Secure Xen on ARM: Status and Driver Domain Separation

Article · January 2007

CITATIONS
4

READS
246

1 author:

Sang-bum Suh
Samsung Advanced Institute of Technology
38 PUBLICATIONS **335** CITATIONS

SEE PROFILE

Secure Xen on ARM: Status and Driver Domain Separation

Sang-bum Suh
sbuk.suh@samsung.com
SW Laboratories
Corporate Technology Operations
Samsung Electronics
Presented at Xen Summit Autumn 2007, Sun Microsystems

Contributors

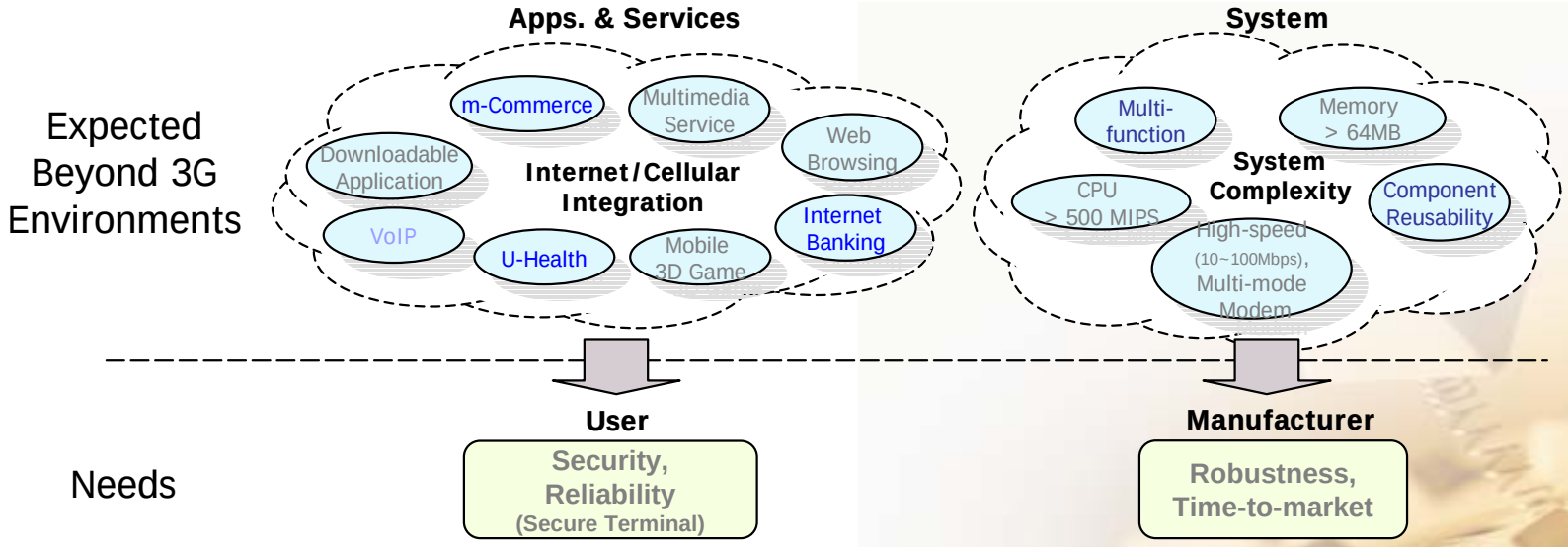
- Sang-bum Suh
- Sung-min Lee
- Joo-young Hwang
- Jung-hyun Yu
- Sangdok Mo
- Chanju Park
- Bokdeuk Jeong
- Sungkwan Heo
- Jaemin Ryu
- Jun-young Sim
- Dong-hyuk Lee
- Igor Nabirushkin
- Alexander Trofimov
- Mikhail Levin
- Il-pyung Park
- Ho-soo Lee

Contents

Overview and Status of Secure Xen on ARM Architecture 1.0

Requirements for Beyond 3G Mobile Phone

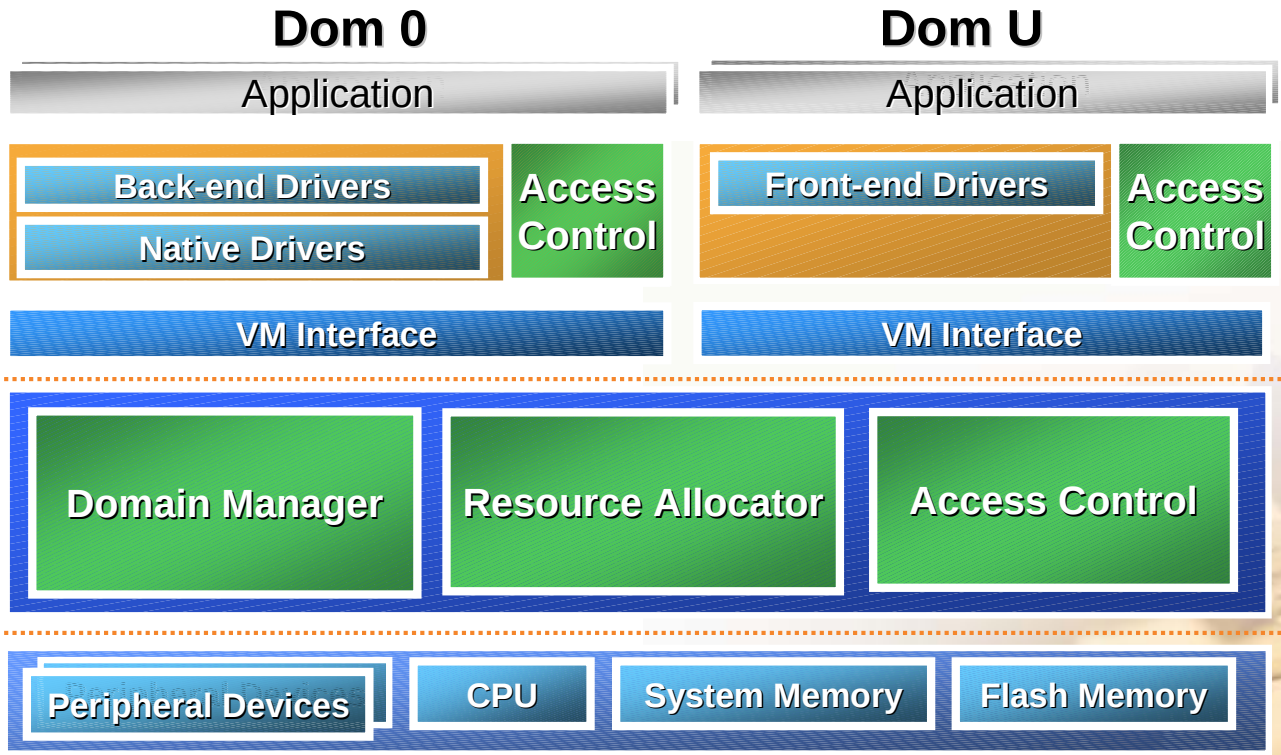
- End user: Secure and reliable mobile terminals for mobile Internet services using WiBro
- Manufacturer: Robustness though complexity of devices gets increased
- Contents provider: Protection of IP rights in end-user terminals
- Carrier companies: Open and Secure Mobile Platform
  - OSTI (Open Secure Terminal Initiative): NTT DoCoMo, Intel



Beyond 3G environments and Needs

Goal and Architecture

- Goal
  - Light-weight secure virtualization technology for beyond 3G mobile phone
- Approach
  - Design and implementation of
    - VMM on ARM using Xen architecture: Xen on ARM
    - Security features using Xen on ARM:
      secure boot, secure SW installation, multi-layer fine-grained access control



Development Environments

◆ HW and SW Environments

❖ A Reference System for Implementation

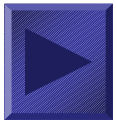
- SW
  - Xen : Xen-3.0.2
  - Linux : ARM Linux-2.6.11
  - GUI : Qtopia
- HW
  - Processor : ARM-9 266Mhz (Freescale i.MX21)
  - Memory : 64MB
  - Flash : NOR 32MB / NAND 64MB
  - LCD : 3.5 inch
  - Network : CS8900A 10Base-T Ethernet Controller

❖ Development Environments

- OS : Fedora Core 6
- Cross-compiler: Montavista ARM GCC 3.3.1
- Debugger : Trace32 ICD (In Circuit Debugger)

Status of Secure Xen on ARM Architecture 1.0

- Xen on ARM:
  - Performance improved
  - Video demo: game on Dom 0 and application/Qtopia on Dom U



video1

- Xen Security features:
  - 5 access control modules and visualization supported:
    - Type Enforcement, Samsung proprietary, BiBA, Bell LaPadula, Chinese wall
    - GUI-based access control policy manager
  - Video demo: access control mechanism against phishing attack

page21

- Driver domain separation: architecture exploration

Driver Domain Separation: Architecture Exploration

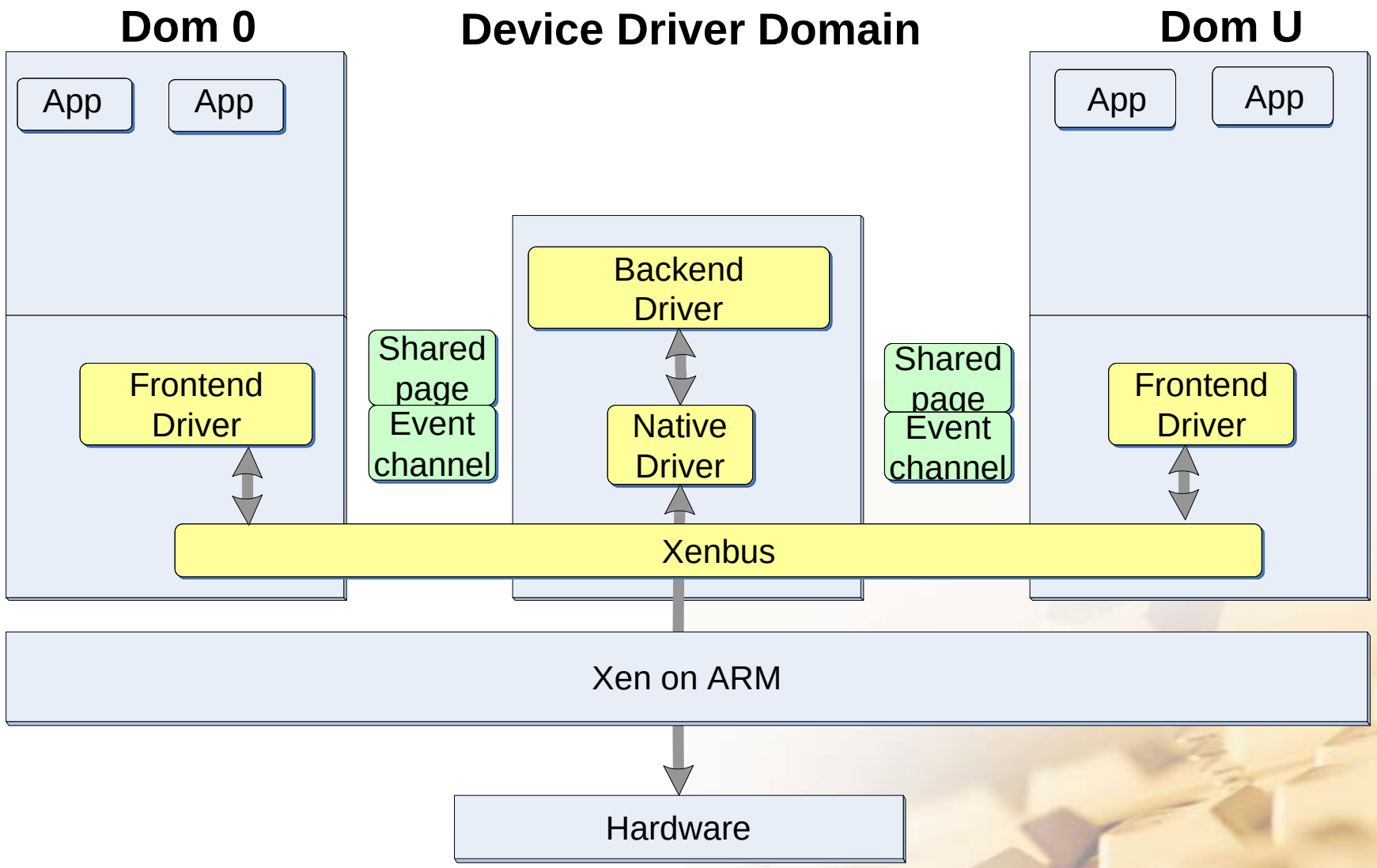
Motivation

- Many downloadable services under beyond 3G mobile environments will be increased.
  - This requires an open mobile platform.

- Open platform will face problems with malware and bugs similar to PC.
  - Secure Xen can help an open mobile platform secure against malware.
  - However, bugs in device drivers may cause Dom 0 to stop working and the applications to have to restart.
    - Relatively short life cycle of peripheral chips in consumer electronics products.
      - Can test cases be updated quickly and be used to detect every bug during development ? Patch is likely.

☞ Device driver domain to be separated from Dom 0 (security applications running on Dom 0 in secure Xen on ARM) kernel.

Driver Domain Separation: Architecture



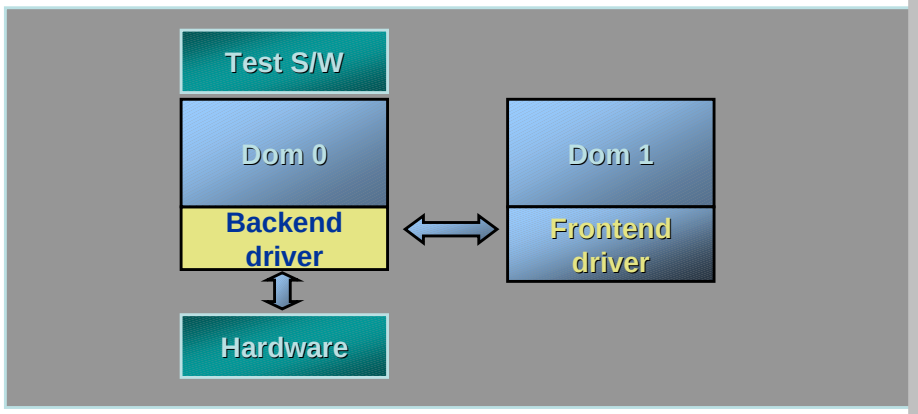
Summary

- Device driver domain
  - Xen-Linux kernel, access control module, backend and native drivers
  - Modification
    - RAMFS used for driver domain during booting
    - Xenbus, Xenstore, and Xen tools modified
    - Booting procedure modified
      - Booting Dom 0 => creating Device Driver Domain => initializing split device driver
- Advantage
  - Service availability can be improved even under driver fault
    - Dom0 and Dom U can work, while due to device driver failure, driver domain has to be restarting.
- Disadvantage
  - Performance degradation due to domain switching between Driver Domain and Dom 0

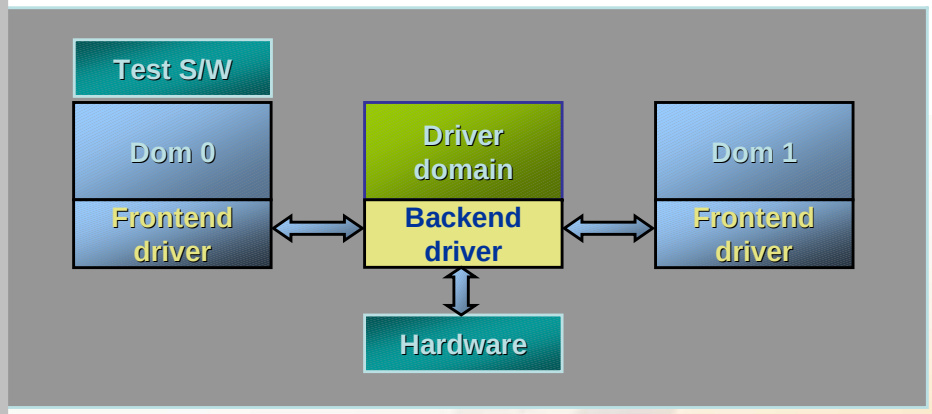
Performance (1/2)

- Environments
  - Virtual Network
  - HW Platform: Freescale i.MX21
    - 266Mhz ARM926lrmsdmq
    - Memory: 64MB DDR
    - Network: CS8900A 10Base-T Ethernet Controller

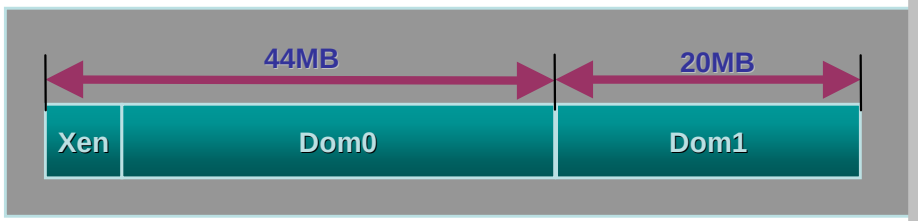
- **Backend in dom0**

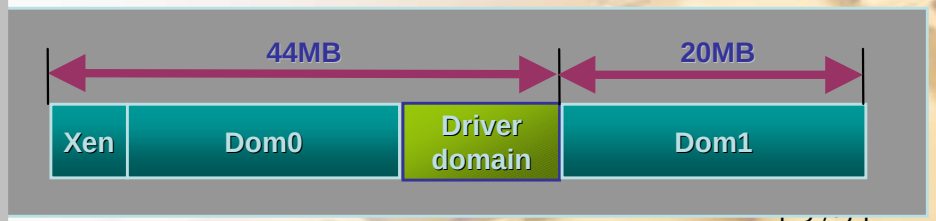


- **Backend in driver domain**



- **System memory configuration**

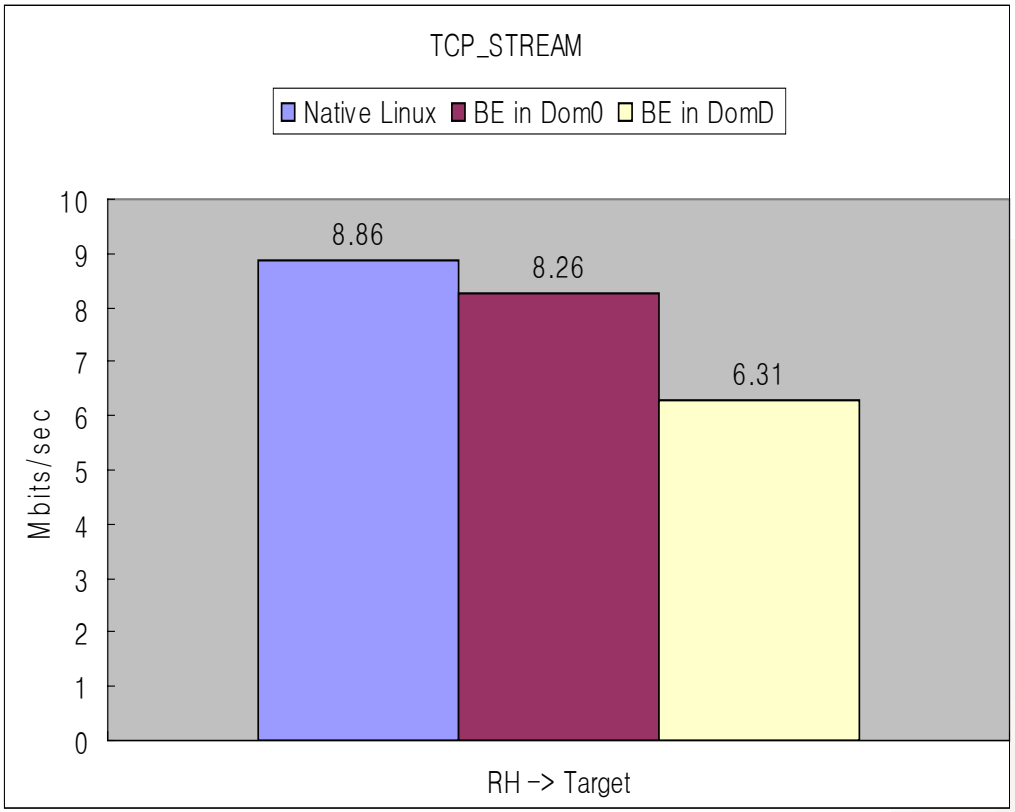




* Driver domain and Dom1 use Ramfs as a root file system.

Performance (2/2)

- Network Test: Netperf BMT
  - Due to a problem with DMA of the HW, performance is degraded further.



*** Native Linux:** network driver in native Linux
*** BE in Dom0:** Backend driver in Dom0
*** BE in DomD:** Backend driver in drive domain
*** RH:** Remote Host

*** TCP_STREAM:** Measuring a bulk data transfer throughput

Future Work

- Performance improvement of driver domain separation
- Minimal OS kernel for driver domain
- State migration

Thank you for attention

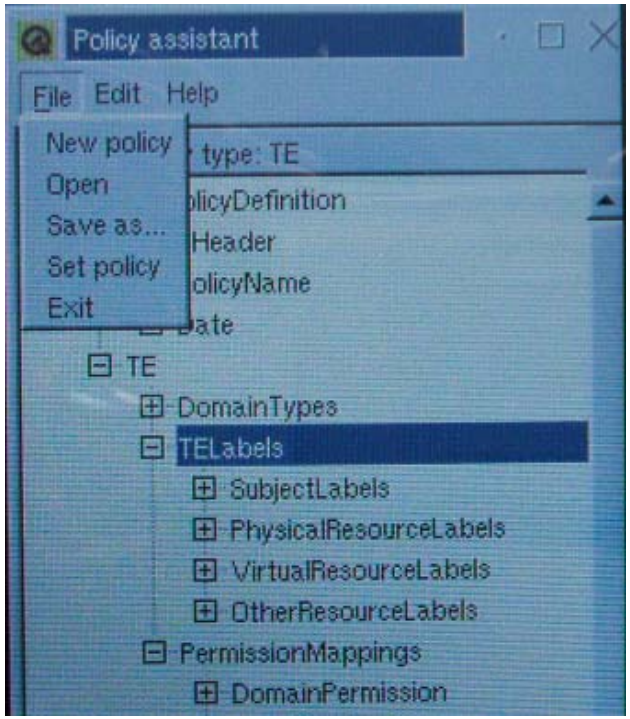
Appendix

Access Control Module (1/2)

- Supporting 5 access control models
  - Type Enforcement
    - A classical access control model which can be enforced for comprehensive system resources protection
    - Physical/virtual resources access control
  - Proprietary
    - Protecting a mobile device from resource drain attacks (e.g., CPU, memory, battery)
  - Bell LaPadula
    - Confidentiality model
    - Virtual resources access control where there are many domains (Good for controlling information flow with security level)
  - Biba
    - Integrity model
    - Virtual resources access control where there are many domains (Good for controlling information flow with security level)
  - Chinese Wall
    - Preventing simultaneous execution of multiple domains where the domains have different interests (i.e., assigned to conflict set)

Access Control Module (2/2)

- GUI-based policy manager
  - Edits XML-based access control policies
  - Sets new access control policies dynamically




```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- <SecurityPolicyDefinition xmlns="te_policy.xsd">
  -->
- <SecurityPolicyDefinition>
  + <PolicyHeader>
  - <TE>
      <!--  Definition of TE types which classify resources  -->
    + <DomainTypes>
    - <TELabels>
        <!--  Virtual machine labeling  -->
      - <SubjectLabels>
        + <VMLabel>
        + <VMLabel>
        + <VMLabel>
        </SubjectLabels>
        <!--  Physical resource labeling  -->
      - <PhysicalResourceLabels>
        - <PhyResourceLabel>
            <Type>te_KPP</Type>
            <PIRQLabel>KPP</PIRQLabel>
            <IOMEMLabel>KPP</IOMEMLabel>
          </PhyResourceLabel>
        - <PhyResourceLabel>
            <Type>te_Flash</Type>
            <IOMEMLabel>FLASHMEM</IOMEMLabel>
          </PhyResourceLabel>
        - <PhyResourceLabel>
            <Type>te_CONSOLE</Type>
            <PIRQLabel>UART1</PIRQLabel>
            <IOMEMLabel>UART1</IOMEMLabel>
          </PhyResourceLabel>
        </PhysicalResourceLabels>
```

Example of the XML-based TE policy

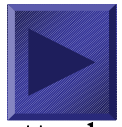
Secure SW Installation

- Basic assumptions about software on the secure domain
  - A small set of software (not much) can be installed by only trusted parties (i.e., manufacturer or service providers verified by the manufacturer)
  - The trusted parties must rigorously test the software based on advanced quality assurance methodology during the development phase
- Secure SW installer installs only software digitally signed by a manufacturer
- Access control at the secure domain (Dom0) allows only authentic secure SW installer to create executable files on the domain
  - Even in case a device owner downloads or creates files on the secure domain, they cannot be executed

Demonstration Scenario

- Connecting to a phishing site
  - Alice connects to a phishing server with her mobile phone after receiving an email fraudulently saying launch of UCC services from her favorite web site
  - She downloads and installs malware masqueraded as genuine SW from that site

- With a conventional single OS-based mobile phone



video2

  - Malware corrupts kernel and sends her sensitive information to an attacker while she is using the Internet banking service

- With a secure Xen-based mobile phone (with secure domain and normal domain)

video3

  - Even in case malware corrupts kernel of the normal domain, there is no information leakage or availability threat owing to domain separation and mandatory access control
  - Secure SW installer installs Gifviewer signed by a manufacturer successfully but fails to install Pacman whose digital signature is invalid
    - Assumption: communication channel between the secure SW installer and manufacturer site which provides downloadable SW is encrypted