Take Only What You Need: Leveraging Mandatory Access Control Policy to Reduce Provenance Storage Costs

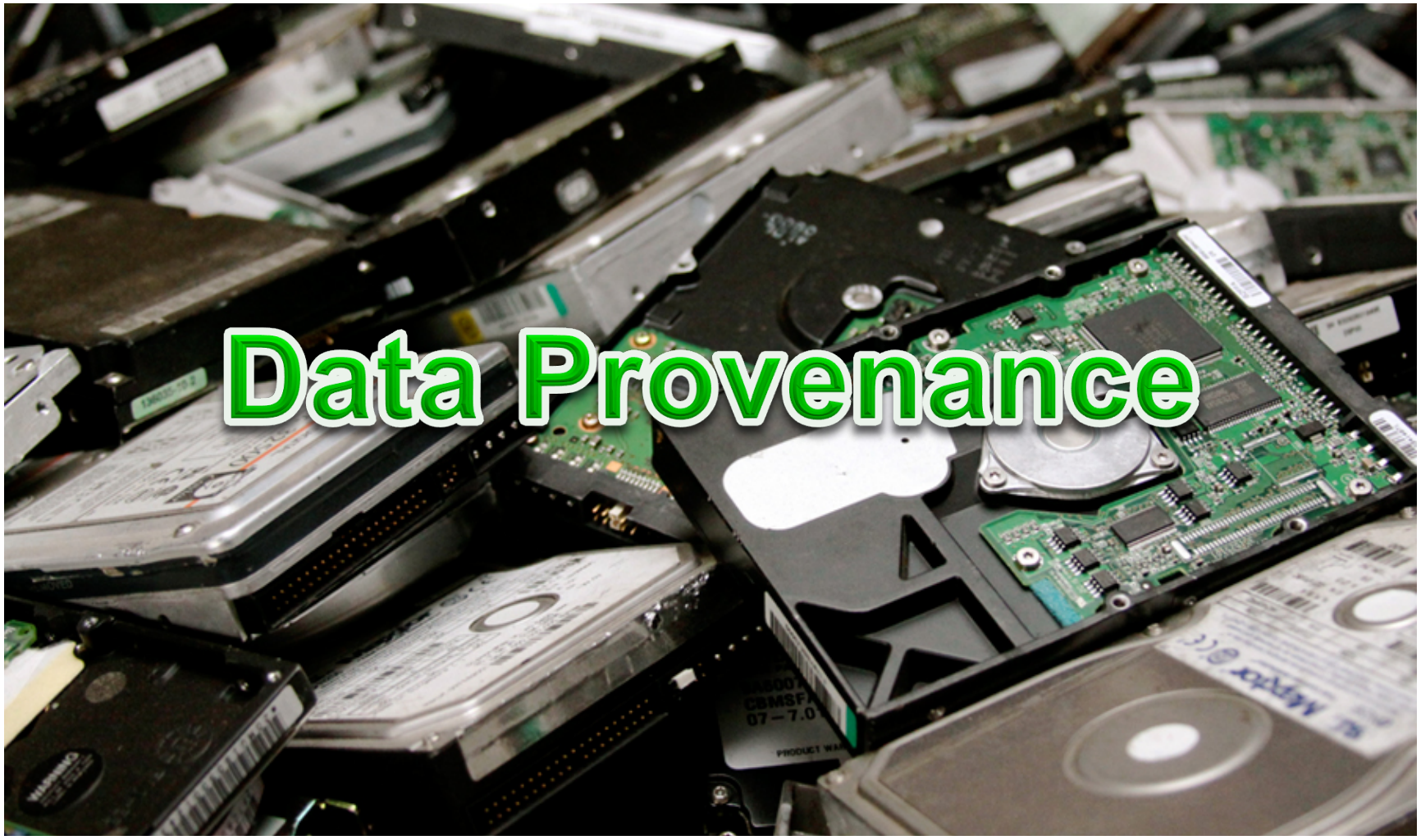
Adam Bates, Kevin R. B. Butler, Thomas Moyer

TaPP '15, Edinburgh, Scotland

9th July, 2015

This work is sponsored by Assistant Secretary of Defense for Research and Engineering under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

Deal-breaker for system provenance?



Provenance-Aware Adversaries

The environment we consider in this work is *not* benign.

Active provenance-aware adversaries attempt to:

- Evade monitoring
- Tamper with prov. logs
- Disable prov. mechanisms

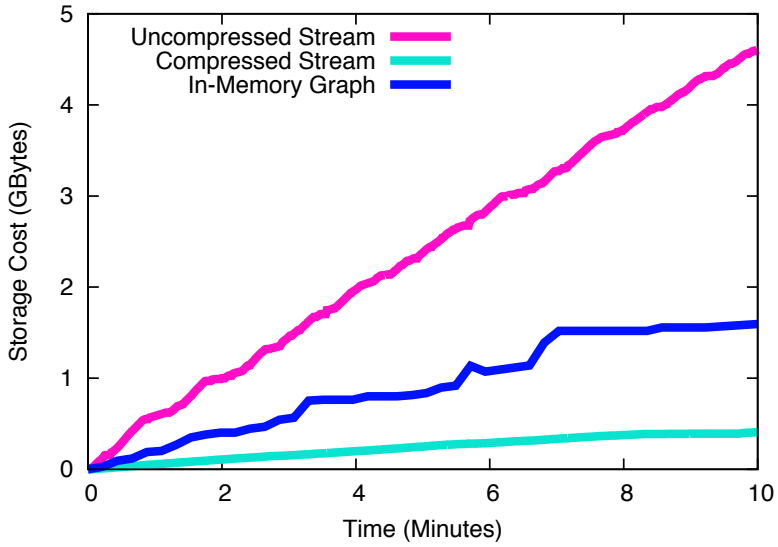
Provenance Monitors:

- Record complete, gapless provenance
- Tamperproof
- Verifiably correct

Deal-breaker for system provenance?

High storage overheads for system layer provenance collection:

- Provenance-aware systems generate GB of metadata on the order of minutes.
- Hi-Fi module generates 4.8 GB during kernel compile.
- After processing, PASS reports similar overheads (~1.5 GB).

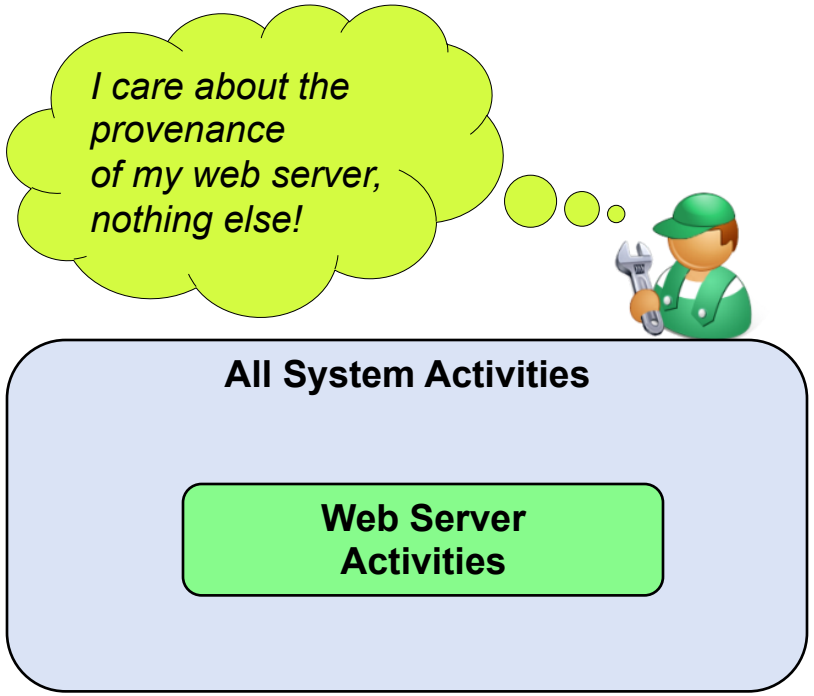


Kernel Compilation benchmark for Hi-Fi using different storage techniques

Deal-breaker for system provenance?

High storage overheads for system layer provenance collection:

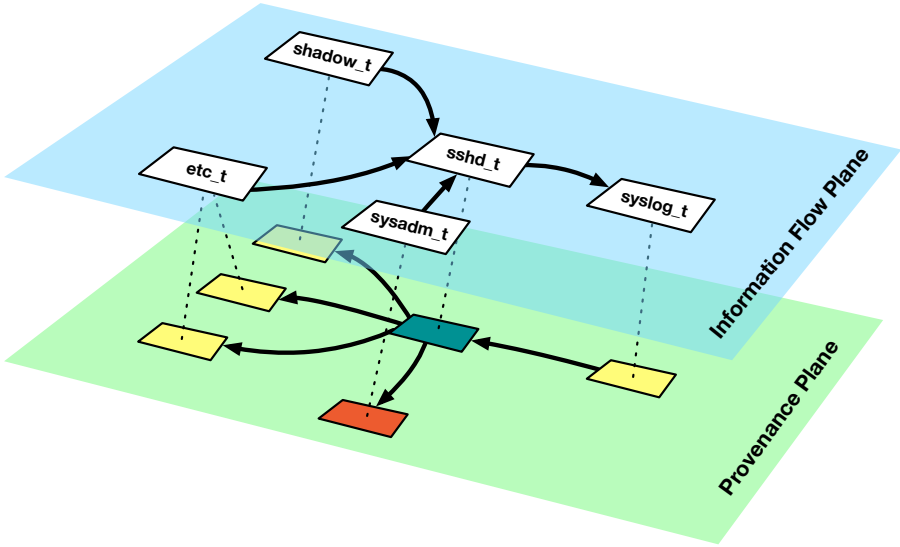
- Worse, a percentage of that provenance is *uninteresting.*
- Provenance compression techniques cannot remove uninteresting data.
- In Discretionary Access Control systems, we cannot guarantee *completeness* without recording *everything*.



“Provenance Walls”

We propose that Mandatory Access Control (MAC) systems can facilitate the performance of selective provenance collection.

- Background
  - Threat Model
  - Storage Overheads
- **Provenance Walls**
  - Provenance & MAC
  - Policy Analysis
- **Future Work**
  - Design & Implementation
  - Challenges
- **Conclusion**



Provenance Walls integrates Provenance with Mandatory Access Control policy.

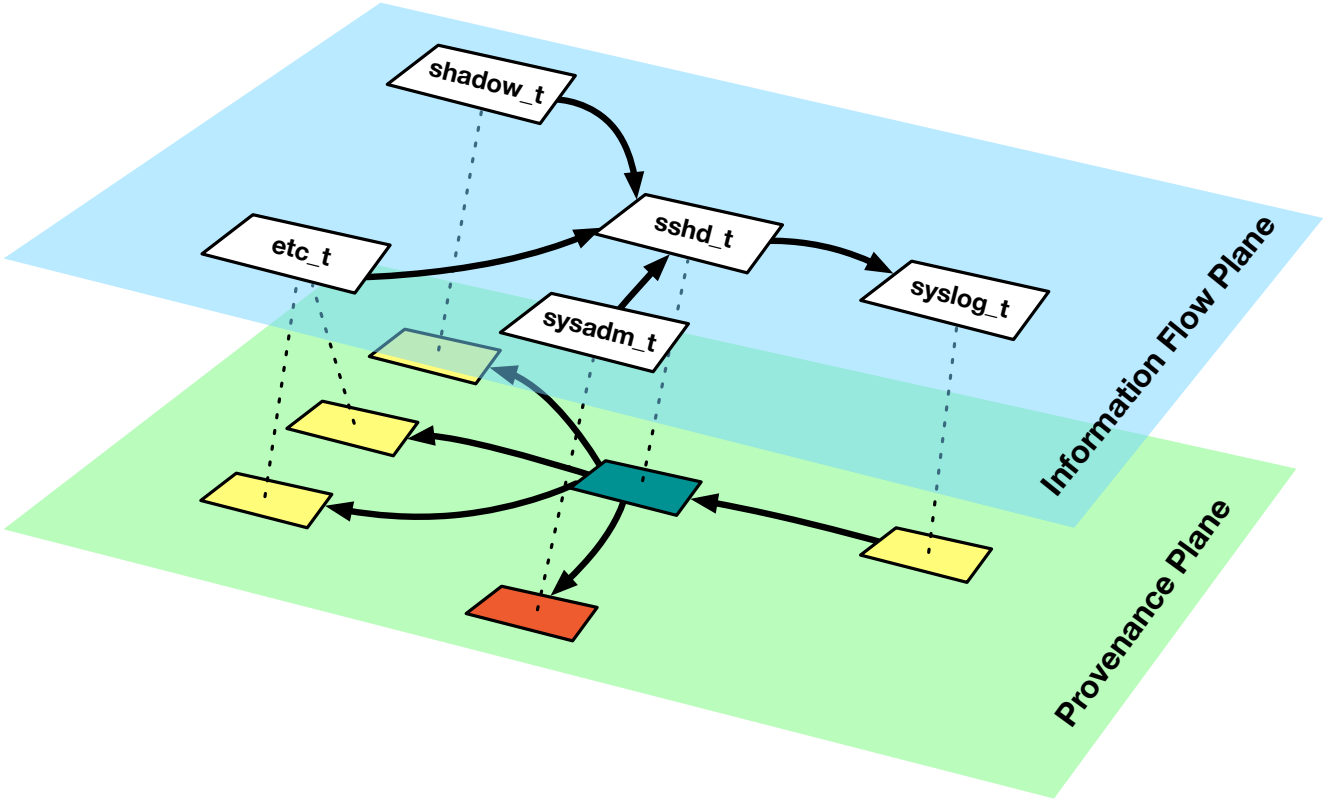
Provenance and Mandatory Access Control

What is the relationship between Provenance and MAC policy?

- **With MAC, we can reason about where data will (not) flow.**
  - MAC answers questions about **possible** future events
- **With Prov., we can reason about where data did (not) flow.**
  - Provenance answers questions about **actual** past events
- **MAC systems assign a security label to every system object.**
  - Objects in MAC namespace map to objects in provenance namespace.

Provenance and Mandatory Access Control

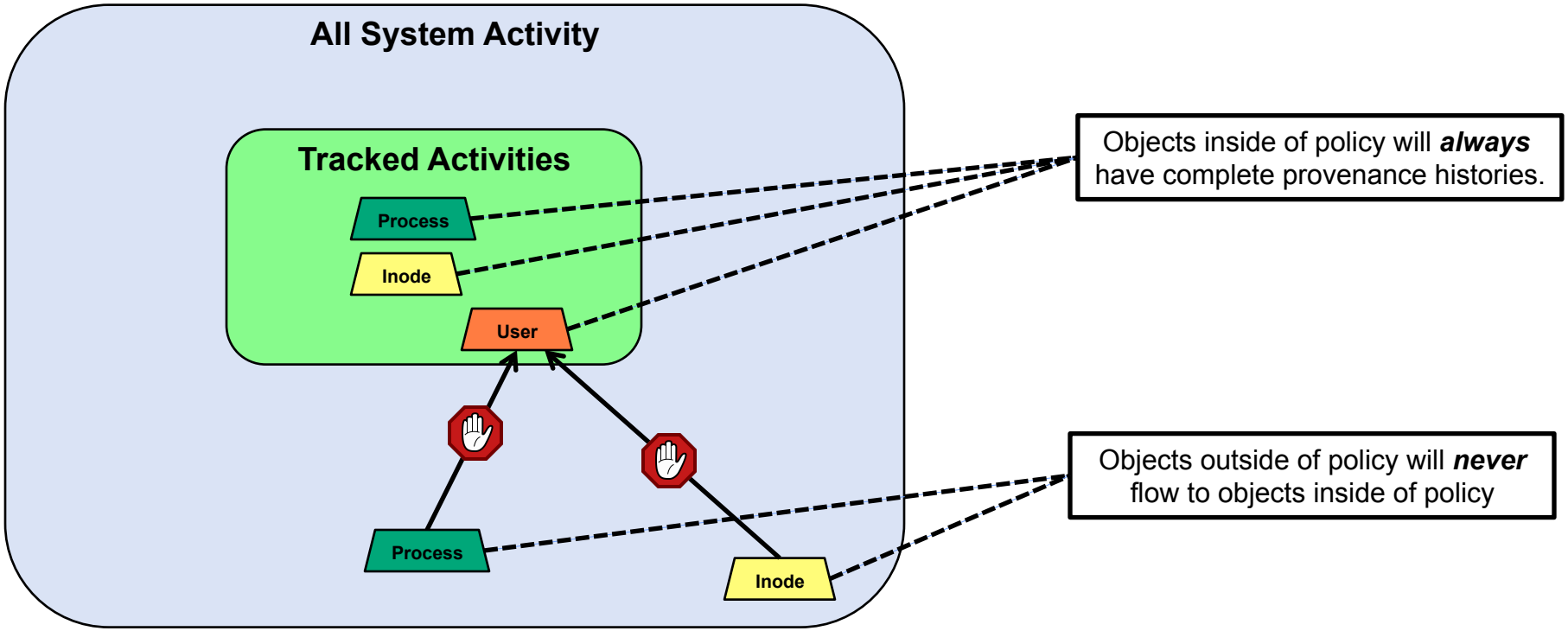
We could define a *provenance policy* in terms of security labels…



… but where does that leave us in terms of assuring *completeness?*

Selective Completeness

Definition: *A provenance sub graph that is complete in its description of a specified system activity… in perpetuity!*



Policy Analysis

Integrity Walls [Vijayakumar et al. 2012]:

- MAC policy analysis tool that identifies an application's attack surfaces.
- Static analysis identifies *executable writers*, *kernel subjects*, and *helper subjects* that form **Minimum Trusted Computing Base (MTCB)**:

```
http_t, http_config_t,
http_user_content_t,
lib_t, http_packet_t
```

- Dynamic analysis is then used to identify adversary-controlled entry points:

```
http_user_content_t,
http_packet_t
```

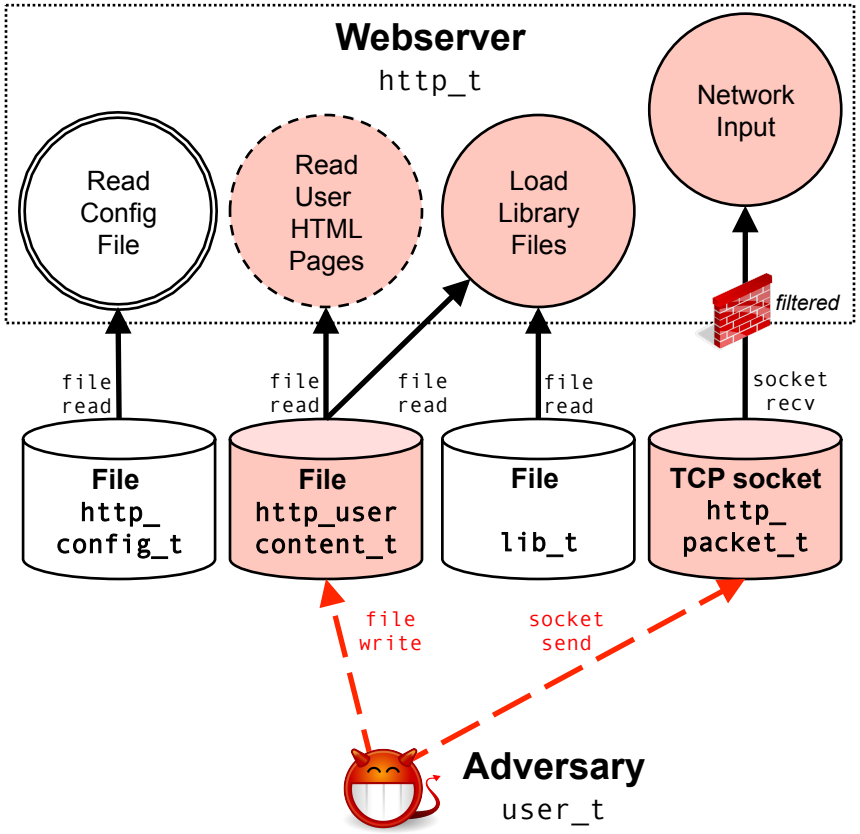


Figure adapted from [Vijayakumar et al. 2012]

Policy Analysis

Integrity Walls [Vijayakumar et al. 2012]:

- Adapt the static analysis tool to create a provenance policy:
- For a given application s, divide the policy P into a set of trusted labels I_s and an untrusted set O_s.
- I_s exhaustively describes the objects that can flow into s.
- I_s is a provenance policy that is *selectively complete* for s.

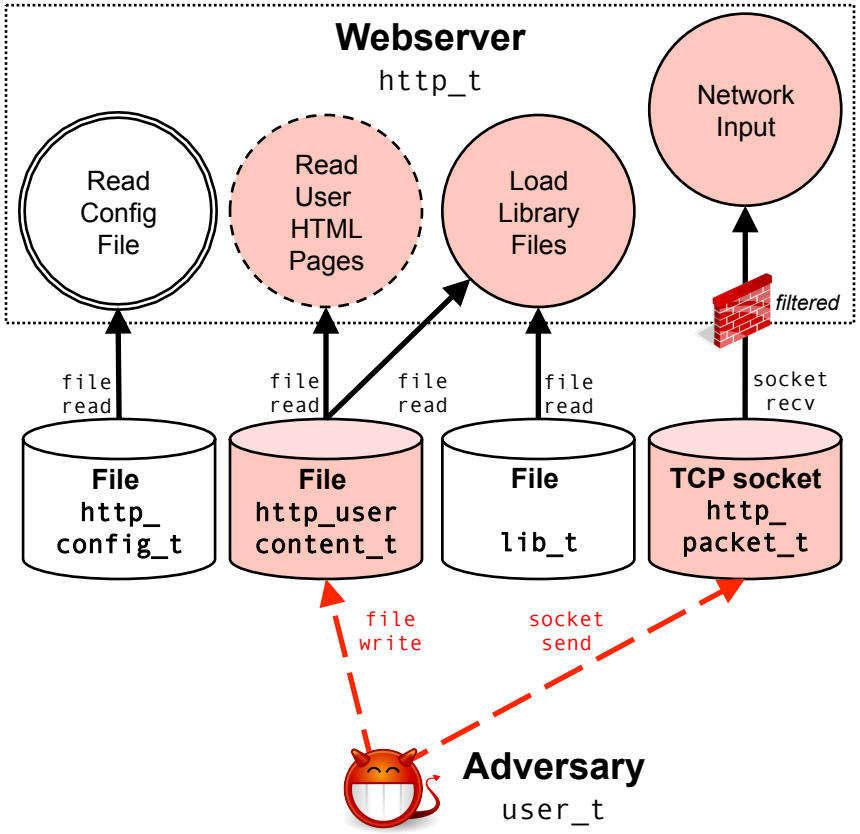
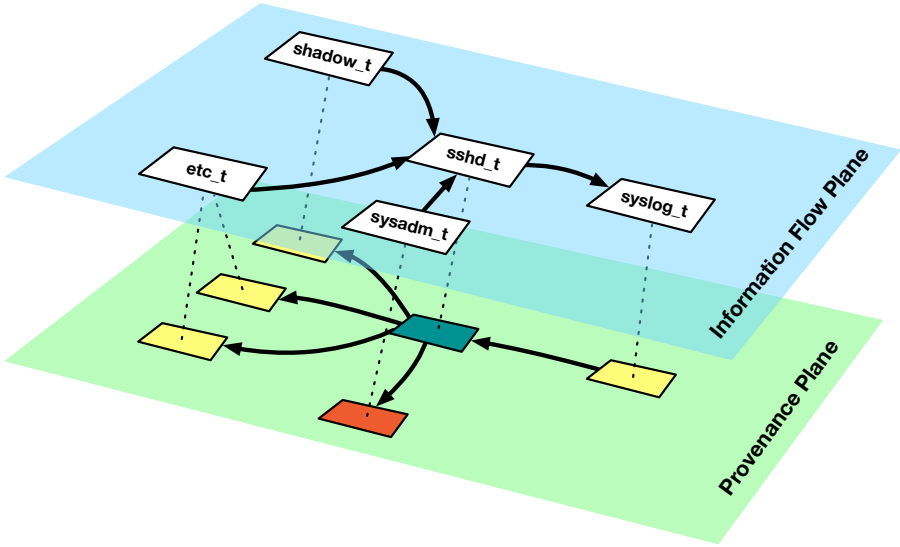


Figure adapted from [Vijayakumar et al. 2012]

“Provenance Walls”

We propose that Mandatory Access Control (MAC) systems can be leveraged to perform policy-based provenance collection.

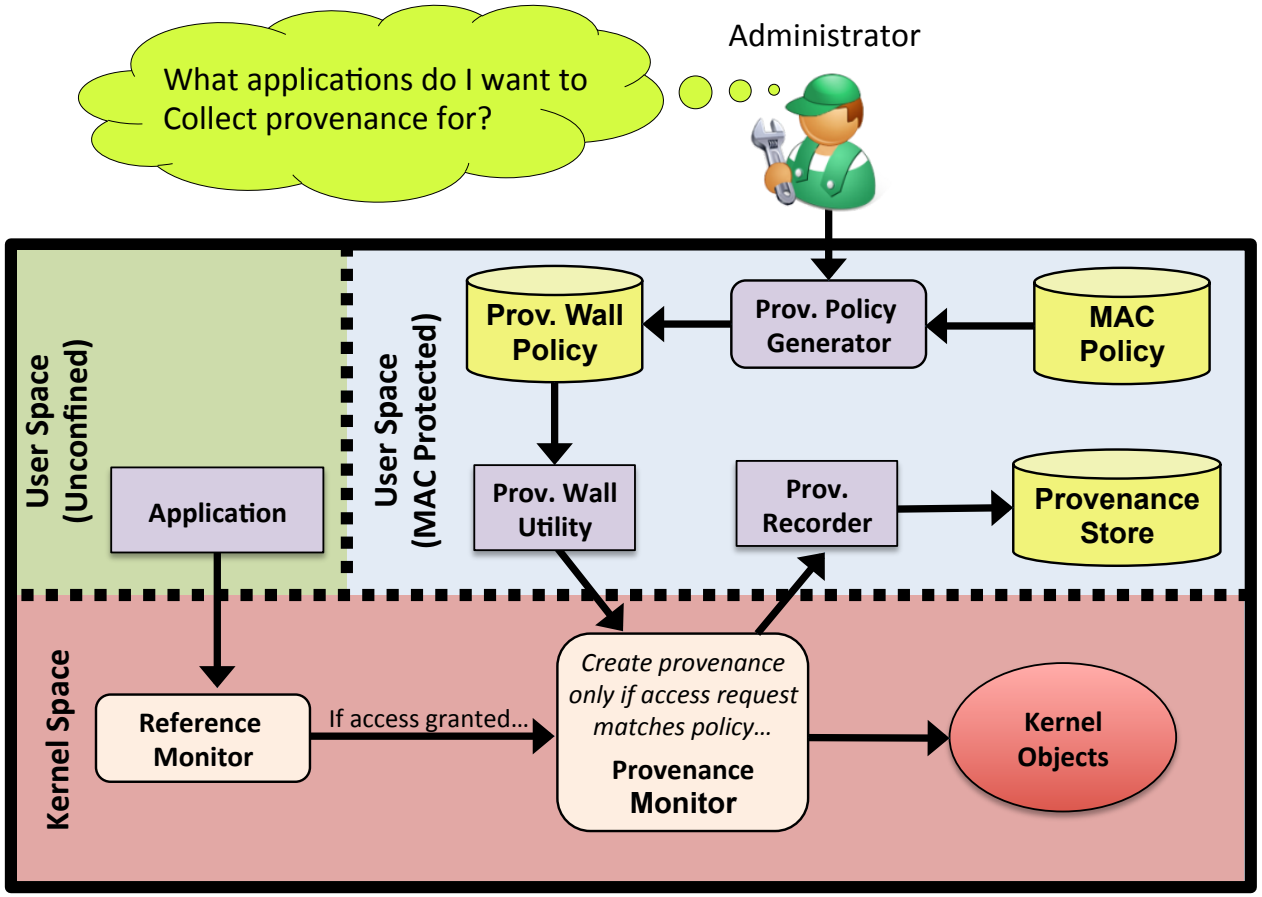
- **Background**
  - Threat Model
  - Storage Overheads
- **Provenance Walls**
  - MAC & Provenance
  - Policy Analysis
- **Future Work**
  - Design & Implementation
  - Challenges
- **Conclusion**



Provenance Walls integrates Provenance with Mandatory Access Control policy.

Provenance Walls Architecture

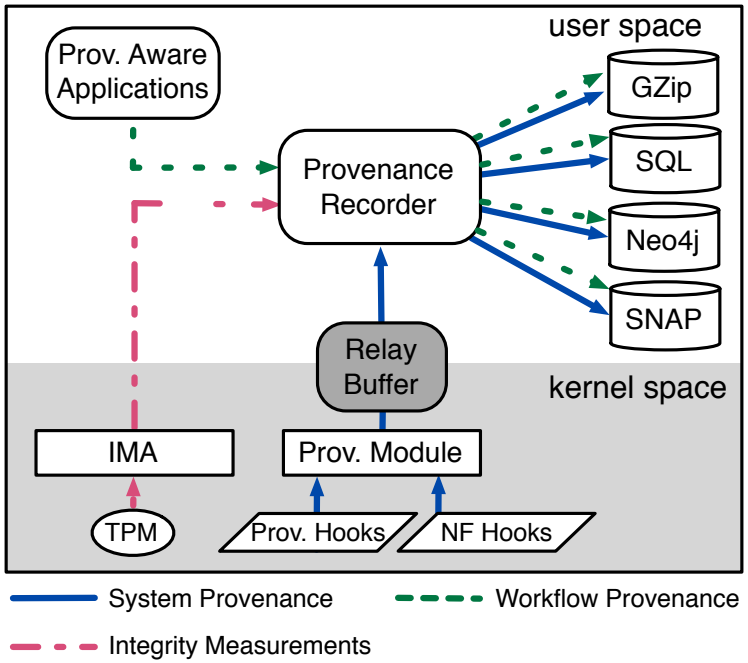
Our architecture for selective provenance recording is shown below:



Linux Provenance Modules

We are developing Provenance Walls using the Linux Provenance Modules (LPM) Framework [Bates et al. 2015]:

- Satisfies “Provenance Monitor Concept”.
- Provenance hooks permit observation of all kernel objects
- Can be simultaneously enabled with SELinux
- We will create a policy-aware version of LPM’s Hi-Fi module [Pohly et al. 2012].



Linux Provenance Modules Architecture

(Highly Contrived) Evaluation

- We made minimal modifications to Hi-Fi to access SELinux security contexts and perform a single policy check.
- **Our Policy:** *"I am not interested in things that happen in user's home directories (*`user_t`*)!!"*
- We then performed kernel compilation test in our home directory:

Module	Provenance Size
Hi-Fi	54 MB
Policy-Aware Hi-Fi	10 MB

Note: Provenance logs are compressed with gzip here.

- **Takeaway:** Savings are domain-specific, and dependent on how many system activities can be pruned.

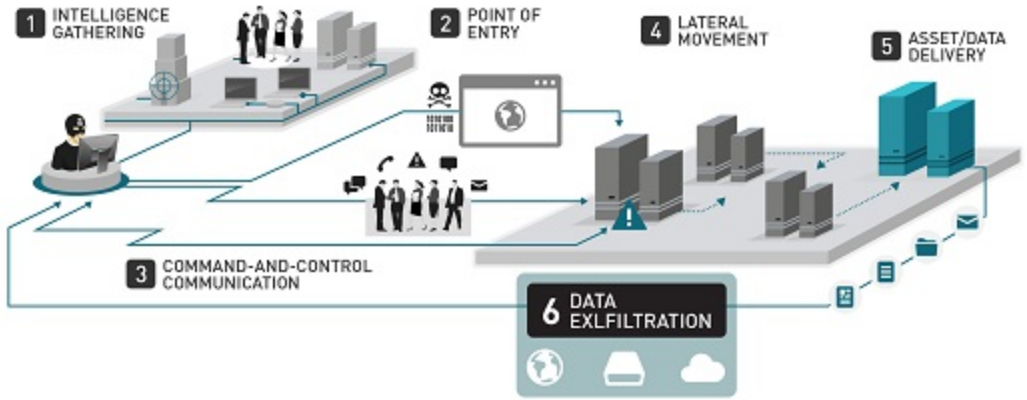
Challenges

- **Policy-Aware Provenance gives rise to new kinds of provenance queries, including:**
  - ***Why is this subgraph missing*?**
    - *Proof that graph omissions are due to correct policy decisions, not error.*
  - ***Where can this data go?***
    - *When reasoning about data provenance, use MAC policy to "look into the future" of system execution.*
  - ***What other data objects are similar to this data object?***
    - *Leverage MAC policy to identify related items by security label*
    - *Objects that are related according to MAC policy may appear unrelated in the provenance graph.*

Challenges

Develop other algorithms for selectively complete policies

- **"Provenance Walls" is great for monitoring a specific, mission-critical application.**
- **Is not adequate for other provenance use cases, such as monitoring data exfiltration:**



Challenges

Will our approach conflict with other reduction techniques?

What to collect? | *What to keep?* | *How to store?*

Tasks:

Specify scope of provenance collection	Reduce dependence explosion, collapse cycles, compact into supernodes, remove attributes.	Provenance-agnostic compression, optimize for storage and/or query.

Related Works:

- **Provenance Walls** [Bates et al. 2015]

- **BEEP** [Lee et al. 2013]
- **Provenance Sketches** [Malik et al. 2010]
- **PASS** [Muniswamy-Reddy et al. 2006]

- **Web / Deduplication** [Xie et al. 2011]
- **Web + Dictionary** [Xie et al. 2012, 2013]

Conclusion

- We are investigating MAC enforcement as a means of reigning in the scope of provenance collection.
- Depending upon the application, the savings are potentially large (82% storage reduction).
- Secure computing deployments not only provide an interesting use case, but also create new opportunities to address open challenges in provenance collection.
- LPM makes it easier to prototype provenance monitors, and simultaneously assures that collection mechanisms are tamper proof and have complete mediation of system activity.

Questions?

Thank you for your time.

Adam Bates

adammbates@ufl.edu

Linux Provenance Modules will be available in August at

http://linuxprovenance.org