What does Remote Attestation buy you?

Matthew Garrett
<mjg59@google.com>

What is attestation?

at·test

/əˈtest/

verb

provide or serve as clear evidence of.

"his status is attested by his recent promotion"

- ○ declare that something exists or is the case.

  "I can **attest to** his tremendous energy"
- ○ be a witness to; certify formally.

  "the witnesses must attest and sign the will in the testator's presence"

Why is this hard?

- If a computer is running trustworthy code, you can trust what it tells you
- If it’s not, you can’t
- How do you tell the difference?

How do we solve this?

- We need a trustworthy third party agent
- On most platforms, this is a Trusted Platform Module

What’s a TPM?

- Small low power device
- Not computationally proficient
- Implements some cryptographic primitives
- Small amount of local storage
- No ability to introspect system state

How does the TPM know what’s happening?

- The OS tells it
- (uhm)

No, really

- We can’t trust the OS
- So we need to root our trust in an earlier component
- So the bootloader tells the TPM about the OS
- And the firmware tells the TPM about the bootloader
- And the firmware bootblock tells the TPM about the firmware
- (And with Bootguard, the Management Engine tells the TPM about that)

What information is given to the TPM?

- A series of cryptographic hashes
- Hashes are stored in Platform Configuration Registers
- But not stored *directly*
- New hash is concatenated to the existing value, and the hash of that stored
- Different PCRs are used to record different boot components

How does the TPM attest?

- The TPM can generate a signature over the PCR values and a nonce
- Examine the signed PCR values, see if they match the expected values
- If so, the system booted in the expected state

…but

- If the OS isn’t trustworthy, we can’t trust the OS to validate the PCRs
- So we need a trusted third party
- (again)

Welcome to remote attestation

- Provide the signed values to a remote machine
- Remote machine can then make a decision

But how do we know we’re talking to a real TPM?

- Every TPM has a unique Endorsement Key
- Endorsement Key has a certificate that chains back to the TPM vendor
- Generate trust by ensuring the TPM has a valid Endorsement Certificate
- Ensure that the Attestation Key corresponds to the Endorsement Key

How do we interpret the values?

- Raw hash values aren’t much use
- Performing the same events in a different order will result in different hashes
- Knowing how we got to the final values is important

Enter the event log

- Each event logged in the TPM is placed in the event log
- The event log is entirely managed by the firmware and OS
- The TPM has no visibility into the event log

How do we trust the event log?

- Replay the events in the order they occur in the event log
- Verify whether the calculated values match the TPM values
- Verify whether individual entries match the hashes (depends on event type)

What’s in the event log?

- Largely spec-defined information
- Hashes of EFI drivers and applications
- Information about the secure boot configuration
- Potentially more!

What else is in the event log?

- Firmware information
- Which is generally very vendor dependent
- And largely undescribed

Things are improving!

- Some vendors provide final PCR0 values in, uh, READMEs
- LVFS has a field to provide information

Not really quite what we want

- PCR0 contains multiple events, because reasons
- Knowing valid values for each of those events is helpful
- (As is knowing how to parse information from them)

We can validate device state. What else?

- We can validate device *identity*
- Platform certificates allow device/TPM binding
- Currently poor ecosystem support

What can we do with strong device identity?

- How do you gain trust in a new server?
- Wouldn’t it be nice if this were easier?
- Netboot, attest, obtain secrets

What else can we do with strong device identity?

- Provide machines with independent proof of their identity
- Allow machines to mint their own identity keys
- Solve SSH TOFU in corporate environments

Where are we on Linux?

- Shim will measure the used certificates into PCR7
- Grub will measure the kernel, initramfs, command line and more
- Kernel exports eventlog via `/sys/kernel/security/tpm0/binary_runtime_measurements`
- IMA takes responsibility at runtime

What do we still need?

- Better PCR0 values from vendors
- Better PCR values from distributions

What can we do with this?

- Remote attestation doesn’t need to be very remote
- All we need is a communications channel to a trusted device
- And some way to indicate success or failure

Possible channels

- USB?
- Bluetooth?

How do we do this?

- HIRS (github.com/nsacyber/HIRS)
- Keylime (keylime.dev)
- Go-attestation (github.com/google/go-attestation)

TODO

- Ubiquitous platform certificates from vendors
- Published PCR0 values from system vendors
- Published driver hash values from device vendors
- Published boot component hashes from OS vendors
- Agreement on what material should be measured and how
- Meaningful firmware validation infrastructure