ProvUSB: Block-level Provenance-Based Data Protection for USB Storage Devices

Dave Tian, Kevin Butler

Adam Bates

Raju Rangaswami

CCS'16, Vienna, Austria
October 25 2016

Portable Media: A Partial History

IronKey

2005

- User Authentication
- Encrypted Storage
- FIPS Certification

Portable Media: A Partial History

IronKey

2005

- User Authentication
- Encrypted Storage
- FIPS Certification

CENTCOM networks infected with worm via USB Drive.

2008

CENTCOM

Portable Media: A Partial History

IronKey

2005

- User Authentication
- Encrypted Storage
- FIPS Certification

2008

CENTCOM networks infected with worm via USB Drive.

CENTCOM

Private Manning

2010

Classified Data exfiltrated on burnt "Lady Gaga" CD.

Portable Media: A Partial History

IronKey

2005

- User Authentication
- Encrypted Storage
- FIPS Certification

2008

CENTCOM networks infected with worm via USB Drive.

CENTCOM

Private Manning

2010

Classified Data exfiltrated on burnt "Lady Gaga" CD.

2012

USB malware used to jump "sneaker net" in act of cyberwarfare.

StuxNet

Portable Media: A Partial History

IronKey

2005

- User Authentication
- Encrypted Storage
- FIPS Certification

CENTCOM networks infected with worm via USB Drive.

2008

CENTCOM

Private Manning

2010

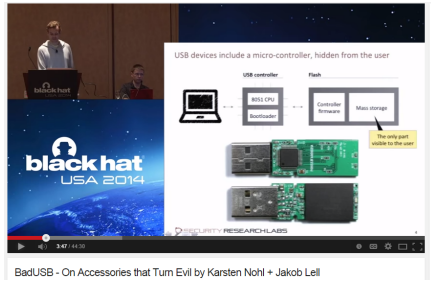
Classified Data exfiltrated on burnt "Lady Gaga" CD.

USB malware used to jump "sneaker net" in act of cyberwarfare.

2012

StuxNet

BadUSB



2014

Firmware-based USB Malware used to compromise machines.

Portable Media: A Partial History

Early smart USB devices tried to prevent data from falling into "wrong hands"…

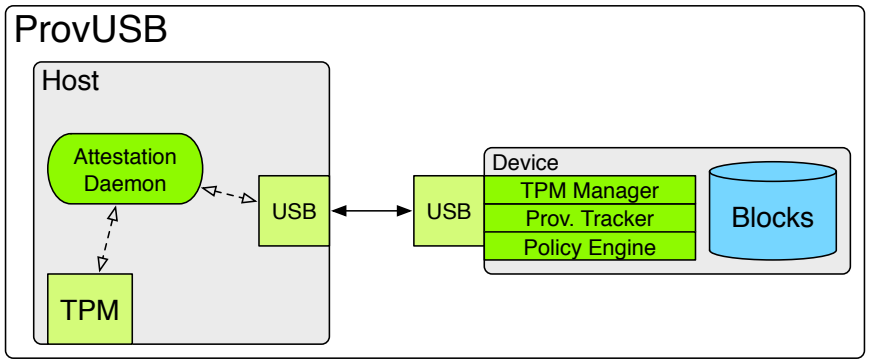
… but can we improve USB security when a smart device is in the "right hands"?

Improving USB Device Security

Current state-of-the-art devices do not address…

- Device Forensics:
  - *Which authorized users could have leaked this data?*
  - *How did the intruder reach our isolated network?*
  - *Which machines are infected?*
- Integrity Assurance:
  - *How do I prevent malware carried on USB within our employee network from reaching our isolated networks?*

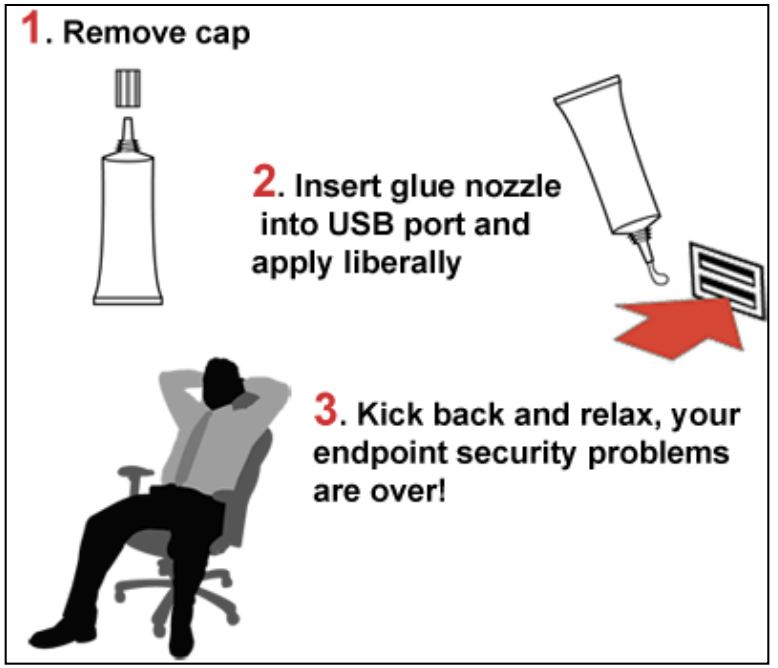
Introducing ProvUSB



- **Host Identification over USB** leverages TPM remote attestation to determine the identity of connected host.
- **Provenance-Based Data Forensics:** produces complete descriptions of all host interactions with device.
- **Integrity Assurance** prevents low integrity data from reaching high-value hosts.

Assumptions & Deployment

- Managed enterprise environments where USB is heavily regulated.
- USB Devices are checked out from Enterprise Security Office.
- All hosts equipped with TPM
- Administrator partitions machines between ***Low Integrity*** (e.g., employee workstations) and ***High Integrity*** (e.g., classified terminal).



ProvUSB is designed as a viable alternative for this kind of system administrator.

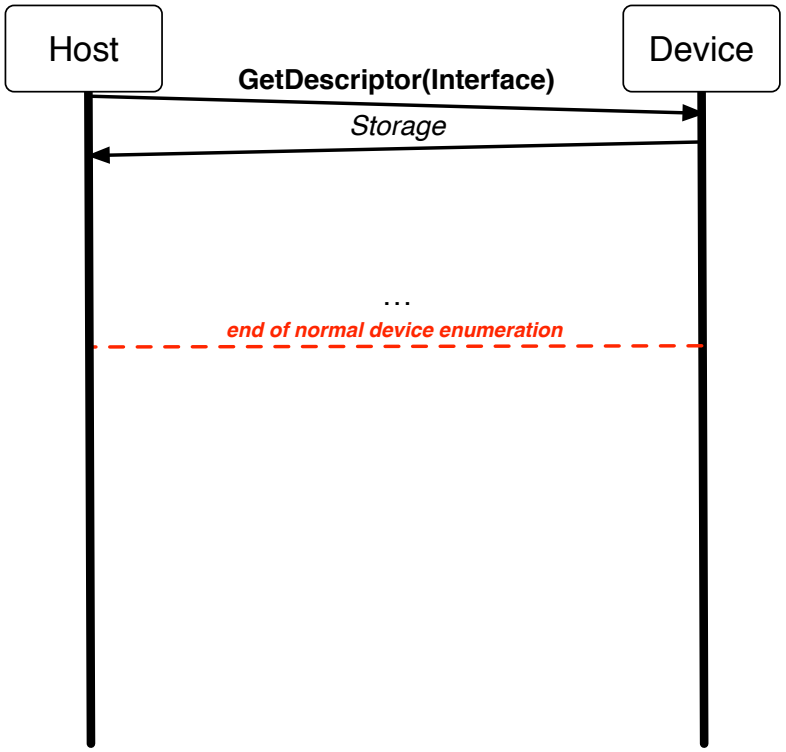
ProvUSB Security Goals

1. **Small TCB:** Trust no software on the host.
2. **Forensic Validity:** Produce complete descriptions of device usage. Loss of any forensic information must be detectable by the administrator.
3. **Tamperproof:** Host must not be able to disrupt monitoring mechanisms on the device.
4. **Integrity Assurance:** Device must prevent **LI** data from flowing to **HI** host.

Step #1: Identifying Host Machine

How can we identify the hosts to which our device is connecting??

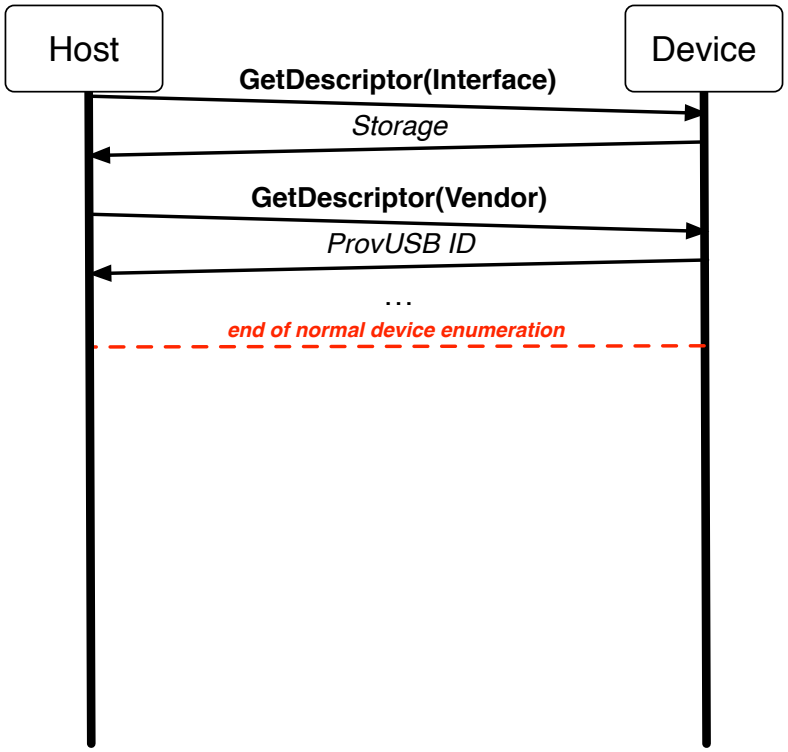
- *TPM attestations over USB to authenticate host prior to mounting storage (Butler, ACSAC'10).*



Step #1: Identifying Host Machine

How can we identify the hosts to which our device is connecting??

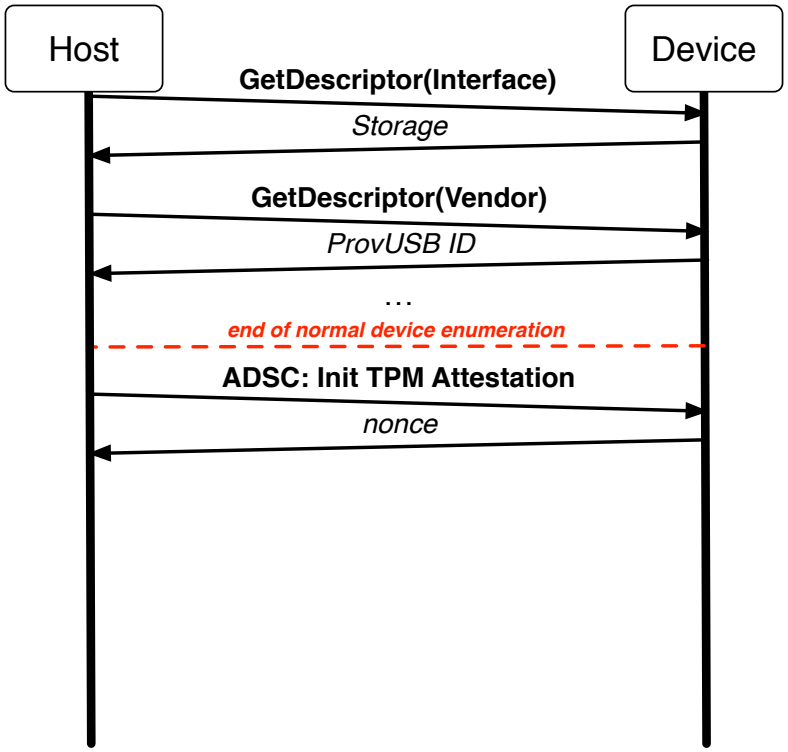
- *TPM attestations over USB to authenticate host prior to mounting storage (Butler, ACSAC'10).*



Step #1: Identifying Host Machine

How can we identify the hosts to which our device is connecting??

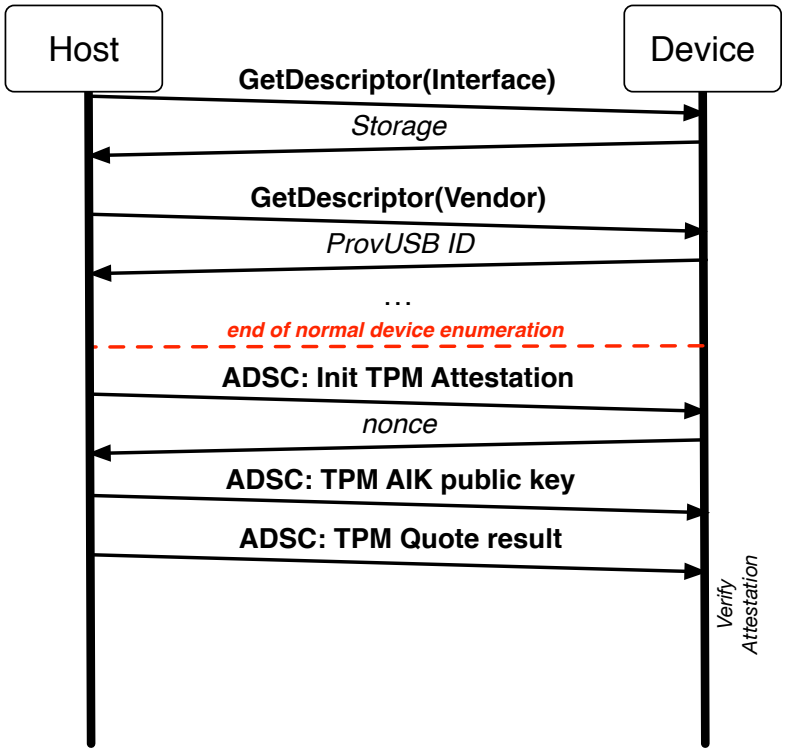
- *TPM attestations over USB to authenticate host prior to mounting storage (Butler, ACSAC'10).*



Step #1: Identifying Host Machine

How can we identify the hosts to which our device is connecting??

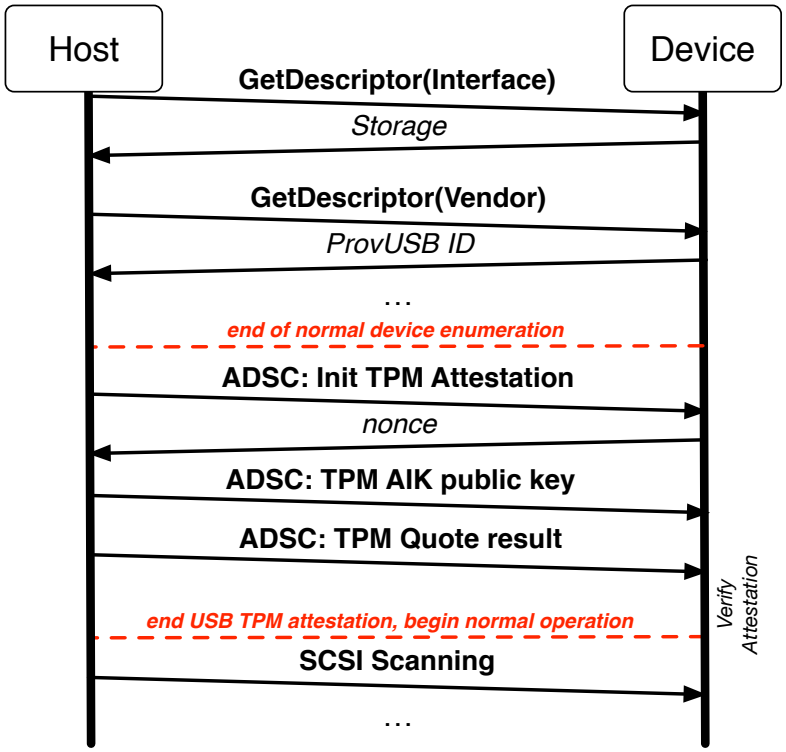
- *TPM attestations over USB to authenticate host prior to mounting storage (Butler, ACSAC'10).*



Step #1: Identifying Host Machine

How can we identify the hosts to which our device is connecting??

- *TPM attestations over USB to authenticate host prior to mounting storage (Butler, ACSAC'10).*



Step #2: Provenance Tracking

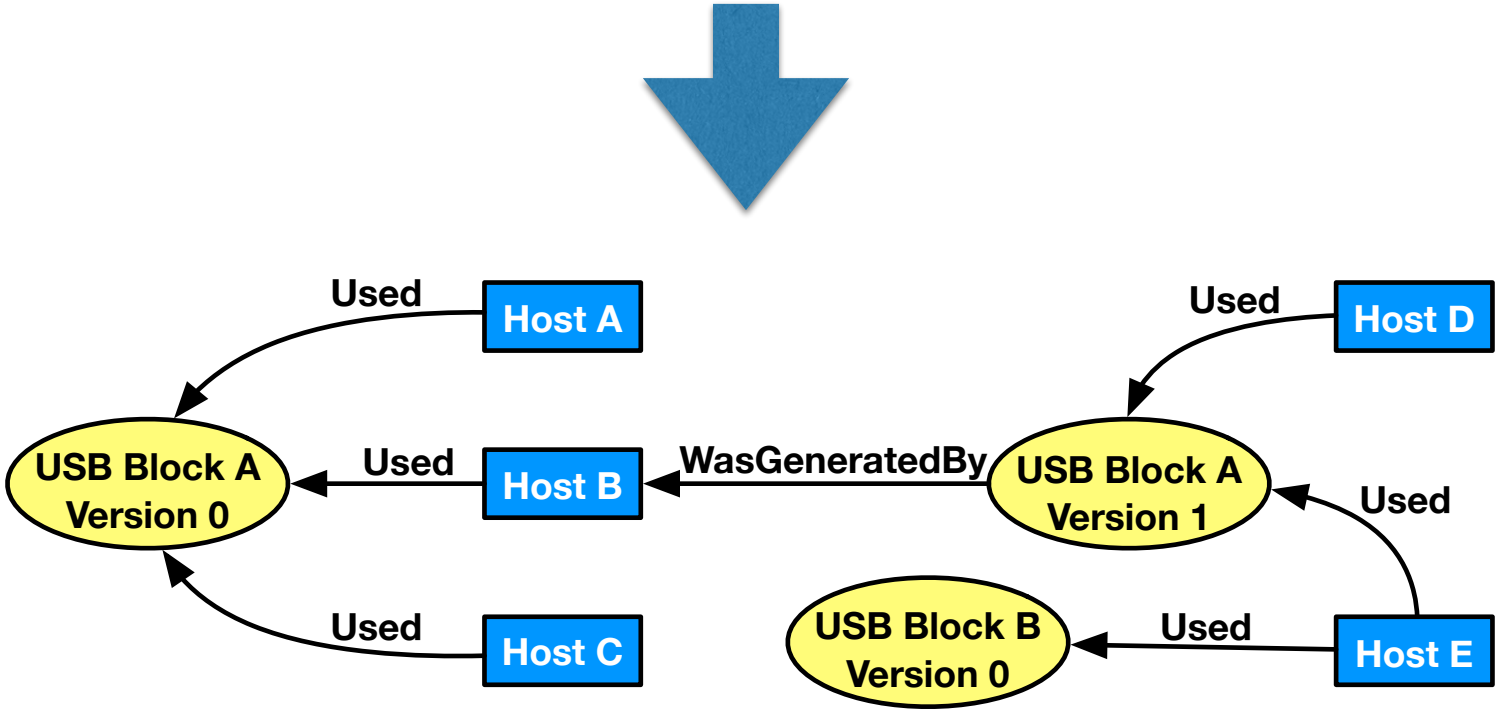
What data abstraction layer should we use when tracking device usage?

- Filesystem Layer would create format dependencies, limiting usefulness.
- Instead, Block Layer provides universality and finer-grained tracking!
- *After collection*, translating from blocks to filenames in a given FS format is fairly straightforward.

Step #2: Provenance Tracking

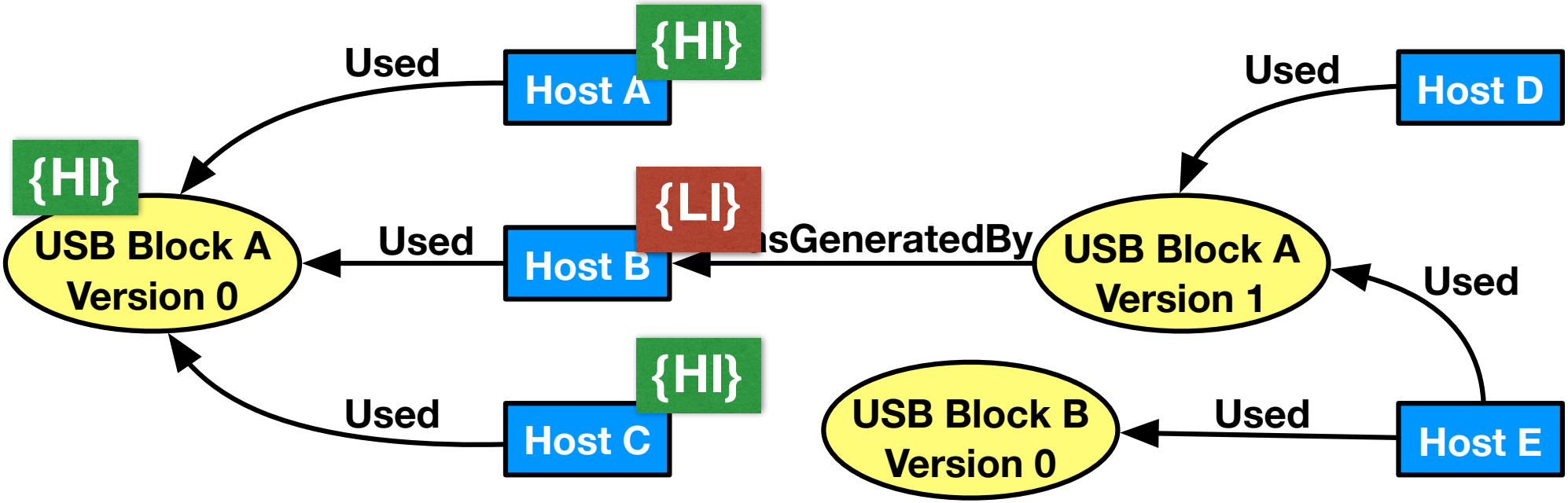
Raw I/O access events are processed into provenance graphs:

```
[  224.557800] provusb: trusted-dev do_read, lba[776], file_offset[397312], amount[4096]
[  224.570709] provusb: trusted-dev do_read, lba[784], file_offset[401408], amount[4096]
[  224.583343] provusb: trusted-dev do_read, lba[792], file_offset[405504], amount[4096]
[  224.596069] provusb: trusted-dev do_read, lba[800], file_offset[409600], amount[4096]
[  224.608978] provusb: trusted-dev do_read, lba[808], file_offset[413696], amount[4096]
[  224.621734] provusb: trusted-dev do_read, lba[816], file_offset[417792], amount[4096]
```



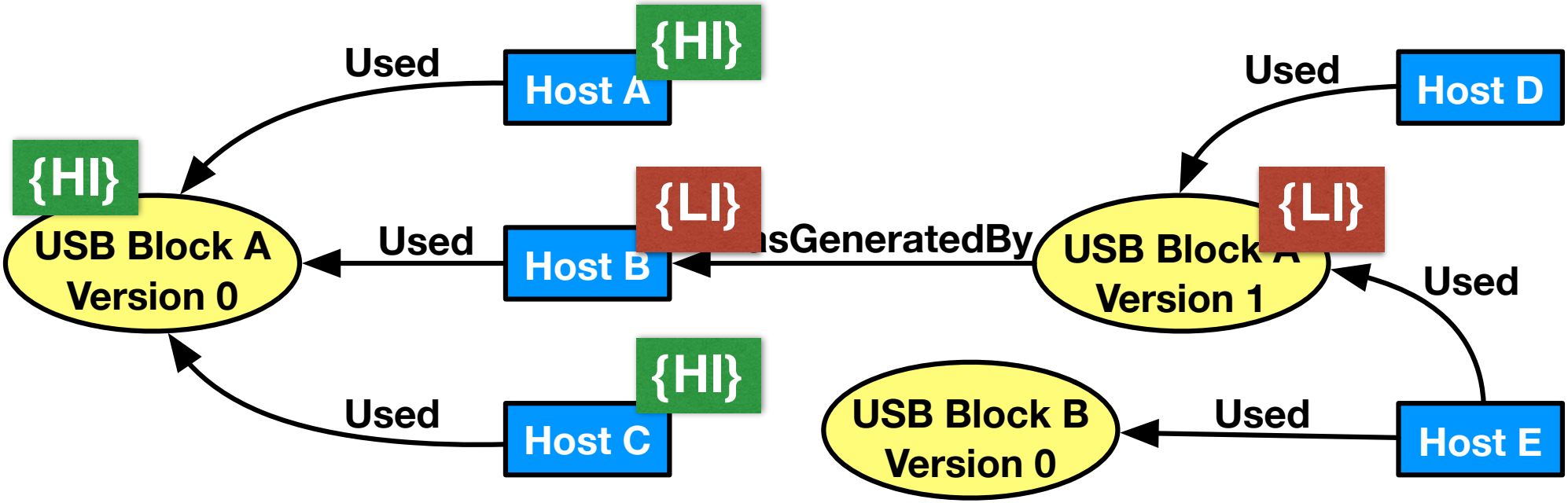
Step #3: Prov-Based Access Control

- ProvUSB's Integrity Model:
  - Host machine labels are statically assigned
  - Data labels are dynamically inferred from graph:

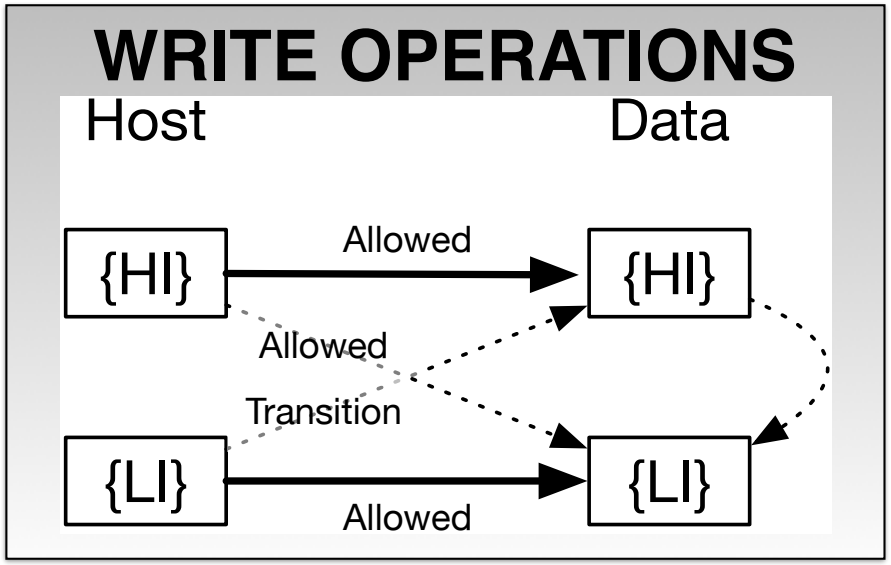


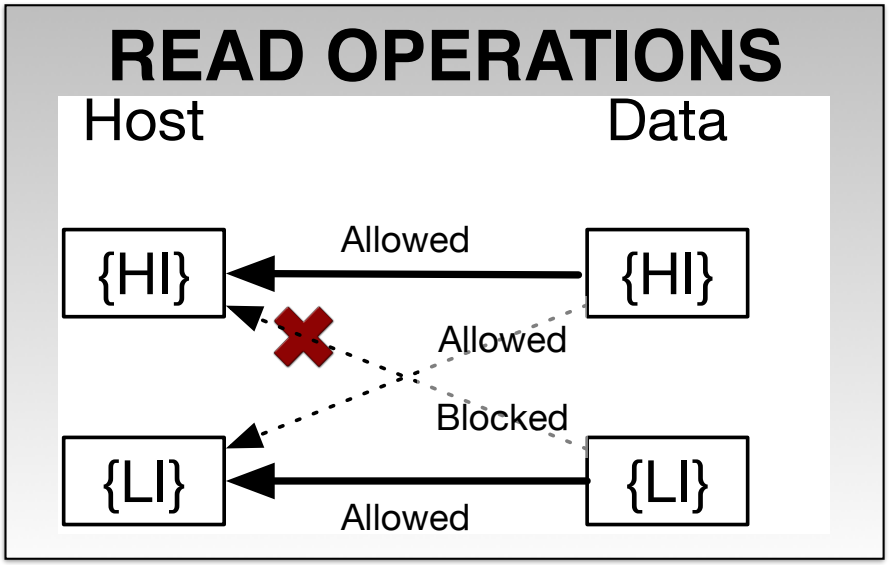
Step #3: Prov-Based Access Control

- ProvUSB's Integrity Model:
  - Host machine labels are statically assigned
  - Data labels are dynamically inferred from graph:



Step #3: Prov-Based Access Control



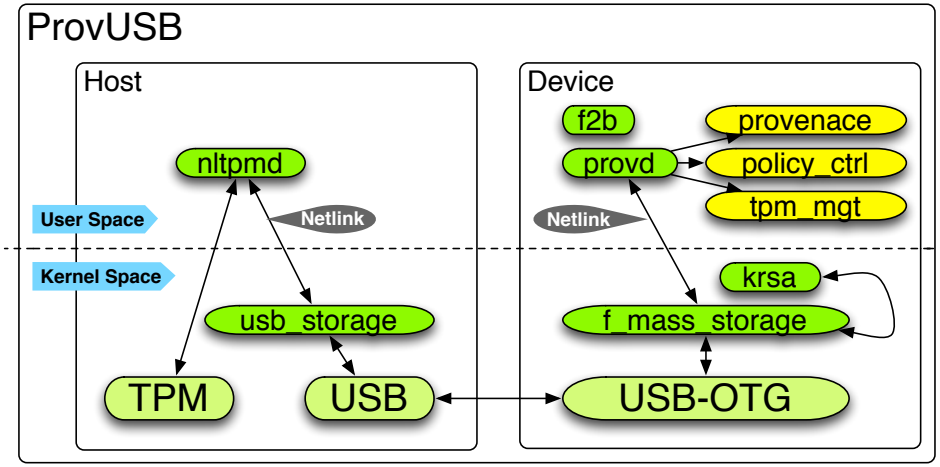


- Write Operations:
  - **Allow all writes**, but downgrade data blocks written to by {LI} hosts
  - **Deny reads** of {LI} data from {HI} hosts

Security Analysis

- **Minimal TCB**: ProvUSB authenticates the host via TPM attestation before permitting the host access to the storage partition.
- **Forensic Validity**: Device is periodically returned to Security Office for provenance extraction. Failure to return triggers incident response.
- **Tamperproof**: ProvUSB logic is not accessible from the host machine. However, production device would require tamper-resistant hardware.
- **Integrity Assurance**: From forensic validity, it follows from prior work that a correct provenance-based integrity model can be enforced.

ProvUSB Implementation



Implemented on Gumstix COM using Yocto, USB OTG.

Other Features:

- Filtering Optimization skips over redundant provenance events to improve storage overhead.
- f2b Utility performs translation between blocks and FAT16 files.

Overhead: Enumeration

ProvUSB imposes high overhead on device enumeration…
but this is a one-time cost per session.

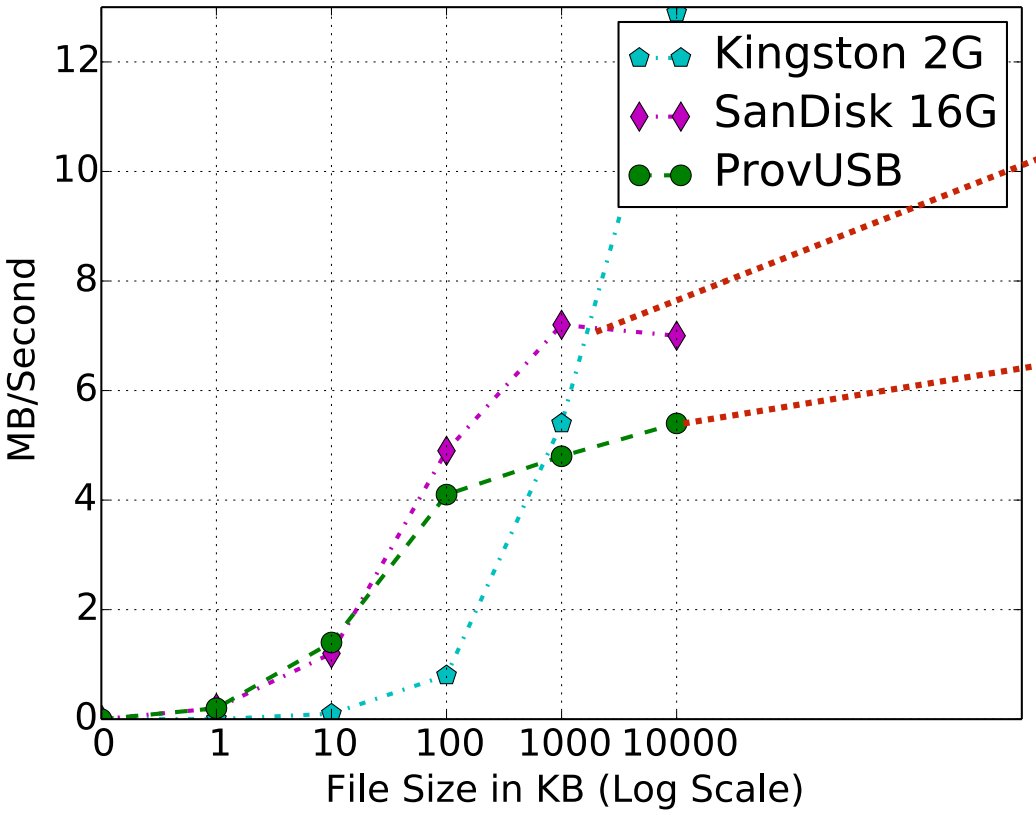
Action	Time
"Dumb" USB Device Enumeration	122 - 343 ms
GumStix Enumeration w/o ProvUSB	65 ms
GumStix Enumeration w/ ProvUSB	850
Host-side TPM Operations	413

~800 ms overhead for connecting device to host.

About 50% of overhead is due to Host-side TPM Delays.

Overhead: Runtime

What is the runtime overhead when using a ProvUSB device?



filebench throughput (MB/s)

"Dumb" USB devices outperform GumStix at large file sizes.

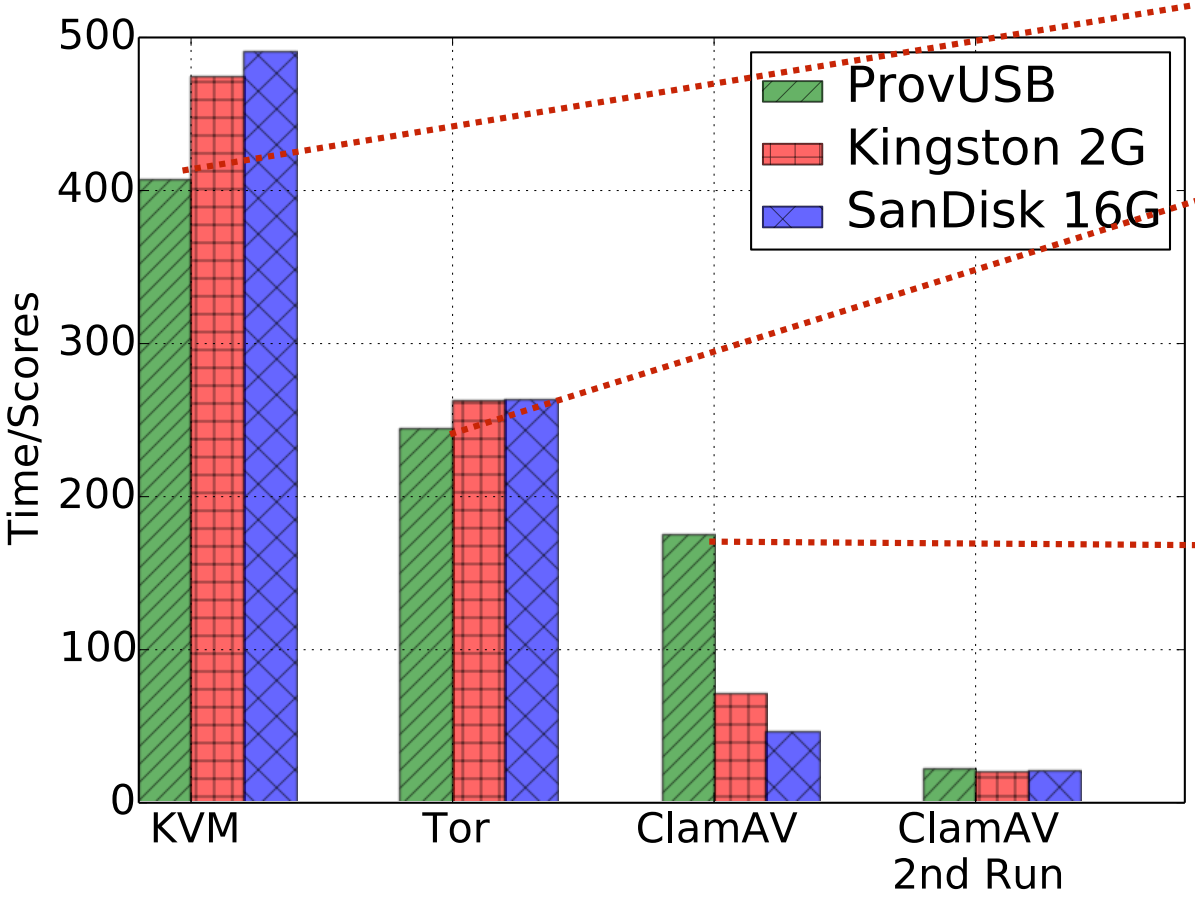
ProvUSB performs comparably to stock Yocto on GumStix.

Device	Large Files
Kingston2G	386.6
SanDisk16G	722.4
ProvUSB	**897.6**

filebench latency (ms)

Overhead: Real Workloads

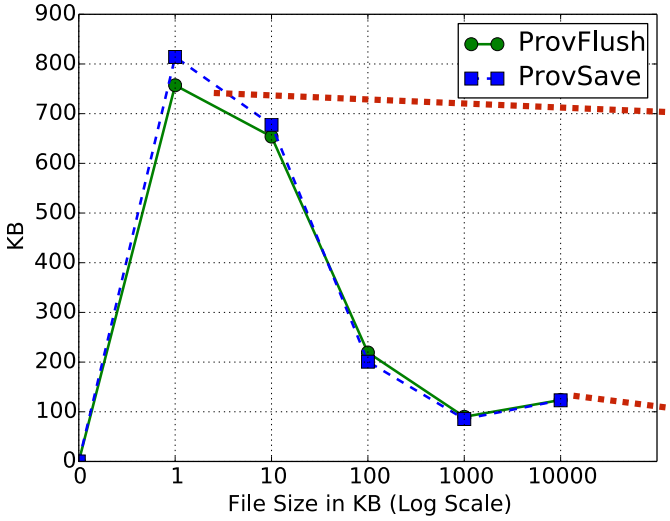
What is the cost of of ProvUSB in daily use?



ProvUSB imposes low runtime overhead when cache hits are frequent.

ProvUSB imposes higher runtime overhead when cache misses are frequent.

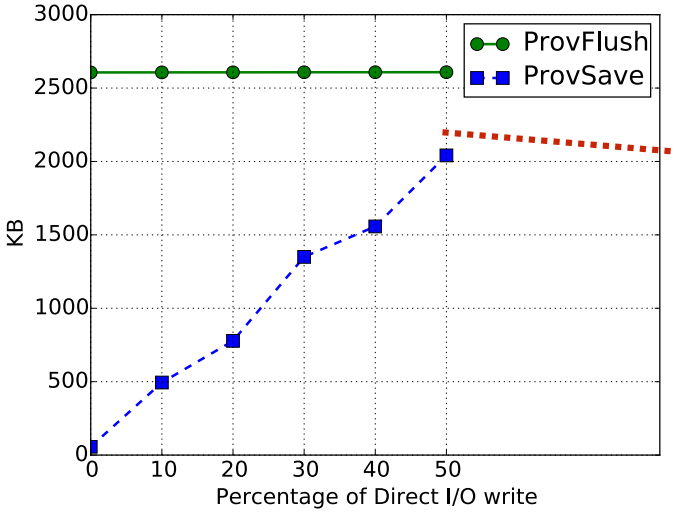
Overhead: Storage



filebench w/ host caching

Generates a manageable 800 KB of provenance during benchmarking.

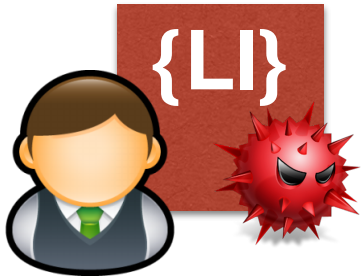
Filtering optimization was not effective due to host side caching.



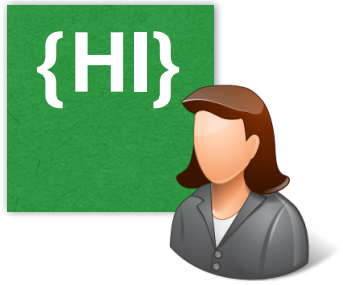
filebench w/ direct I/O

When Direct I/O is used, filtering optimization significantly reduces storage costs when write frequency is low.

Case Studies



Employee Bob's Workstation

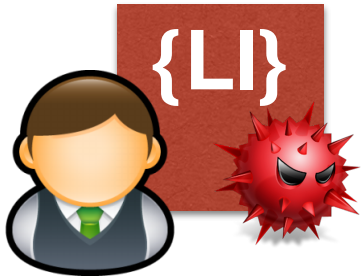


Administrator Alice's Workstation

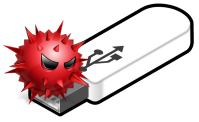
Case #1: Detect Malware Propagation.

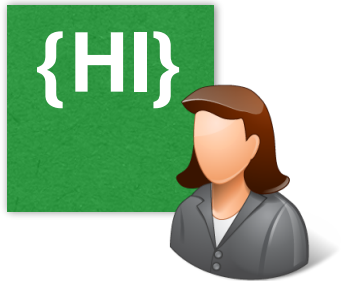
1. Alice discovers an infection in her network enabled by **autorun.inf**.
2. Alice recovers the block number of **autorun.inf** using the **f2b** utility.
3. Alice follows the chain of infections back to Bob's workstation.

Case Studies



Employee Bob's Workstation



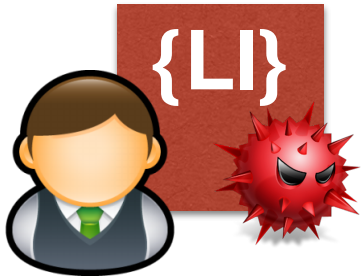


Administrator Alice's Workstation

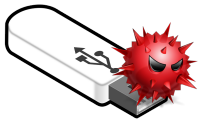
Case #1: Detect Malware Propagation.

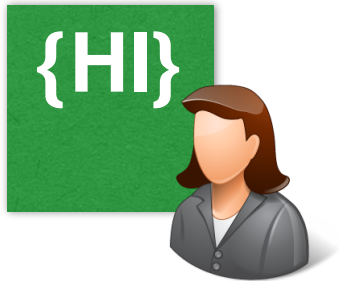
1. Alice discovers an infection in her network enabled by **autorun.inf**.
2. Alice recovers the block number of **autorun.inf** using the **f2b** utility.
3. Alice follows the chain of infections back to Bob's workstation.

Case Studies



Employee Bob's Workstation



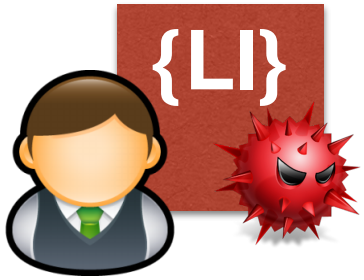


Administrator Alice's Workstation

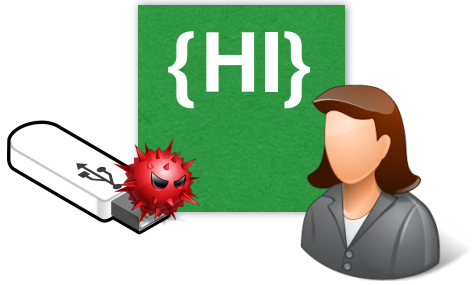
Case #2: Prevent Integrity Violation.

1. Device is connected to Bob's infected workstation.
2. Blocks written by Bob's Workstation are dynamically marked **{LI}**.
3. When device is plugged in at Alice's Workstation, the **{HI}** host is prevented from reading **{LI}** data, preventing **autorun.inf** from executing.

Case Studies



Employee Bob's Workstation



Administrator Alice's Workstation

Case #2: Prevent Integrity Violation.

1. Device is connected to Bob's infected workstation.
2. Blocks written by Bob's Workstation are dynamically marked **{LI}**.
3. When device is plugged in at Alice's Workstation, the **{HI}** host is prevented from reading **{LI}** data, preventing **autorun.inf** from executing.

Conclusion

- Smart USB devices should do more than just protect data confidentiality!

- On-device forensics and supplement network monitoring in order to reason about data movement in large, complex organizations.

- With manageable administrative overhead, Smart USB devices can prevent low integrity data from reaching to critical network end points.

Thanks!

All bugs are introduced by our lead author, Dave Tian (root@davejingtian.org)

Questions?