Contents

Contributors:
- Daniel Fitz (Sanchez)
- Jake Dunn (nomad)
- *Notes from UQAttic.net*

1 Exam Structure

Exam Is Open Book!

1.1 What is Covered?

- All lecture content (Weeks 1, 2, 3, 5), (Required text book reading)
- All Seminars
  - For seminars that are based on a research paper (#20 - #50), both content on slides and paper is relevant
  - For all other seminars, only information presented in seminar is covered by exam
  - Emphasis in exam will be on seminars based on research papers
- Guest Lecture by Peter Robinson (W4) is not covered

1.2 Format

Two parts with total $\approx$ 55 marks

- Part A: 8 Questions ($\approx$25 marks)
  - Questions on Lectures
  - Material covered in lectures in weeks 1,2,3 and 5
  - Short Answer/Problem
- Part B: Answer 3 questions (30 marks)
  - Questions on Seminar Presentations
  - Mix of essay-style and short answer questions
  - Select and answer 3 out of 4 questions
  - Cannot do own seminar question, get an extra question to choose from

1.2.1 Part B Example Questions

- Describe what the XREP protocol presented in the paper tries to achieve, and discuss the basic mechanisms that it is using
- Further discuss for what environments it can be applied and describe its limitations and vulnerabilities
- Describe the relevance of the parameter K in the proposed protocol
- Describe at a high level what Aurasium is, and the key goals it is trying to achieve
- Describe how Aurasium interacts with the Andriod system and applications
- Describe if and how malicious application can detect the presence of Aurasium

2 Seminar Paper Summaries

#20 On Scaling Decentralized Blockchains

- Tackles the question of whether blockchains can be scaled up to match the performance of a mainstream payment processor, and what it takes to get there
- Finds that fundamental protocol redesign is needed for blockchains to scale significantly while retaining their decentralization
- Identifies a three-way trade-off among consensus speed, bandwidth, and security. You can do two well, but usually one is traded off
- Separates the different areas of a Blockchain system into 5 distinct planes: Network, Consensus, Storage, View and Side (Planes)
- **Network Plane:** role of propagating transaction messages, specifically valid transactions. Two major inefficiencies:
  - To avoid DoS attacks, where an invalid transaction is attempted to be propagated, a node must fully receive and attempt to validate the transaction before ignoring it if it's invalid
  - Transactions are propagated, and then later, a block is propagated when it is mined (which contains all the previously propagated transactions). Each transaction is transmitted twice
- **Consensus Plane:** the role of mining blocks and verifying their legitimacy and addition to the Blockchain
  - Held back by proof-of-work, which facilitates the 'three-way trade-off' identified above. Changing this has the potential to overcome this problem
- **Storage Plane:** essentially a 'global memory' that stores and provides availability for authenticated data produced by the Consensus Plane
  - Essentially the distributed ledger
  - Weakness with Bitcoin's distributed ledger is that each node stores the entire ledger, resulting in many duplicates
- **View Plane:** facilitates the function of a view over the UTXO (unspent transaction outputs) set. It has a lot of similarities to the Storage Plane
- **Side Plane:** allows off-the-main-chain consensus

#30 Cold Boot Attacks on Encryption Keys

Halderman et al., Lest We Remember: Cold Boot Attacks on Encryption Keys, USENIX Security Symposium, 2008

2.0.1 Abstract

"Contrary to popular assumption, DRAMs used in most modern computers retain their contents for several seconds after power is lost, even at room temperature and even if removed from a motherboard. Although DRAMs become less reliable when they are not refreshed, they are not immediately erased, and their contents persists sufficiently for malicious (or forensic) acquisition of usable full-system memory images.

We show that this phenomenon limits the ability of an operating system to protect cryptographic key material from an attacker with physical access. We use cold reboots to mount successful attacks on popular disk encryption systems using no special devices or materials. We experimentally characterize the extent and predictability of memory remanence and report that remanence times can be increased dramatically with simple cooling techniques.

We offer new algorithms for finding cryptographic keys in memory images and for correcting errors caused by bit decay. Though we discuss several strategies for partially mitigating these risks, we know of no simple remedy that would eliminate them."

2.0.2 Notes

- DRAMs typically lose their contents over a period of several seconds, if the chips are cooled to low temperatures (-50°C), the data can persist for minutes - hours.
- The researchers used non-destructive disk forensics techniques to create memory images, and extract the keys needed to decrypt several popular disk encryption systems (BitLocker, TrueCrypt and FileVault)
- If a system is rebooted, often the BIOS will overwrite portions of memory, and some systems are configured to perform a destructive memory test during its Power-On Self Test (POST)
- A warm boot is initiated by the host operating system and gives the OS a chance to cleanly exit application and wipe memory. A cold boot is initiated either by pressing the system restart button, or temporarily removing the power, this gives the OS not opportunity to scrub memory state.
- In order to create images of the memory, the DRAM in a running system is first cooled, then a cold boot is initiated and the system is booted into a special forensics tool via PXE network boot/USB etc. The cooled DRAM could also be removed an installed into another machine, bypassing any BIOS/POST protections.
- Algorithms were developed to extract cryptographic keys and correct bit errors in the range of 5% - 50%, this is achievable by comparing the key to other key precompute schedules stored in memory. ie. RSA's p + q values are often stored alongside the private key in order to perform faster computations.
- The paper describes in detail the process for reconstructing DES, AES, RSA and tweak keys.
- A method was developed in order to identify keys in memory, even in the presence of bit errors. This is done by again looking for additional key schedules, searching for blocks of memory that closely satisfy the combinatorial properties of a valid key schedule
- The paper also describes in detail the process for identifying AES, DES, RSA and file system encryption keys in a memory dump.
- Countermeasures discussed include:

  Scrubbing Memory: software should overwrite keys when they are no longer needed, and prevent keys from being paged to disk

  Limiting booting from external media: the majority of attacks were only possibly by booting into the forensics tools

  Suspending a system safely: require a password in order to wake a suspended computer, memory should be encrypted during suspension with a key derived from the password.

  Avoid precomputation: precomputations should be cached for a certain period and scrubbed if not re used.

  Key Expansion: Apply some transform to keys when they are stored in memory in order to make it more difficult to reconstruct

  Physical Defence: Lock/Encase the DRAM to prevent removal

  Architectural changes: Design DRAM that loses its state very rapidly past the intended refresh interval

  Encrypting in the disk controller: Perform disk encryption operations in disk controller rather than by software in the main CPU, and storing the keys in the disk controller rather than DRAM.

  Trusted computing: Deploying trusted computing hardware will help the machine determine whether it is safe to store keys in DRAM at this time or not.