



Systems and Internet Infrastructure Security

Network and Security Research Center
Department of Computer Science and Engineering
Pennsylvania State University, University Park PA

Towards Integrity-Verifiable Mobile Systems

Trent Jaeger

*Systems and Internet Infrastructure Security (SIIS) Lab
Pennsylvania State University*

April 11, 2008

- Components
 - ▶ Mobile Phone Devices
 - ▶ Base Stations
 - ▶ Various Telecommunications Servers

- Voice
- Data
- Owned by Multiple Vendors

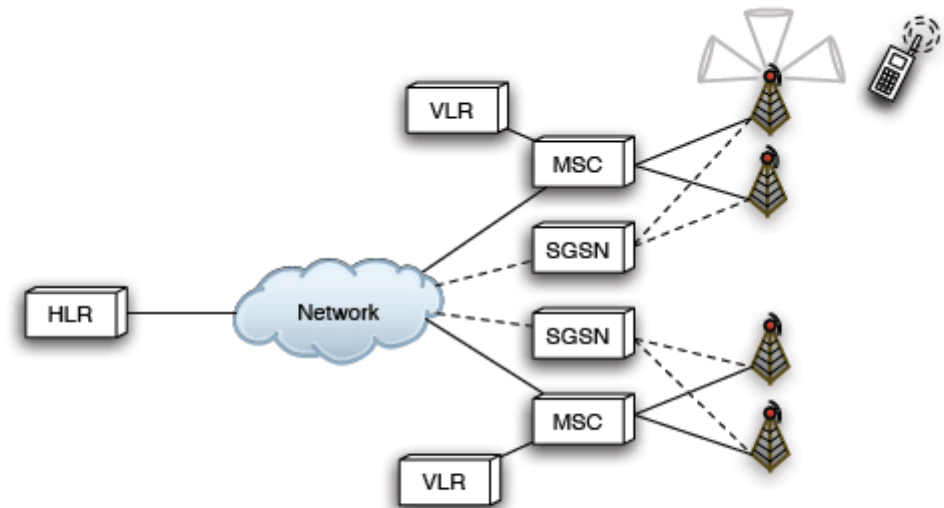


Figure 1: A high level description of GSM network architecture. Mobile phone register for voice and data services through the MSC/VLR and SGSN, respectively.

Phone Devices

- Formerly closed, unusual systems
 - ▶ Then, general purpose, phone-specific OS
 - Symbian
 - ▶ Now, general purpose OSes
 - Windows Mobile, Linux
 - ▶ *With all their baggage!*
- **Claim:** Can get malware on phones easily
 - ▶ Lots of open, privileged software
 - trusts phone programs (and attackers)
 - ▶ *Can attack the phone and the network!*



Phone Malware Delivery

- We have SMS/MMS
 - ▶ *How about an attachment?*
- We have Bluetooth
 - ▶ An insecure, wireless communication
 - ▶ *How about a some free coupons?*
- We have software downloads
 - ▶ Such as freeware games
 - Now, 2M plus US users download a freeware game (March 2006) from itfacts.biz
 - ▶ *How about some free software?*
- We have cheap, eBay phones



Symbian Malware Download

- Nokia 9500, Symbian Series 80
- Bluetooth
 - ▶ Connection-oriented -- “pairing”
 - User must approve connect -- first only
- Bluetooth: Takes a couple of clicks to install a file
 - ▶ (1) Accept code; (2) Install code
 - ▶ Pairing entry makes (1) unnecessary
- Lots of flexibility for attacker
 - ▶ *Can generate fake messages for each step!*
 - ▶ *Can enter Bluetooth pairing entries (w/o detection)!*



What Can Malware Do?

- On a Symbian phone
- Some resources are difficult to breach (!)
 - No known kernel compromises
- Hierarchical write-integrity model (like Vista)
 - System
 - Symbian-signed
 - Other
 - ▶ Symbian prevents a process originating from a lower “level” from modifying a file at a higher level
- We cannot easily modify files in a Symbian-signed directory from a third-party process

What Can Malware Do?

- Data Secrecy
 - ▶ *Phone systems do not protect secrecy at all!*
 - ▶ Download secret files (e.g., contacts database)
 - *Nearly any file!*
- Server Programs Trust All Processes
 - ▶ *Phone integrity can be compromised by servers programs!*
 - ▶ Server programs: E.g., windowing, telephony, installer, bluetooth, etc.
 - *Accept input from any process running on the system*
 - *Even ones that aren't on the system... (Bluetooth)*

What Can Malware Do?

- *Some specific experiments...*
- **Keylogger**
 - ▶ Window server enables multiple clients to request callbacks on keystrokes
 - E.g., obtain key entries for interacting with banking system
 - ▶ Can also record DTMF (sound of buttons)
- **Telephony Misuse**
 - ▶ Submit any AT command implemented on the phone
 - *Even over Bluetooth!*
 - ▶ Make calls from another phone over Bluetooth
 - ▶ Create spurious work for the network

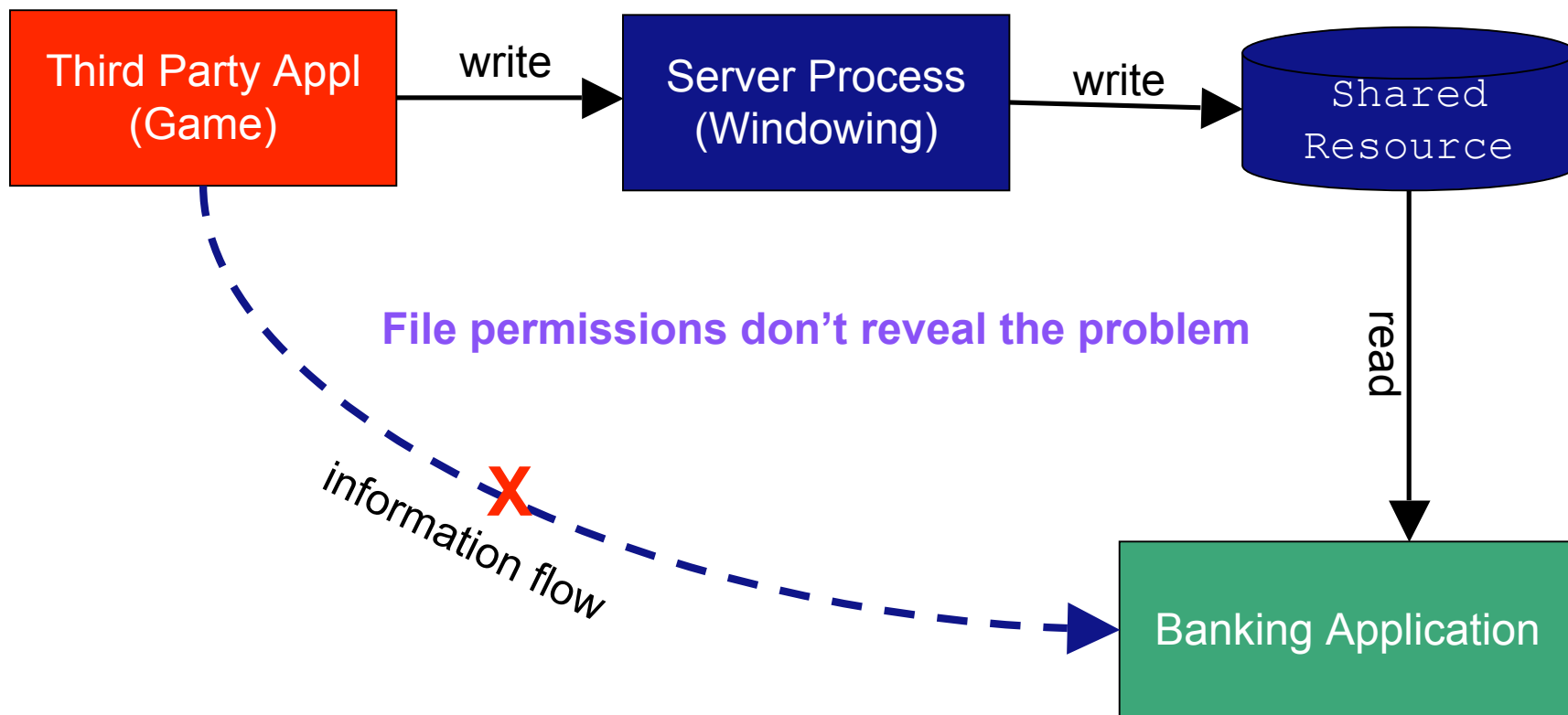
- *Provable and Practical Secrecy and Integrity*
 - ▶ Objective: “firewall” between trusted processes and untrusted processes/inputs
 - **Define perimeter**, allowed information flows, justification of protection
 - ▶ Before it is too late...
 - **Measure the integrity** of the resulting system, enforcement of expected flows
 - ▶ *Prove to others*
- *Experience from Symbian Study*
 - ▶ Small number of integrity levels (but for read and write)
 - ▶ But, servers may interact with low integrity processes
 - ▶ Protection of user data aligns with integrity



Integrity Approaches

- Goal: Prevent compromise of trusted applications

Integrity property: Trusted processes don't depend on untrusted ones

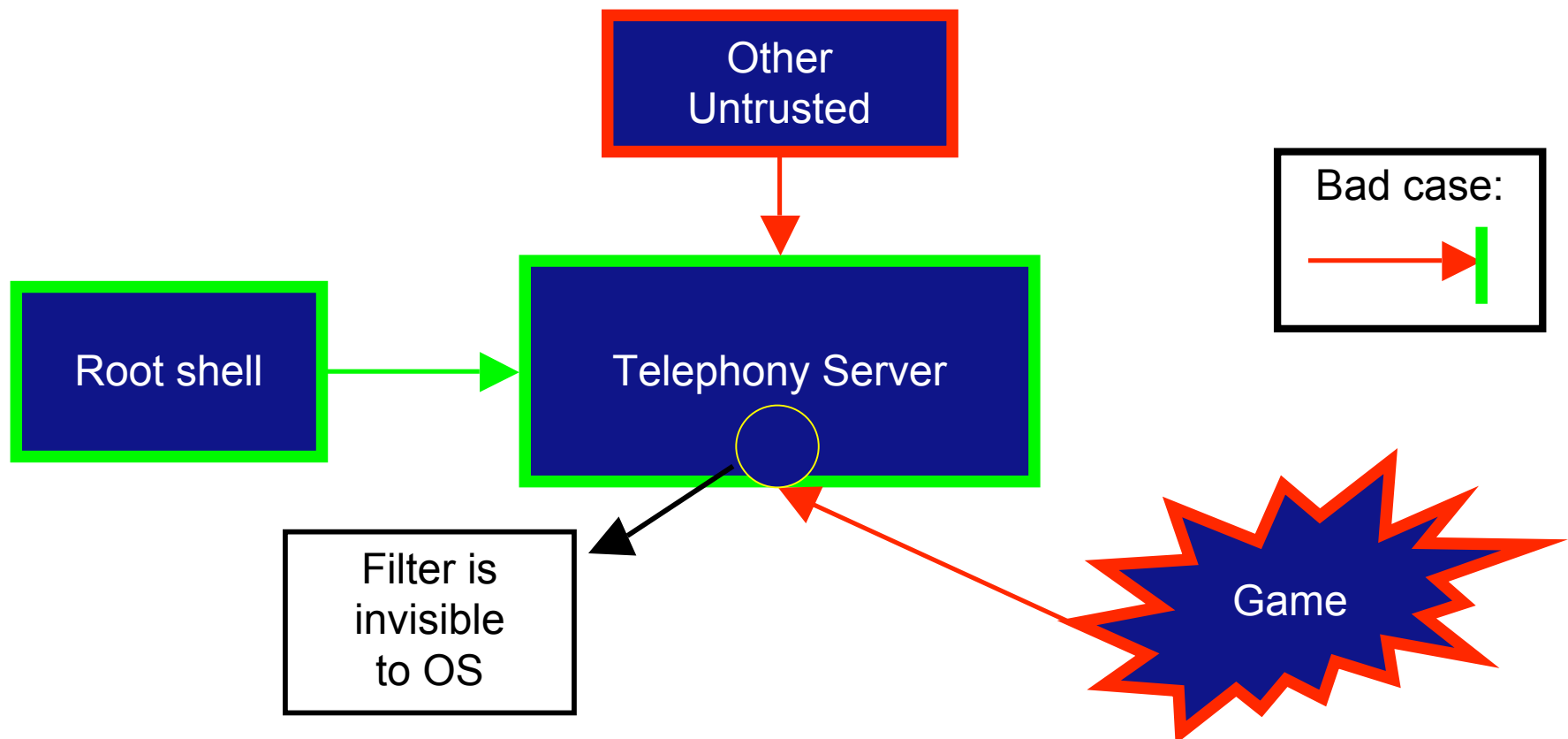


Some Integrity Challenges

- **Prevent Unauthorized Access to the Banking Application**
 - ▶ No low integrity information flows to the banking application (Biba)
 - ▶ MAC Policy and reachability analysis (lots of prior work)
 - *Many false positives – Not acceptable*
- **Prevent Server Compromise**
 - ▶ Trusted processes must “discard or upgrade” low integrity inputs (Clark-Wilson)
 - ▶ Cannot expect full formal assurance -- *Too difficult/expensive*
- **Prevent “Confused Deputy” in Server**
 - ▶ Limit permissions based on requestor
 - ▶ Server must prove it enforces system policy -- *Enough policy already!*

Prevent Server Compromise

- Practical Integrity Models
 - ▶ We like “CW-Lite” [NDSS 06], but there are others
- Practical Enforcement -- LSM or virtualization or...

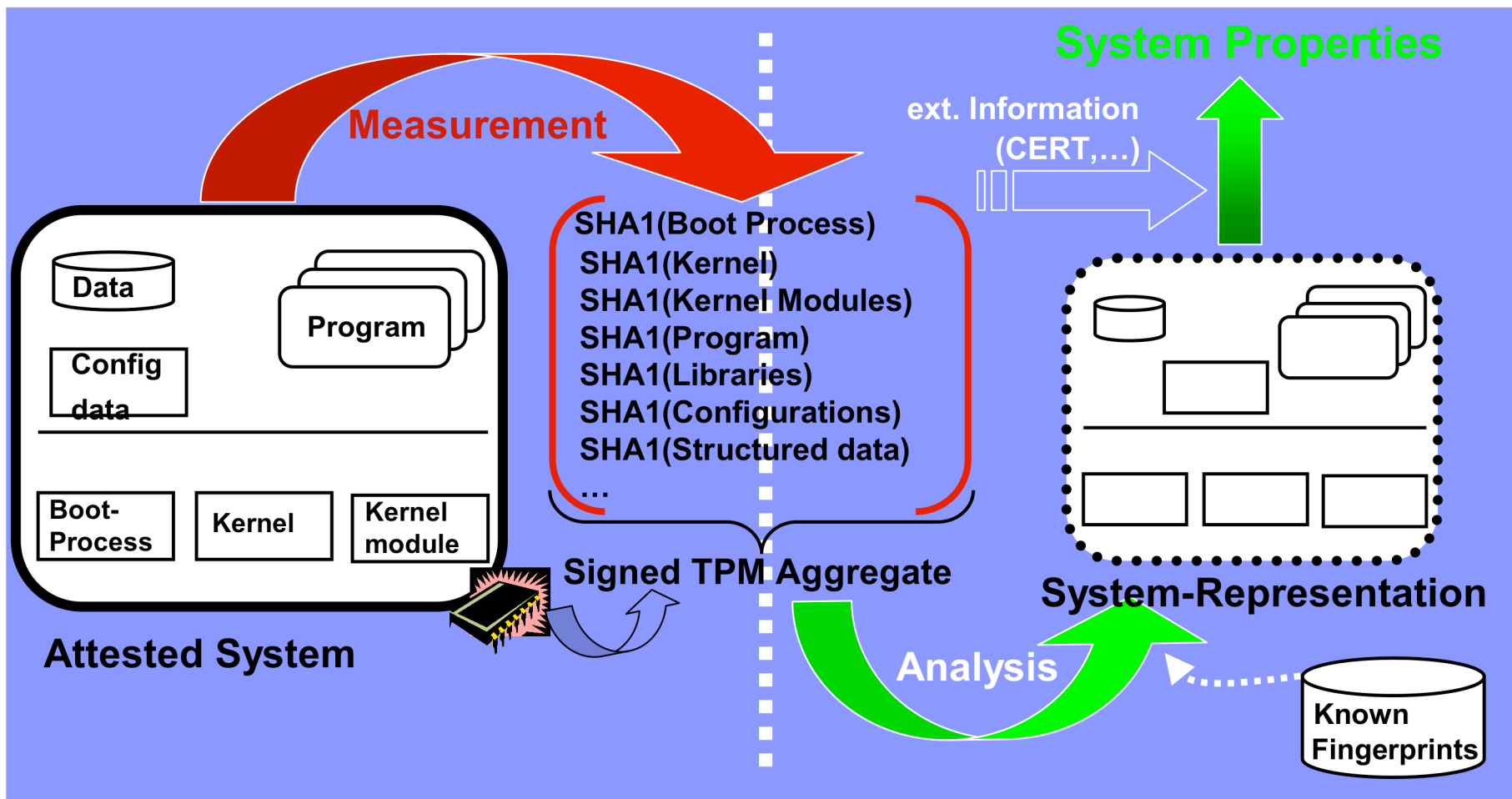


Prevent Confused Deputy

- Can simplify system policy
 - ▶ Define *integrity perimeter*
 - CW-Lite (or others) captures exceptions (allow)
 - ▶ Only one user in system
- Verify that each server enforces system policy
 - ▶ Convey system policy to server (“channels”)
 - ▶ Need a *reference monitor*
 - ▶ Ensure server fulfills system reference monitor guarantees [USENIX Security 08]

Measure System Integrity

- E.g., Linux IMA [USENIX Security 2004]



- Proof Dimensions: origin, time, composition, heterogeneity

- Opportunities
 - ▶ All mobile system devices
- Telecommunications Systems
 - ▶ All trusted code
 - ▶ Based on a trusted installation [ACSAC 07]
 - ▶ *Explore protocol designs that leverage integrity measurement*
- Phone Devices
 - ▶ Measure within the integrity perimeter (PRIMA measures info flow)
 - Applied to phone systems
 - ▶ *Explore how to enable effective verification*

- *Currently, Verification is too difficult*
 - ▶ How does the verifier know what code is trusted?
 - ▶ How does the verifier know what code was really running (i.e., no runtime attacks)?
 - ▶ How does the verifier know that the data has been protected across boot cycles?
 - ▶ Can we provide the verifier with guarantees about the future?
 - ▶ Can we simplify the proofs for the verifier?

- Phone systems are becoming general purpose
 - ▶ *Malware will find phones*
 - ▶ *Phone software trusts its applications too much*
- A variety of attacks are currently possible
 - ▶ *Need an approach for ensuring that devices can be “responsible”*
- Configure and measure high integrity phone systems
 - ▶ *Simplified systems and policies*
 - ▶ *Integrity measurement*
- Also, develop integrity for telecommunications servers

- Research projects
 - ▶ <http://www.cse.psu.edu/~tjaeger/research/>
- Penn State SIIS Lab
 - ▶ <http://siis.cse.psu.edu/>
- Email
 - ▶ tjaeger@cse.psu.edu