Faculty of Computer Science Institute for System Architecture, Operating Systems Group

SwitchBlade: Enforcing Dynamic Personalized System Call Models

Martin Süßkraut, Christof Fetzer

presented by Bjoern Doebel

Dresden, 2010-01-06

Highway of Threats


UNAUTHORIZED ACCESS


MOBILE DEVICE ATTACK


SYSTEM COMPROMISE


CYBER ESPIONAGE


SOCIAL ENGINEERING

SPAM


MALWARE


INSIDERS


DENIAL OF SERVICE


DATA LEAKAGE


PHISHING


IDENTITY THEFT

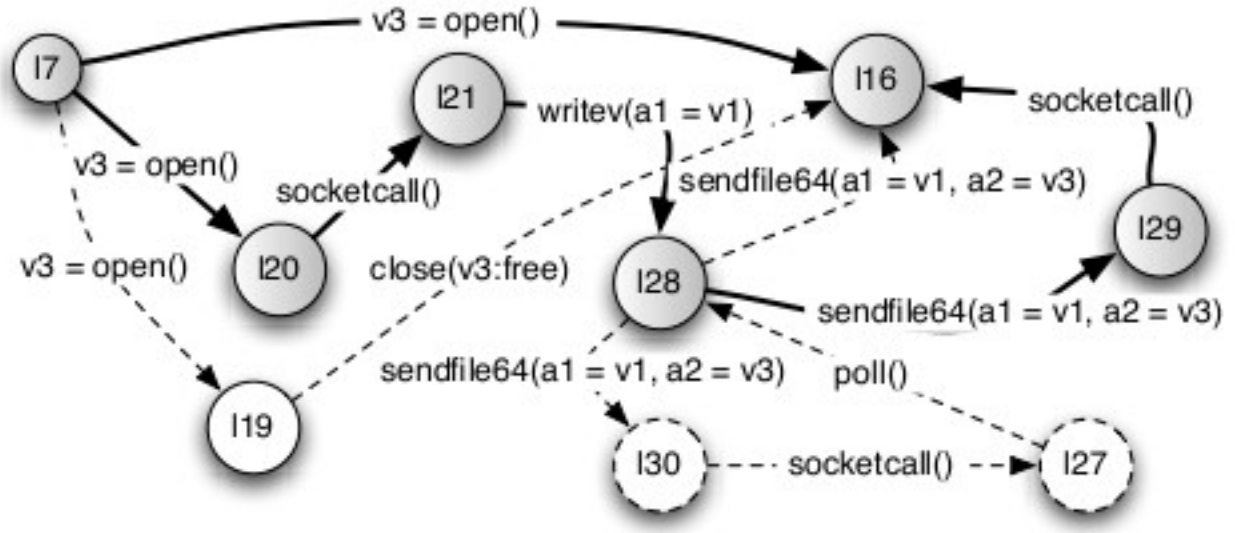
Are you Vulnerable?

Detecting malware

- Applications perform system calls.
- Malicious system call behaviour may differ from original program.
  - Anomaly-based detection
    - Statistics about system call sequences
    - Attacker can hide behind fake system calls
  - Misuse-based detection
    - Static system call model

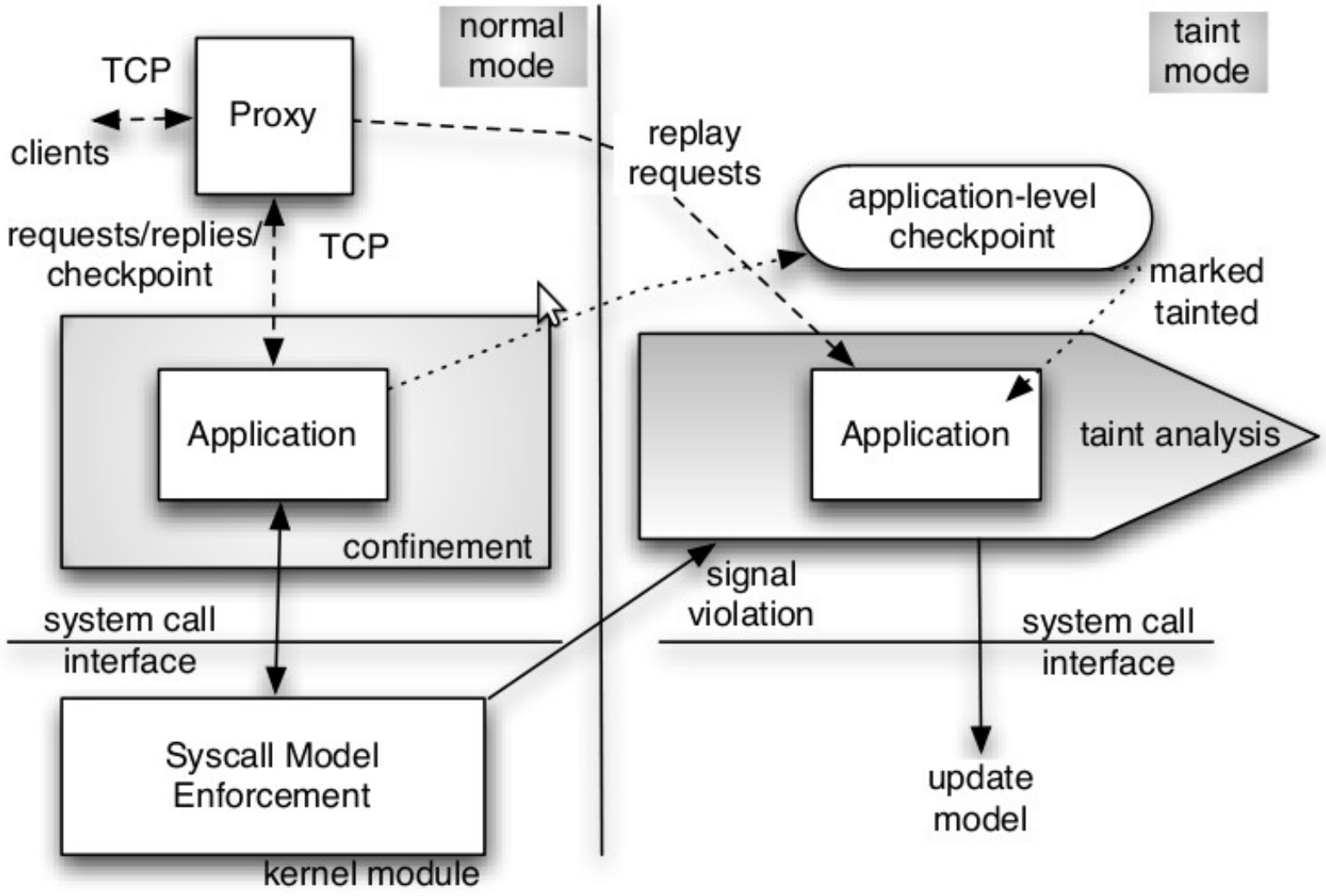
System Call Models

- Nodes := stack traces
- Edges := system calls including parameters, variables etc.



TECHNISCHE UNIVERSITÄT DRESDEN

Switchblade Architecture



Preventing exploitation

- Model personalization
  - Generate model locally
    → decreased attack vectors
    → increased overhead
    → increased false positive rate

- Model randomization
  - Insert random invalid system calls into application (e.g., by wrapping functions)
  - Works if attacker does not get to see wrapper code (execute-only pages)

Evaluation

- Worked for exploits tested.
- Models
  - Tens (Apache) to hundreds (vim) of nodes
  - Pretty workload-specific
- Overhead
  - Normal mode: 18 – 81 %
  - Taint mode: >5,000 %
  - Larger than other tools, but more fine-grained model

Discussion

- Circumvent Switchblade?
  - Attacks are not prevented – we can still inject exploit code and get access to all program data.
  - For interesting applications, syscall parameter constraints may not be useful
    - web browser spawns arbitrary processes for plugins
    - web browser creates arbitrary internet connections and sends arbitrary data
  - Valgrind is not meant to prevent applications from breaking out of its control.