

Bootstrapping Trust in Commodity Computers

Bryan Parno, Jonathan McCune, Adrian Perrig
Carnegie Mellon University

A Travel Story

Trust is Critical

Will I regret having done this?

Does program P compute F?
Is F what the programmer intended?

Bootstrapping Trust
What F will this machine compute?

Bootstrapping Trust is *Hard!*

Challenges:

- Hardware assurance
- Ephemeral software
- User Interaction

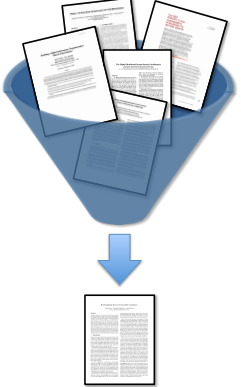
Bootstrapping Trust is *Hard!*

Challenges:

- Hardware assurance
- Ephemeral software
- User Interaction

In the paper...

- Bootstrapping foundations
- Transmitting bootstrap data
- Interpretation
- Validation
- Applications
- Human factors
- Limitations
- Future directions
- ... and much more!

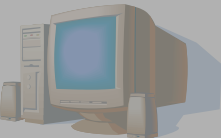


7

1) Establish Trust in Hardware

- Hardware is *durable*


Open Question: Can we do better?



8

2) Establish Trust in Software

- Software is *ephemeral*
- We care about the software *currently* in control
- Many properties matter:



Which property matters most?


...

9

A Simple Thought Experiment

- Imagine a perfect algorithm for analyzing control flow
 - Guarantees a program always follows intended control flow
- Does this suffice to bootstrap trust? *No!*

We want code *identity*



10

What is Code Identity?

- An attempt to capture the behavior of a program
- Current state of the art is the collection of:
 - Program binary
 - Program libraries
 - Program configuration files
 - Initial inputs

Function f

Inputs to f

- Often condensed into a hash of the above

11

Code Identity as Trust Foundation

- From code identity, you may be able to infer:
 - Proper control flow
 - Type safety
 - Correct information flow
 - ...
- Reverse is not true!

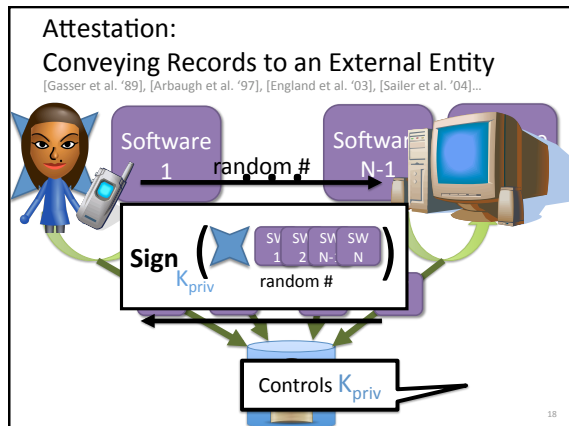
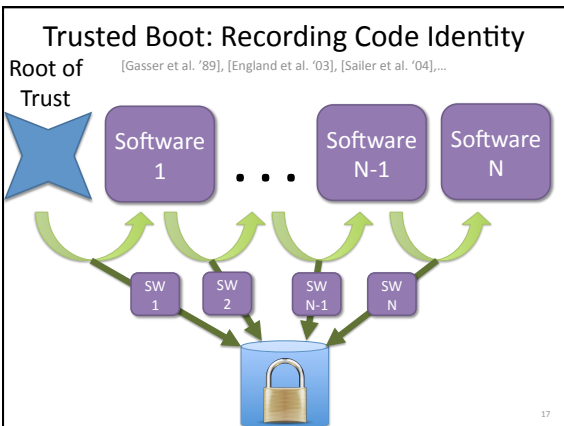
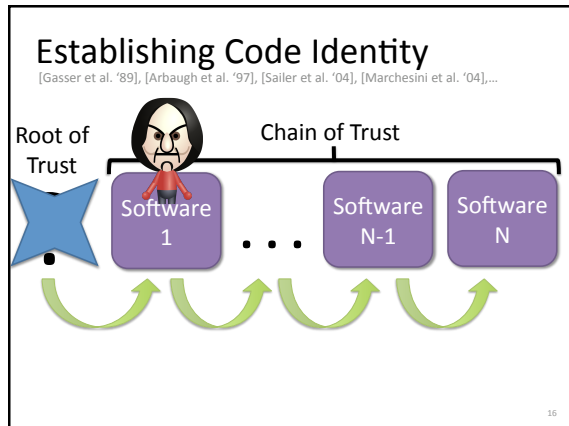
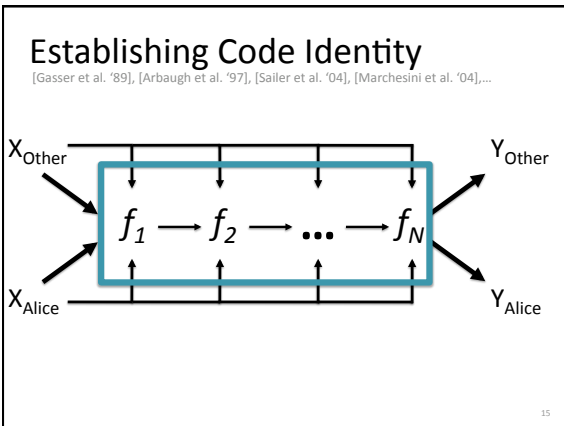
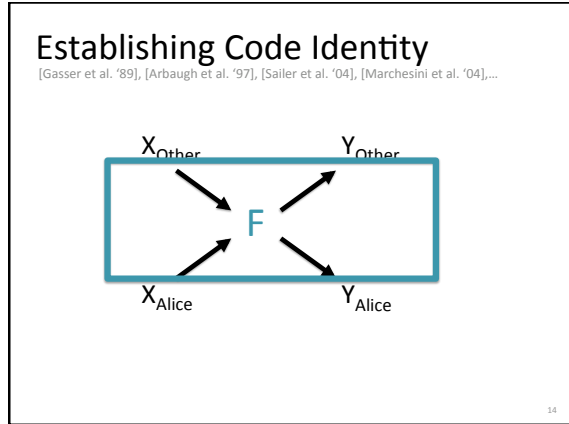
12

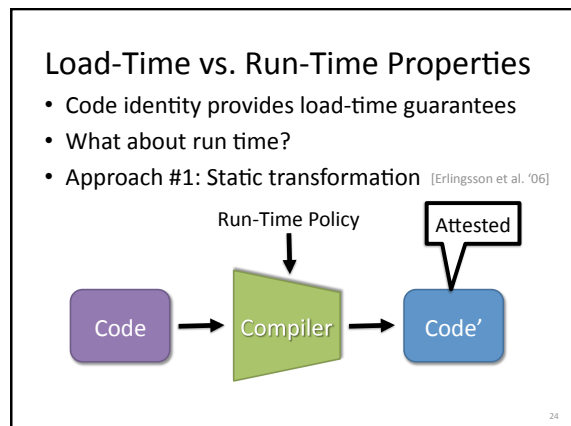
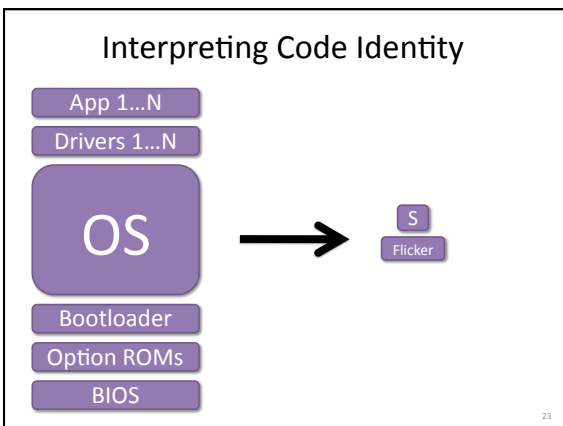
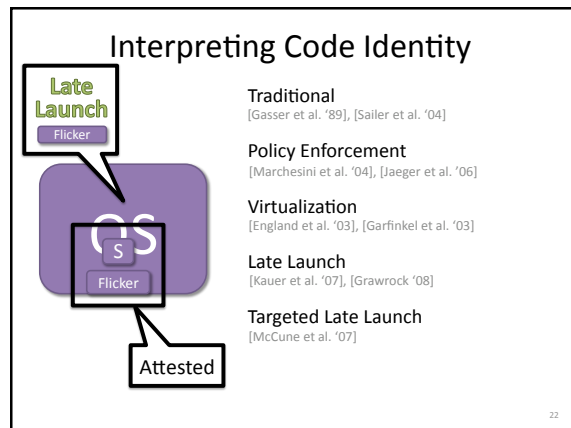
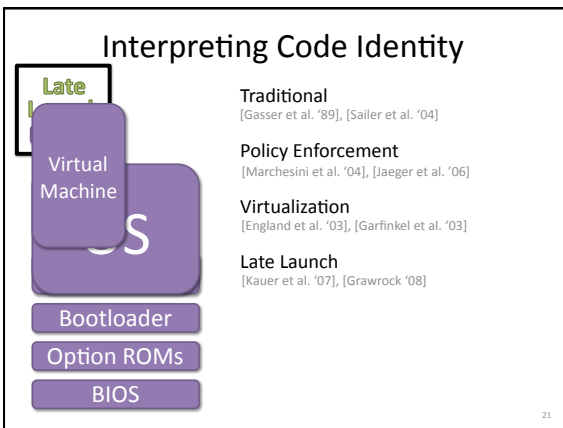
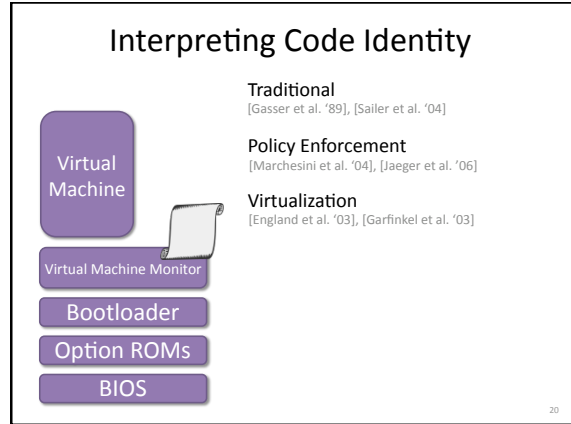
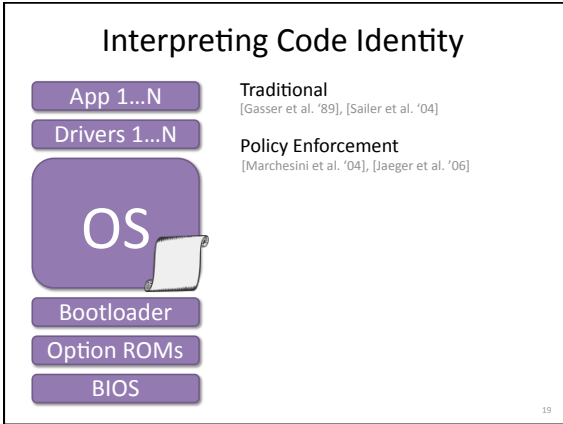
What Can Code Identity Do For You?

- **Research applications**
 - Secure the boot process
 - Count-limit objects
 - Improve security of network protocols
- **Commercial applications**
 - Secure disk encryption (e.g., Bitlocker)
 - Improve network access control
 - Secure boot on mobile phones
 - Validate cloud computing platforms

- Thwart insider attacks
- Protect passwords
- Create a Trusted Third Party

13





Load-Time vs Run-Time Properties

- Code identity provides load-time guarantees
- What about run time?

Open Question:
How can we get complete run-time properties?

The diagram shows a central box labeled 'Code'. To its left is a box labeled 'Attested' with an arrow pointing to 'Code'. Below 'Code' is a box labeled 'Enforcer' with an arrow pointing to 'Code'. To the right of 'Code' is a box labeled 'Run Time' with an arrow pointing to 'Code'. Below 'Enforcer' is a box labeled 'Load Time' with an arrow pointing to 'Enforcer'.

Roots of Trust

Cheaper →

Open Question:
What functionality do we need in hardware?

[Weingart '87] [ARM TrustZone '04] [Chun et al. '07] [Spinellis et al. '00]
 [White et al. '91] [TCG '04] [Levin et al. '09] [Seshadri et al. '05]
 [Yee '94] [Zhuang et al. '04] ...
 [Smith et al. '99] ...

Human Factors

The illustration shows a woman on the left holding a mobile phone. In the center, a bidirectional arrow connects her to a computer monitor on the right. Above the arrow are four boxes labeled 'SV', 'SV', 'SV', 'SW'. Below the arrow are four boxes labeled '1', '2', 'N', 'N'.

Open Questions:

Conclusions

- Code identity* is critical to bootstrapping trust
- Assorted hardware *roots of trust* available
- Many *open questions* remain!

Thank you!
parno@cmu.edu