Trust II

Signed Apps

Signed Apps

- Download app from iOS, Android app stores
- App is cryptographically signed by developer
- Developer's public key signed by Apple, Google

Signed Apps

- Issues
  - Signature only says “developer certifies this app”
  - Doesn’t say *why* they certify the app
  - Doesn’t say how they made the app
  - Doesn’t say what other parties were involved

Signed Apps

- What could go wrong?

Signed Apps

- What could go wrong?
  - The developer could be malicious
  - The developer could be compromised
  - A dependency's developer could be malicious
  - A dependency's developer could be compromised
  - ...etc

Signed Code

- Developer releases, signs *source code*
- We compile it ourselves
  - Don’t have to trust their compiler
  - Don’t have to trust their filesystem
  - Don’t have to trust their network
  - etc
- What could go wrong?

Signed Code

- What could go wrong?
  - *Reflections on Trusting Trust*
  - How do you trust your compiler?

Reproducible Builds

- One technique is called *reproducible builds*
- Need deterministic compiler
  - Same source code? Same binary.
- Developer signs
  - Source code
  - Hash of binary
- Builds are reproducible; hashes should match

Reproducible Builds

- Much safer! Why?
  - Have to compromise developer's *and* customer's compilers
  - Debian reproducible builds
    - "It should be possible to reproduce, byte for byte, every build of a package in Debian."
    - Still distribute binaries, but source code is an auditing mechanism

Reproducible Builds

- What could go wrong?

Reproducible Builds

- What could go wrong?
  - What if your hashing program is compromised?
  - What if your kernel is compromised!?

Rootkits

- Malware that modifies key pieces of a system
  - Important programs, kernel, etc
- Lie to the rest of the system about anything
- e.g., Stuxnet
  - Mess with centrifuges
  - Lie to data feeds so everything looks fine
- e.g., Flame
  - Bypass filesystem drivers, write data directly to disk

Rootkits

- But we can just wipe the drive and reinstall
- Let’s go deeper...

Rootkits

- February 2015, report on the *Equation Group*
  - (They love cryptography)
- Wrote a virus which infects hard drive firmware
- Good luck getting rid of that...

Drive Encryption

- Possible Solution: Drive Encryption
- Encrypt your data so that it's opaque, unforgeable to attackers
- At boot, enter password to unlock
- Kernel transparently decrypts reads, encrypts writes
- What could go wrong?

Drive Encryption

- What could go wrong?
  - Your decryption program has to be unencrypted
  - What if it’s compromised?

Verified Boot

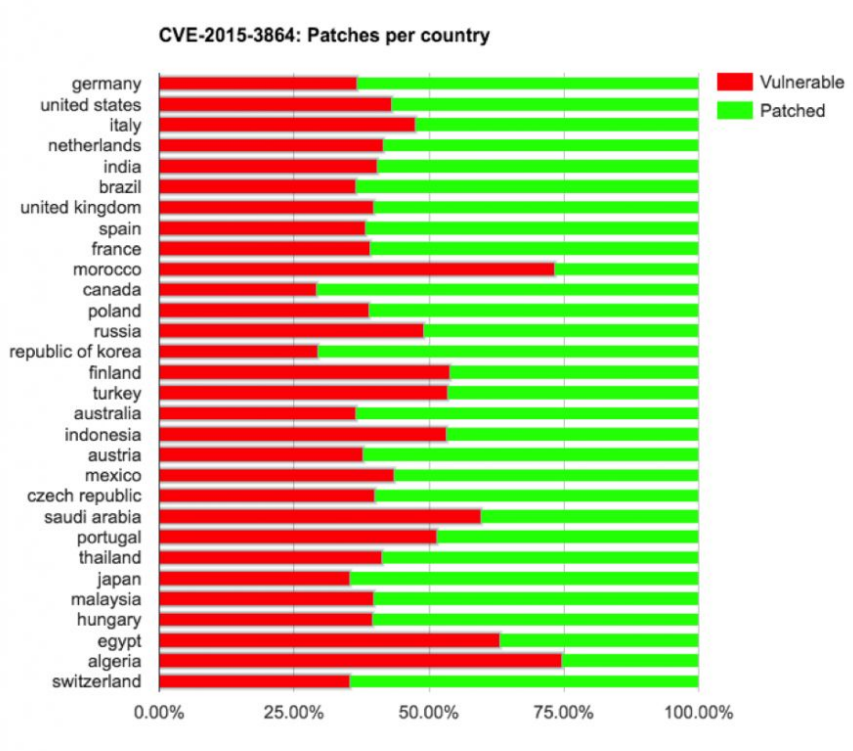
- Possible Solution: Verified Boot
- You need:
  - A tamper-proof chip that can perform crypto
  - A cryptographically signed hash of a “known good” kernel by a known public key
- At boot, verify the authenticity of the kernel

Auto-Update

Stagefright

- July, 2015 - RCE vulnerability found in Android's `libstagefright`
- Exploited by sending MMS to victim
- CVE reported September 30, 2015

Stagefright: Patches as of Mar 2016



https://blog.zimperium.com/wp-content/uploads/2016/03/3864-vulnerable-by-country-

Stagefright

- Once vulnerability is public, patch time is critical
- Stagefright patch time was mostly OEMs
- In general, though, users don’t install updates quickly

Auto-update

- Solution: auto-update
- Apps/systems update themselves without asking user
- 2008 study of web browsers
  - "...at least 45.2%, or 637 million users, were not using the most secure Web browser version on *any* working day from January 2007 to June 2008." (emphasis added)

Auto-update

- Generally considered much more secure...
- Upside: app creator can push updates
  - Fix bugs, security vulnerabilities
- Downside: app creator can push updates
  - Apple vs. FBI

BYOD vs. COPE

BYOD vs. COPE

- BYOD - *Bring Your Own Device*
  - Employees use personal devices
- COPE - *Corporate Owned, Personally Enabled*
  - Company manages devices, employees use for personal
- Pros? Cons?

BYOD vs. COPE

- Pros? Cons?
  - BYOD
    - Happier employees
    - Can’t control security as easily
    - Usually install administrator app, require antivirus, etc
  - COPE
    - Slightly less happy employees
    - Can make stronger guarantees about security of the device
- An example of controlling process (COPE) vs. verifying outcome (BYOD)