# Trust

## What it is and how to get it

Dr. Perry Alexander

Information and Telecommunication Technology Center
Electrical Engineering and Computer Science
The University of Kansas
palexand@ku.edu

Formatted with the Beamer Class for $\LaTeX\,2_\varepsilon$

# Defining Trust

**Trust**

“An entity can be trusted if it always behaves in the exptected manner for the intended purpose [**?** ]”

### Properties

- Unambiguous identification
- Unimpeded operation
- First-hand observation of good behavior *or* indirect experience of good behavior by a trusted third party

# Capabilities for Trust

- *Strong Identification* — An unambiguous, immutable identifier associated with the platform. The identifier is a protected encryption key in the TXT implementation.
- *Reporting Configuration* — An unambiguous identification mechanism for software and hardware running on the platform. The mechanism is hashing in the TXT implementation

# Trust is a Preorder

$T^{\mathrm{x}}[\mathrm{y}]$ is an homogeneous relation over actors that is true when *x trusts y*. $T^{\mathrm{x}}[\mathrm{y}]$ is a preorer:

- Reflexive - $\forall x \cdot T^{\mathrm{x}}[\mathrm{x}]$
- Transitive - $\forall x, y, z \cdot T^{\mathrm{y}}[\mathrm{x}] \wedge T^{\mathrm{z}}[\mathrm{y}] \Rightarrow T^{\mathrm{z}}[\mathrm{x}]$

# Roots of Trust

As the name implies, a *root of trust* forms a base for transitively building trust. Roots of trust are trusted implicitly.
There are three important Roots of Trust:

- Root of Trust for Measurement (RTM)
- Root of Trust for Reporting (RTR)
- Root of Trust for Storage (RTS)

# Root of Trust for Measurement

A *Root of Trust for Measurement* is trusted to take the base system measurement.

- Typically a hash function called on an initial code base
- In the Intel TXT process the RTM is SENTER
- SENTER measures SINIT into PCR 18 in a TPM
- SENTER runs securely in the Intel processor

# Root of Trust for Reporting

A *Root of Trust for Reporting* is trusted to authenticate the base system report or quote

- Typically a key used for signing reports
- In the Intel TXT processes this is the TPM's Endorsement Key (EK)
- Created in the TPM factory and bound to its platform by a certificate
- $EK^{-1}$ is stored in the TPM
- $EK^{-1}$ cannot be accessed directly and can only be used by the TPM
- $EK^{-1}$ is not used for signing, but for decrypting credentials
- $EK$ is maintained by a Certificate Authority

# Chaining Trust

- Trust is transitive
  - $T^x[y] \wedge T^y[z] \Rightarrow T^x[z]$
  - Construct chains of trust
  - Remember "directly observed or indirectly observed by a trusted third party"
- Roots of Trust define the basis for trust
  - Use Roots of Trust to establish base for chain
  - RTM generates a trusted first measurement
  - RTS protects first measurement
  - rtr signs base quote for appraiser

# Root of Trust for Storage

A *Root of Trust for Reporting* is trusted to protect the base stored data

- Typically a key stored in a trusted location
- In the Intel TXT boot process this is the TPM's Storage Root Key (SRK)
- Created by TPM_TakeOwnership
- $SRK^{-1}$ is stored in the TPM
- $SRK^{-1}$ cannot be accessed directly and can only be used by the TPM

# Building the Base

1. Power On
   - PCRs reset to -1
2. SENTER is called (RTM)
   - SENTER resets PCRs to 0
   - SENTER hashes SINIT into PCR 18 (RTS)
3. SINIT is called (Trusted)

# Presentation Outline

- Review access control modeling objectives
  - modeling platform MAC
  - modeling local access control
- Overview access control policy definition
  - design and modeling assumptions
  - platform boot policy definition
  - local policy definitions
- Overview models
  - domain and system models
  - communication model
  - theorems and status
- Identify next steps
  - runtime and moving beyond the SVP line
  - adding M&A detail

# Access Control Modeling Objectives

What we're about here

Reporting joint work with Geoffrey Brown, Indiana University (submitted) in which we verify two physical layer protocols.

- Biphase Mark Protocol (BMP)
- 8N1 Protocol

These protocols are used in data transmission for CDs, Ethernet, and Tokenring, etc. as well as UARTs.

- Correctness is reasonably difficult to prove due to many real-time constraints.
- Many previous formal modeling/verification efforts for these protocols.

# Columns and Blocks

## Trying figures next to lists

Some normal text goes here
just for introduction

- Appraisal
- Measurement
- Attestation
- vTPM

Why is this column getting
higher?
Maybe it's not
Center alignment seems best.
I like this for two column test
and graphics
Getting higher???

# Big Picture

## Armor Architecture

# Simple Block

**Introduction to LaTeX**

Beamer is a LaTeXclass for creating presentations that are held using a projector..."

This is a definition

**Not really a proof.**

1. This is a step

$\square$

*Not really a proof.*

1. This is a step
2. This is another step

$\square$

**Not really a proof.**

1. This is a step
2. This is another step
3. This is a third step
4. This is a third step
5. This is a third step
6. This is a third step

$\square$

## List with Overlays

- Item 1 followed by a pause

# List with Overlays

- Item 1 followed by a pause
- Item 3 followed by a pause

## List with Overlays

- Item 1 followed by a pause
- Item 2 followed by a pause
- Item 3 followed by a pause

# Previous Efforts

- BMP has been verified in PVS twice and required
  - 37 invariants and 4000 individual proof directives (initially) in the one effort
  - 5 hours just to *check* the proofs in the other effort
  - A formal specification and verification of an independent real-time model in both efforts
- BMP has been verified in (the precursor to) ACL2 by J. Moore and required
  - A significant conceptual effort to fit the problem in the logic, arguably omitting some salient features of the model
  - The statement and proof of many antecedent results
  - J. Moore reports this as one of his “best ideas” in his career

# Not Your Father's Theorem-Prover

The verifications are carried out in the SAL infinite-state bounded model-checker that combines SAT-solving and SMT decision procedures to *prove* safety properties about infinite-state models.

- Theorem-proving efforts took multiple engineer-months if not years to complete.
- Our initial effort in SAL consumed about *two engineer-days*.
  ...and we found a significant bug in a UART application note.

# Parameterized Timing Constraints

SMT allows for *parameterized* proofs of correctness. The following are example constaints from the BMP verification:

```
TIME: TYPE = REAL;

TPERIOD: TIME = 16;
TSAMPLE: INTEGER = 23;

TSETTLE: {x: TIME |      0 <= x
                    AND (x + TPERIOD < TSAMPLE)
                    AND (x + TSAMPLE + 1 < 2 * TPERIOD)};

TSTABLE: TIME = TPERIOD - TSETTLE;

ERROR: {x: TIME |      (0 <= x)
                  AND (TPERIOD + TSETTLE < TSAMPLE*(1-x))
                  AND (TSAMPLE*(1+x) + (1+x) + TSETTLE < 2 * TPERIOD)};

RSAMPMAX: TIME = TSAMPLE * (1 + ERROR);
RSAMPMIN: TIME = TSAMPLE * (1 - ERROR);
RSCANMAX: TIME = 1 + ERROR;
RSCANMIN: TIME = 1 - ERROR;
```

# SRI's SAL Toolset

- Parser
- Simulator
- Symbolic model-checker (BDDs)
- Witness symbolic model-checker
- Bounded model-checker
- Infinite-state bounded model-checker
- Future releases include:
  - Explicit-state model-checker
  - MDD-based symbolic model-checking

All of which are "state-of-the-art"

Please direct your attention to the whiteboard.

# Timeout Automata[1] (Semantics)

An *explicit* real-time model.

- Vocabulary:
  - A set of state variables.
  - A *global clock*, $c \in \mathbb{R}^{0\leq}$.
  - A set of *timeout* variables $T$ such that for $t \in T$, $t \in \mathbb{R}^{0\leq}$.
- Construct a transition system $\langle S, S^0, \rightarrow\rangle$:
  - States are mappings of all variables to values.
  - Transitions are either *time transitions* or *discrete transitions*.
    - Time transitions are enabled if the clock is less than all timeouts. Updates clock to least timeout.
    - Discrete transitions are enabled if the clock equals some timeout. Updates state variables and timeouts.

[1] B. Dutertre and M. Sorea. Timed systems in SAL. *SRI TR*, 2004.

# Disjunctive Invariants

Even with *k*-induction, getting a sufficiently strong invariant is still hard! *Disjunctive invariants* help. A disjunctive invariant can be built iteratively from the counterexamples returned for the hypothesized invariant being verified.

```
t0:  THEOREM system |-
       G(   (    (phase = Settle)
             AND (rstate = tstate + 1)
             AND (rclk - tclk - TPERIOD > 0)
             AND (tclk + TPERIOD + TSTABLE - rclk > 0))
         OR
            (    (phase = Stable)
             AND (rstate = tstate + 1)
             AND (rclk - tclk - TSETTLE > 0)
             AND (tclk + TPERIOD - rclk > 0)
             AND (rdata = tdata))

                         .
                         .
                         .
```