **IFC uses security labels**: classified, secret, top secret, ...
  - All data is labelled
  - All principals operate at a labelled level
  - Simple limiting rules applied consistently: *e.g.*, "no write down"

DIFC and DEFC

- Decentralised IFC: security **label set can change, live**
  - Principals can create new labels, and issue privileges for labels
  - … has been applied in programming languages & OSs
    - *e.g.*, Asbestos (UCLA), Flume (MIT), JIF (Cornell), D-star (Stanford)
- Developed **Decentralised Event Flow Control** (DEFC)
  - We can treat all messages as multi-part structures
    - Apply IFC labelling independently to each part
    - Each part has its own data and security label
    - For transport, treat event as an atomic unit

Provenance of data in cloud computing

- (D)IFC significantly overlaps **provenance tracking**
  - Provenance describes the origin and dependencies of data
  - Common to reconstruct provenance for *post hoc* analyses

- Applying CamFlow engine to provenance tracking
  - CamFlow is a Linux-based system across kernel and user mode
  - CamFlow designed to provide **near-real-time provenance**
  - **Application-level semantics** can guide provenance filtering
  - Keen to move into provenance tracking in **distributed systems**

Data sovereignty management

- I believe provenance tracking through cloud is crucial
  - GDPR and other **protections of citizens** requires provenance

- Researching SDN routing, provenance and IFC links
  - OpenFlow is open source and runs REANNZ' Science DMZ
  - Plan: **apply DIFC to OpenFlow** control decisions

- REANNZ interest: to use labels to contain types of data
  - Provides a mechanism to **support data sovereignty needs**

Edge integration into serverless computing

- More computing types to cloud: will **get latency issues**
  - Data centres need to be large-scale to be cost effective
  - Cannot site data centres everywhere they need to be
- **Edge computing** is emerging as an intermediary
  - Saw that Amazon **Lambda runs in AWS edge nodes**
  - Feel open Function as a Service key to distributed computing
- Likely increases of **in-network programming** (*e.g.*, 5G)

Further off—reliable fog

- **Fog computing** aims for cloud to spread everywhere
  - Currently IoT would be the endpoint of much fog computing

- IoT has too hard a time getting security right, presently
  - How do you **securely deploy and configure devices**?
  - How do you do a **software update safely** on all devices?

- One possibility: **IoT and commodity OSs converge**
  - Would require lower-power use than current commodity OSs

Amazon Cloud9 and other cloud IDEs

- **Web-based IDE for AWS services** (IDE-as-a-Service?)
  - IDE is open source; runs on EC2 or your own Linux server
    - (but needs connectivity back to AWS, so SSH from your own server)
  - Provides real-time collaboration within editor
  - Integrated debugger; source code revisions
- AWS integration convenience:
  - Command line with **pre-authenticated aws** tool use
  - Serverless software development: preloaded SDKs and libraries
  - AWS **continuous integration and deployment**

Serverless Application Model (SAM)

- AWS CloudFormation mentioned previously
  - **Orchestrates AWS IaC** (YAML/JSON)
  - Cross-account; cross-region; dependencies managed
- **SAM extends CloudFormation** for serverless apps.
  - Integrates with Cloud9 (IDE) and AWS deployment tools
- SAM gives YAML **syntax for key serverless components**:
  - functions; databases; event source mappings; APIs
  - Language is open source (and available on GitHub)