# Policy Access Control and Policyrep

# Policy Access Control – What happened

- We've been talking about it for years
  - Where is it already?
  - Prototype on oss.tresys.com
- Where we went wrong the first time
  - Policy server introspected the policy manually
  - Required code in the server for each component
  - Policy format changes required updates
  - Could only work on linked or expanded copy
    - Linked copy had no semantic rules, diffing was hard
    - Expanded copy had no disabled optionals, diffing was incomplete

# Policy Access Control (PAC) – Take 2

- Where we right the second time
- Access control hooks embedded in the libraries
- Server registered callbacks and made avc calls
- Operated on a special expanded copy
  - Which was fully semantic
  - And included disabled optional blocks
- Policy format changes do not require server changes
  - Unless additional components require access control
- Also controls things added via libsemanage

# PAC – The problems

- Not upstream
  - Trying to remedy this
  - It's quite a bit of code, some of it is ugly
- Very, very slow
- Not sure how to address this
- Requires multiple policy expansions
- With all optional blocks included
- Multiple symbol copy passes
- Many lookups in the "previous" policy
- Tons of memory used (don't put on a mobile)

# Policyrep

- Wanted a more maintainable library
  - For policy representation, compilation, etc
  - Which could be a basis for future language development
  - Such as interfaces, tunables, templates, inheritance, etc
- Original prototype in Python
  - Could read refpolicy “language” (interfaces, et al)
  - Currently used as the core of audit2allow
  - Doesn’t handle some things so well
- C++ chosen for a real implementation
  - Fairly large codebase
  - Some dependencies we do not like
  - Interfaces into libsepol are unwieldy (encapsulation)
  - Very slow – probably addressable

# Policyrep (Python implementation)

- Python policyrep has been expanded
- Support for interfaces
- Converts higher level language features into source policy consumable by checkpolicy
- May be able to create policy readable by old (v15) checkpolicy (FMAC)
- Would need to change libsemanage to call it
- Possibly wouldn’t need module format
- Slow

# Where we are at

- Policy access control prototype -> production
  - First step is to get support upstream
  - Even if it isn't turned on by default
- Get people interested
  - Policyrep allows new language features
  - May encourage new developers
  - PAC coupled with policy management
- Refpolicy must support language features
  - Or they will go the way of clone and auditdeny

# What is next?

- Topic of discussion (please give suggestions)
- Continue with C++ implementation?
  - Work on speed issues
  - encapsulation, dependencies
- Do language enhancements w/Python version?
- What to do with module format?
  - Policyrep should obsolete it, backward compat
- What happens to policy access control?

# Questions