Extending the Linux Integrity Subsystem for TCB Protection

David Safford, safford@us.ibm.com
Mimi Zohar, zohar@linux.vnet.ibm.com

Linux Integrity Subsystem: Review

- IMA-measurement (Linux 2.6.30): "**trusted boot"**
  –Hardware TPM signs list of file hashes, attests to third party
  –Complete TCB verification, but verification does not scale

- IMA-appraisal (Linux 3.7): "**secure boot"**
  – Verifies signatures on files
  – Scalable prevention, but does not do attestation

- IMA ima-sig template "**combined model**"
  – Attestation of hashes and signatures, but does not protect TCB

- This work: IMA-locked "**locked down**"
  – Policy based complete protection and attestation of TCB

What is the Linux TCB, and what is immutable?

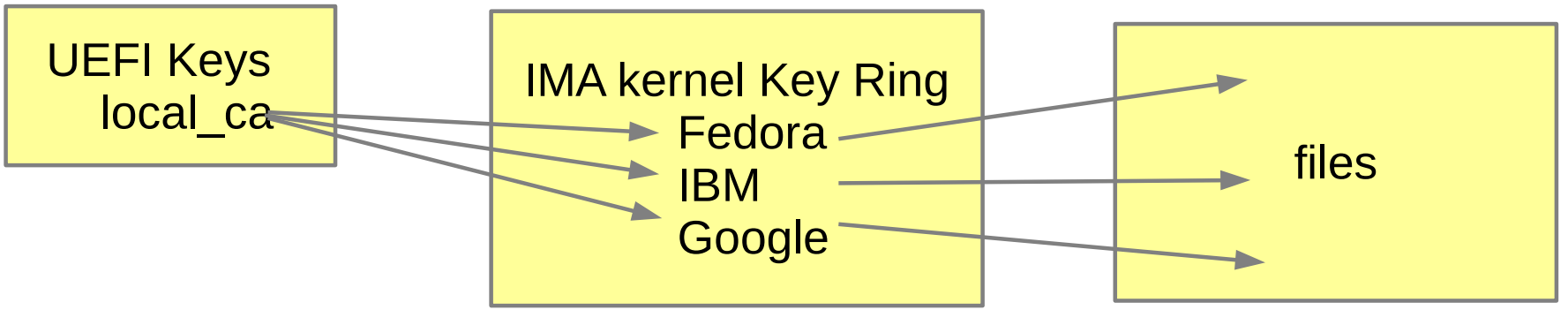
- *The set of all hardware, firmware, and software components that are critical to a system's security* - Butler Lampson

- This is a simple definition, but it's really hard to determine in practice
- Some things are obvious:
  - Kernel, Login, Pam
- And extensions
  - Kexec image, Kernel modules, firmware, security policies, eBPF programs, setuid programs
- Troublesome things:
  - Anything run by root, interpreted code, some things read by root....
- Really troublesome things
  - Password file, DNS configuration file.....
- Our Approach:
  - Let IMA measurement/audit list everything that was actually accessed, sign it and see what happens

What Is the Threat Model?

- We want to defend TCB files against “remote root” attacks
  - A remote attacker who has somehow gained root

- Assumption:
  - We have secure boot and IMA-appraise-digital-signature
  - The respective public key(s) are anchored in uefi, and thus not modifiable by remote attack.
  - Anyone with physical access to the machine is trusted.



Linux Integrity Subsystem: original 'ima' Measurement and Attestation list

- 'ima' filedata-hash digest limited to 20 bytes and filenames to 255 characters

```
PCR     template-hash  template  filedata-hash  pathname

10 88da93c09647269545a6471d86baea9e2fa9603f ima
a218e393729e8ae866f9d377da08ef16e97beab8 /usr/lib/systemd/systemd

10 e8e39d9cb0db6842028a1cab18b838d3e89d0209 ima
d9decd04bf4932026a4687b642f2fb871a9dc776 /usr/lib64/ld-2.16.so

10 babcdc3f576c949591cc4a30e92a19317dc4b65a ima
028afcc7efdc253bb69cb82bc5dbbc2b1da2652c /etc/ld.so.cache

10 68549deba6003eab25d4befa2075b18a028bc9a1 ima
df2ad0965c21853874a23189f5cd76f015e348f4 /usr/lib64/libselinux.so.1
```

PCR 10 is signed by the TPM as proof that the software maintained list has not been tampered

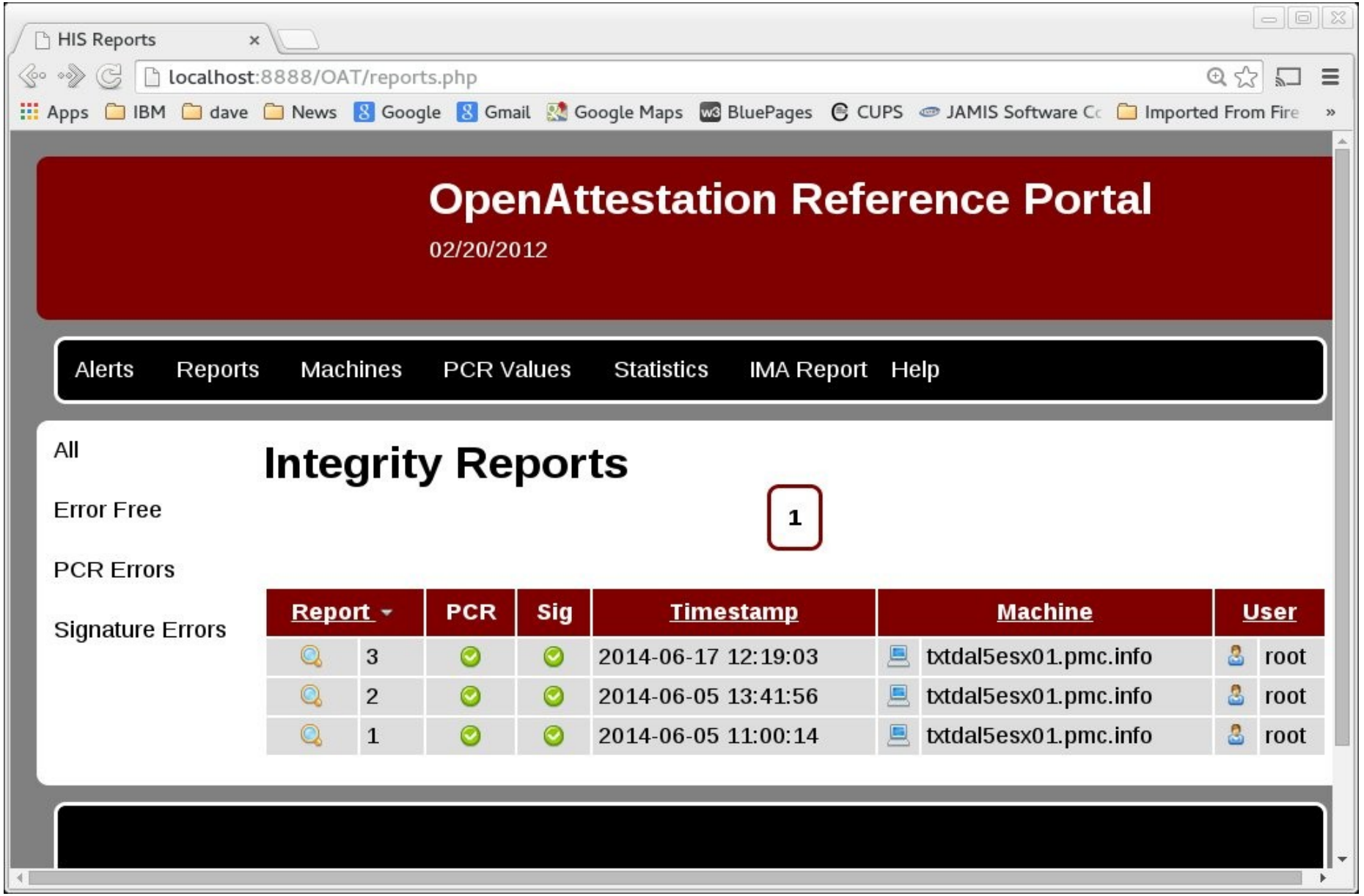
Linux Integrity Subsystem: Appraisal

- Including file data signature in the measurement list provides file provenance and makes attestation validation **much** more scalable

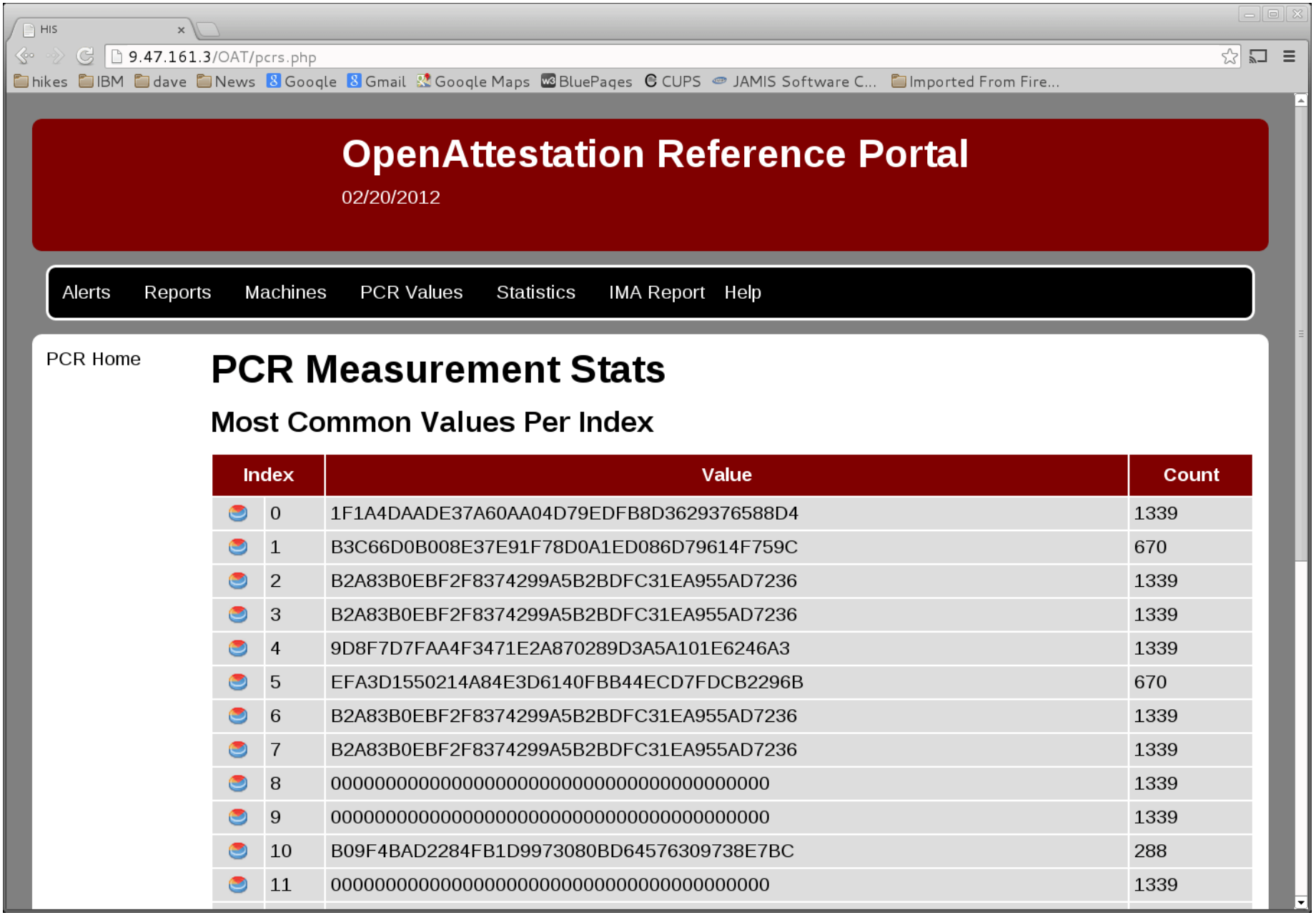
```
          Fixed fields                    builtin/user-defined format
  --------------------------     ------------------------------------
PCR     template-hash  template  filedata-hash  filename  file-signature
                                     (d-ng)       (n-ng)    (sig)
10 c2f79235e5a3c6541a7dd5268be223c0a71f4b90 ima-sig
7cf25b8e6156800a493fcdc3c2f1f493b190f994
/usr/lib64/libc-2.16.so
030202afab4511008051615301fda9e67a7dc29a8bd262deb613cf4ceafe71b75d64be8cc73d5431
2330127a0d03975185c28d0a0bfc344d5ae319b603a49c3de8f9f8ccc7f7ec753173294eb24a84b6
a38c5c3db44cb19ba2e28c4423e769f1150b1a07b514d1f29105ccbebad8f84589d0a7bdf9caab1b
30b54c26db1e2f33a59d5cad956a5a4144

10 654a1fbebf81242699fba49859e49e039079fcbb ima-sig
e3359301b44dceeab530217caaf466361bf2dc5b /usr/bin/ibm-notes8
030202eb11e7300808732fa83c7187f09c8c70b7d3c9a8ab37b73a6e5e0b9c1cbbb285ebd8e6226
f13aa4c61ff9b16604c84bf13458f564cf3f919807ae70f4521702ba8ab62b864224526be2bfdf2e
1eb7302ce8e1bb6a2f1349c219de832c0f0c596fdf9f76674a515548a88e456c3b2a952ca0f4ab09
7e4f27249fe198bfd141f92059a98c29b4
```

Basic OpenAttestation (OAT) Reports

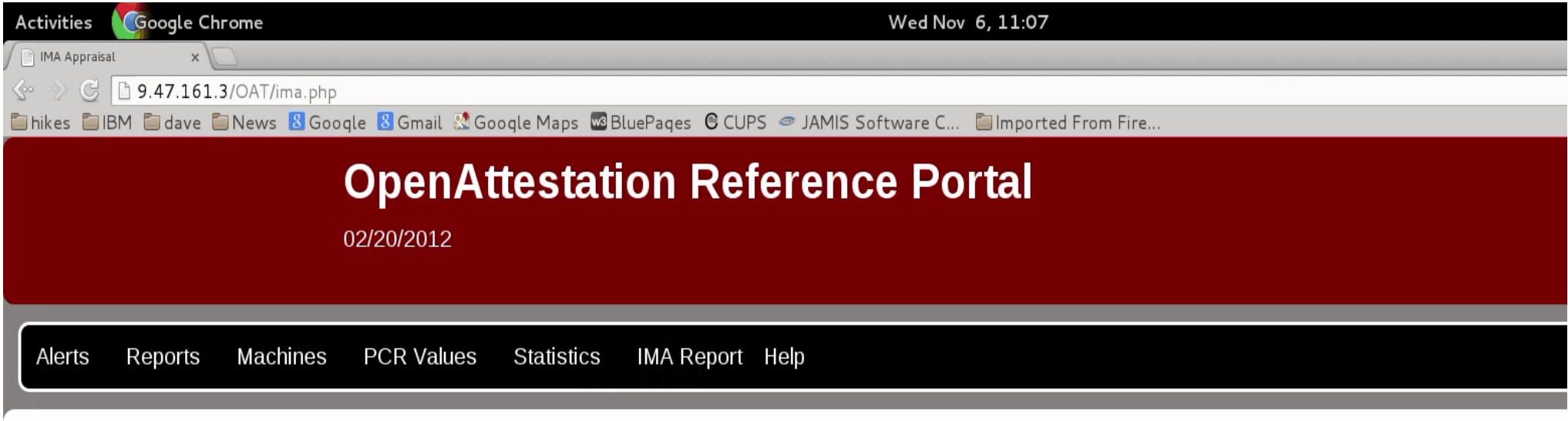


OAT PCR Values



Index		Value	Count
	0	1F1A4DAADE37A60AA04D79EDFB8D3629376588D4	1339
	1	B3C66D0B008E37E91F78D0A1ED086D79614F759C	670
	2	B2A83B0EBF2F8374299A5B2BDFC31EA955AD7236	1339
	3	B2A83B0EBF2F8374299A5B2BDFC31EA955AD7236	1339
	4	9D8F7D7FAA4F3471E2A870289D3A5A101E6246A3	1339
	5	EFA3D1550214A84E3D6140FBB44ECD7FDCB2296B	670
	6	B2A83B0EBF2F8374299A5B2BDFC31EA955AD7236	1339
	7	B2A83B0EBF2F8374299A5B2BDFC31EA955AD7236	1339
	8	0000000000000000000000000000000000000000	1339
	9	0000000000000000000000000000000000000000	1339
	10	B09F4BAD2284FB1D9973080BD64576309738E7BC	288
	11	0000000000000000000000000000000000000000	1339

OAT Extension with IMA-Appraisal Summary



Verifying IMA Measurement List:
Calculated PCRAggr:D4 32 E7 3E 3D 51 97 97 88 D9 7C FA 69 37 A3 66 5D 2B A8 C2
Actual PCR-10: D4 32 E7 3E 3D 51 97 97 88 D9 7C FA 69 37 A3 66 5D 2B A8 C2

Total Measured Files: 693
/etc/keys/ibm_signed.der 2
/etc/keys/ubuntu_signed.der 689
/etc/keys/fedora_signed.der 0
/etc/keys/google_signed.der 0
unsigned 1

Signature Errors:

Unsigned: 1
691 010 9096e6e6ed015b9788f27eae367a8713f4778401 ima-sig sha1:86603190e1fa93af60
8bbcd96e658118b6a5391f /bin/trojan_nosig

Invalid signature: 1
692 010 accdbf520b9593a191b0764a146d53626901bfca ima-sig sha1:50ce91a7efc7cfa874
8eca9ff794e8f132be6f83 /bin/trojan_badsig 0302024eaa24370080c2e056146f42dd736786
47ed52d92bf053a43f8acf77ead57f9a676af128ef72bb09752896e156d2a1332ff0059d9e2025da

IMA Policy Language

```
rule format: action [condition ...]

action: measure | dont_measure | appraise | dont_appraise | audit
condition:= base | lsm  [option]
base:   [[func=] [mask=] [fsmagic=] [fsuuid=] [uid=]
 [fowner]]
lsm:[[subj_user=] [subj_role=] [subj_type=]
 [obj_user=] [obj_role=] [obj_type=]]
option: [[appraise_type=]] [permit_directio]

base:   func:= [BPRM_CHECK][MMAP_CHECK][FILE_CHECK][MODULE_CHECK]
mask:= [MAY_READ] [MAY_WRITE] [MAY_APPEND] [MAY_EXEC]
fsmagic:= hex value
fsuuid:= file system UUID (e.g 8bcbe394-4f13-4144-be8e-5aa9ea2ce2f6)
uid:= decimal value
fowner:=decimal value
lsm:    are LSM specific
option: appraise_type:= [imasig]
```

Why These Are Not Sufficient

- Immutable Elf, .so, #!/.... are easy – just sign everything

- Problem 1: What did root just read?
  - Interpreted files that are read into interpreter (bash '.', python “import”....)
  - Versus files that are just read, like log files....
  - Config files – things like DNS server config
- Problem 2: IMA Limitations
  - The IMA policy language cannot differentiate between these files
  - IMA does not have enough information about how the files are being used
  - IMA bypass possible for mutable (hashed) files
    - Unlink, rename, setfattr

Proposed Additions

- Combination of:
  - IMA-Appraisal-Digital-Signatures
  - BSD Immutable Files
- Add new **IMA Policy Action** - “lock”
- Add new kernel command line option – ima_appraise= fix | lock
- Add new IMA securityfs file - “lock”
- By default, IMA enforces “lock” actions
- Locking can be turned on immediately with ima_appraise=lock, or on with the “lock” securityfs file, after any boot time maintenance
  - Unlocking implies boot to singleuser mode, and only signed code will execute
  - This trusted code will lock when done, and before going to multiuser.

IMA Locked Mode:

- By Policy, blocks:
  - Unlink
  - Rename
  - Setfattr
  - Rmdir
- Files and directories
  - Protects “trusted paths” (/bin, /usr/bin, /etc, ...)

Security Analysis

- All Immutable files in the TCB are signed, appraised, and locked
  - Signatures are verified by keys rooted in secure boot.
  - Remote root attacker cannot sign files, as private keys are not on filesystem, and changing local_ca requires physical presence
- All mutable files in the TCB are hashed, appraised, and locked
  - Cannot be changed by remote root attacker, since they are locked before multiuser or network services are started.

Demo

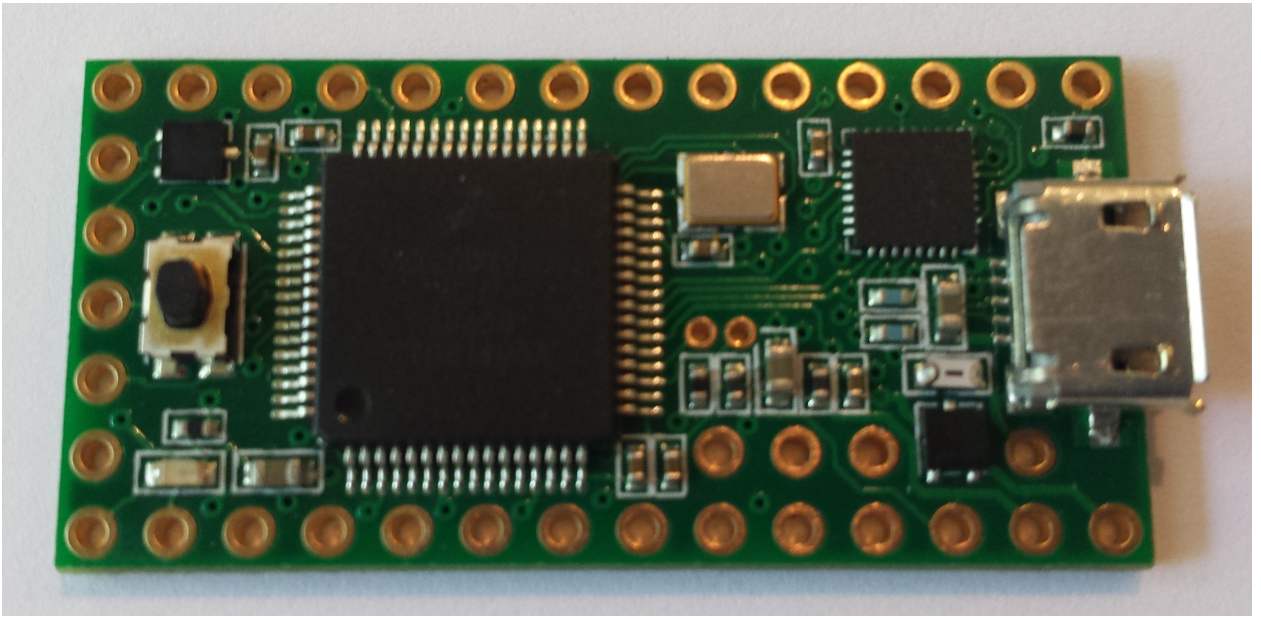
- Basic OAT
- OAT + IMA-appraisal
- Locking
  - Example attack: DNS reconfiguration attack

Bonus: Teensy Token

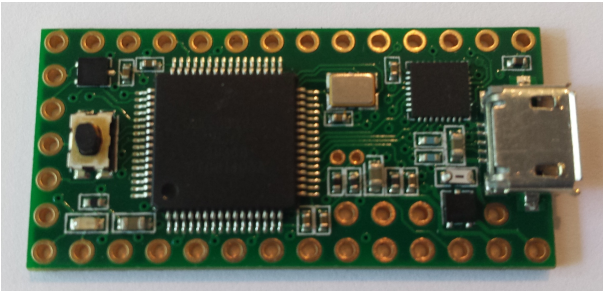
- USB Security Token:
  - Functions
    - TPM for IMA attestation
      - TPM_EXTEND
      - TPM_READPCR
      - TPM_QUOTE
    - TPM for Signing files Locally
    - TPM for ssh remote login
  - Features
    - Private keys generated on token, never leave the token
    - Physical presence (button) required to reprogram.
    - No APIs other than TPM subset
    - Request LED, approval touch pad

Bonus: Teensy Token

Teensy 3.1
- 32 bit ARM
- 256k flash
- 64k RAM
- micro-usb

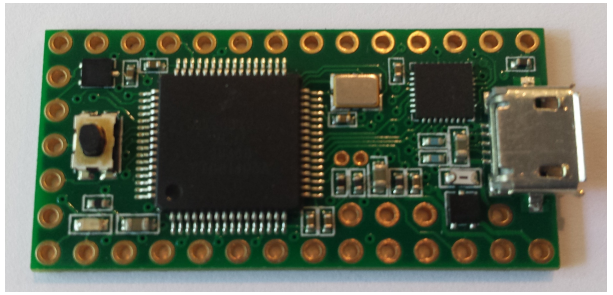


Bonus: Teensy Token - construction



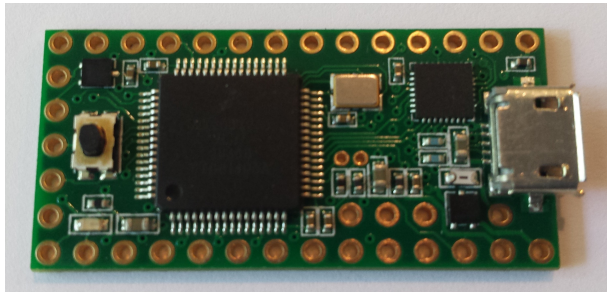
Teensy 3.1

Bonus: Teensy Token - construction



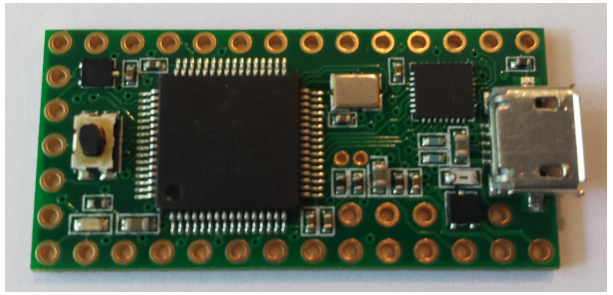
Teensy + case (old sandisk)

Bonus: Teensy Token - construction



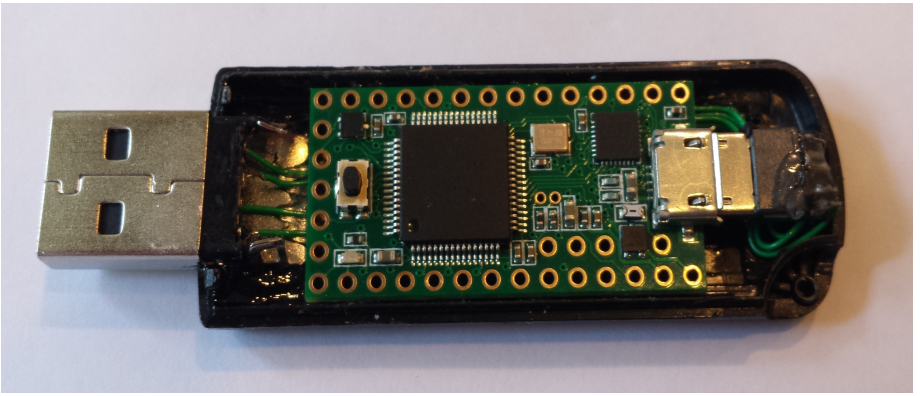
Teensy + case (old sandisk) + usb cable

Bonus: Teensy Token - construction



Teensy + case (old sandisk) + usb cable + magic

Bonus: Teensy Token - construction



Teensy Status

- TCG compliant TPM functions complete:
  - TPM_EXTEND
  - TPM_READPCR
  - TPM_QUOTE
  - Speed over USB comparable to physical TPM (~4 seconds for 4K files)
- RSA and SHA from polar SSL
- USB RawHID support from teensy C library
- TPM (1.2) lower half

- Working on approval to release the code

Summary

- New IMA “locked” mode to protect mutable files
  - Now can protect all TCB files

- Teensy Token
  - IMA anchor
  - IMA file signer with protected private key
  - SSH authenticator with protected private key