IBM Research

Integrity Measurement Policies

Mimi Zohar, David Safford, Reiner Sailer

Linux Integrity Module (LIM)
Integrity Measurement Architecture (IMA)

- **LIM – a Kernel Framework for Integrity Features**
  - Integrity Measurement API
    - Collection (e.g. hashing inode data)
    - Appraisal (e.g. is this a “good” measurement?)
    - Storage (e.g. commit to TPM for attestation)
- **IMA – LIM provider for inode measurements**
  - Measurement **collection**, caching, sharing
  - Measurement list **storage**, TPM based attestation
- **EVM – Verifying file measurements**
  - HMAC appraisal of file data, metadata (selinux labels)

Measurement Challenge: What to measure?

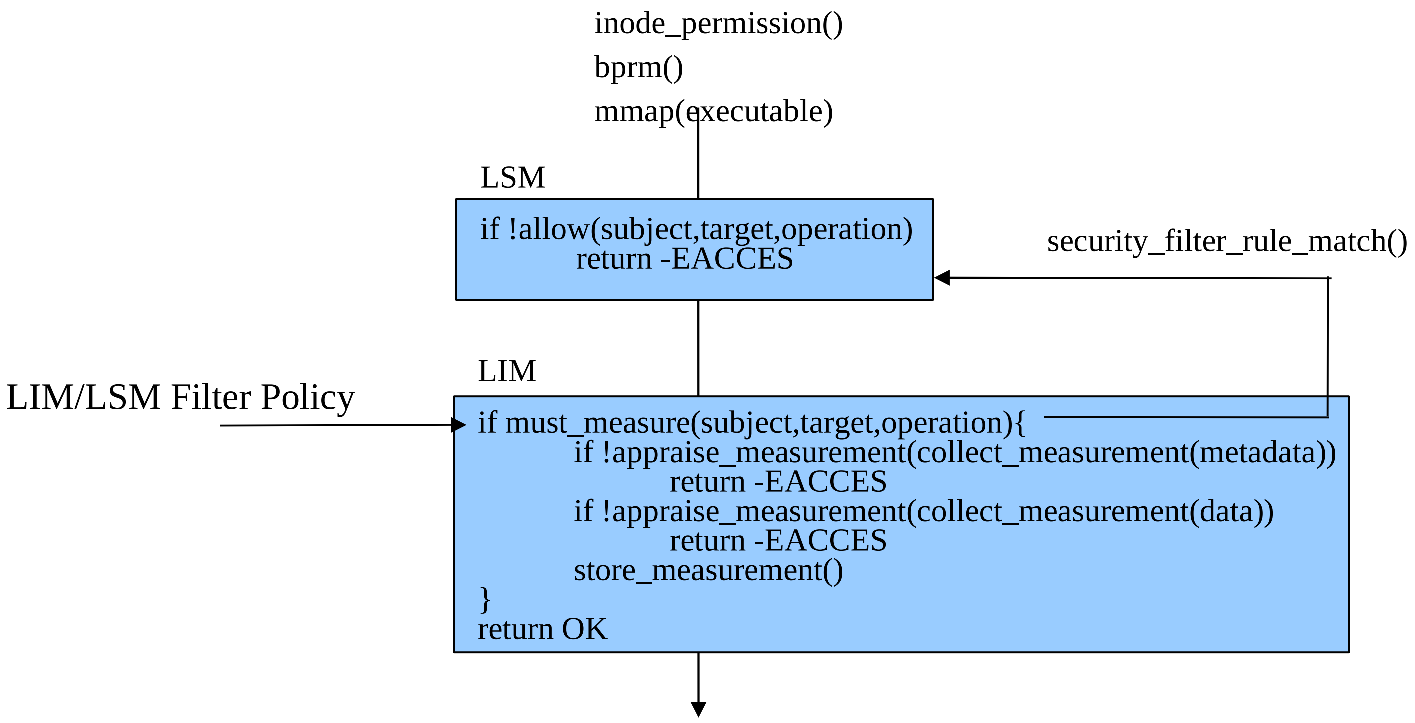
- **Want to measure all files, but unacceptable performance**
- **Some measurement decisions are easy:**

  All executed files, #! scripts (bprm hook)
  All files mmap'ed executable (mmap hook)
- **Some Read()'s are sensitive, but not all...**

  scripts, config files are sensitive
  NOT – log files, LARGE files (KVM images...)

Need a measurement policy integrated with LSM, to take advantage of selinux subject, object, type labels

Measurement Policies and Selinux



inode_permission()
bprm()
mmap(executable)

LSM

```
if !allow(subject,target,operation)
        return -EACCES
```

security_filter_rule_match()

LIM

LIM/LSM Filter Policy

```
if must_measure(subject,target,operation){
          if !appraise_measurement(collect_measurement(metadata))
                    return -EACCES
          if !appraise_measurement(collect_measurement(data))
                    return -EACCES
          store_measurement()
}
return OK
```

Default Measurement Policy

```
Rule format:  action subj= obj= type= func= mask= fsmagic=

  action is one of measure or dont_measure
  subj, obj, type are LSM string
  func is one of INODE_PERMISSION, FILE_MMAP, BPRM_CHECK
  mask is one of MAY_READ, MAY_WRITE, MAY_APPEND, MAY_EXEC
```

```
static struct integrity_measure_rule_entry default_rules[] = {
  {.action = DONT_MEASURE,.fsmagic = PROC_SUPER_MAGIC},
  {.action = DONT_MEASURE,.fsmagic = SYSFS_MAGIC},
  {.action = DONT_MEASURE,.fsmagic = DEBUGFS_MAGIC},
  {.action = DONT_MEASURE,.fsmagic = TMPFS_MAGIC},
  {.action = DONT_MEASURE,.fsmagic = SECURITYFS_MAGIC},
  {.action = MEASURE,.func = FILE_MMAP,.mask = MAY_EXEC},
  {.action = MEASURE,.func = BPRM_CHECK,.mask = MAY_EXEC},
  {.action = MEASURE,.func = INODE_PERMISSION,.mask = MAY_READ},
};
```

Example Selinux/Measurement Policy

```
#
# Integrity measure policy
#
# PROC_SUPER_MAGIC
dont_measure fsmagic=0x9fa0
# SYSFS_MAGIC
dont_measure fsmagic=0x62656572
# DEBUGFS_MAGIC
dont_measure fsmagic=0x64626720
# TMPFS_MAGIC
dont_measure fsmagic=0x01021994
# SECURITYFS_MAGIC
dont_measure fsmagic=0x73636673
measure func=BPRM_CHECK
measure func=FILE_MMAP mask=MAY_EXEC
measure subj=system_u func=INODE_PERMISSION mask=MAY_READ
```

Quick Demo/Example

- **This System is running**
  - Fedora 9
  - Selinux in enforcing mode, with a targeted policy
  - Linux-2.6.26-git3
  - LIM Framework
  - IMA with SELinux based measurement policy