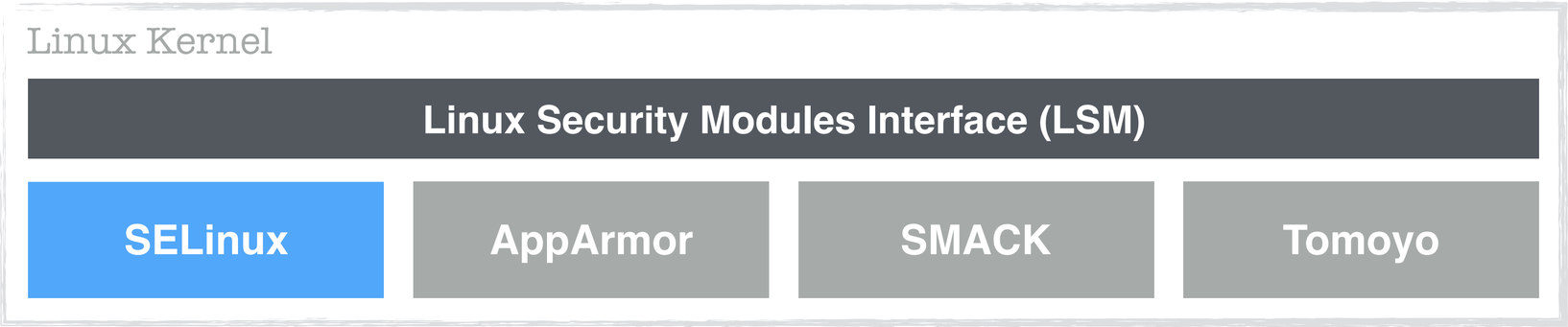
PIXELS CAMPFIRE

SELinux
for regular people

Carlos Rodrigues
Bright Pixel

May 2017

What is SELinux?



- One of several Linux Security Modules*

  …originated from the FLASK** research project
  …part of the *mainline* kernel for the last **14 years**

- Enabled **by default** on Red Hat / CentOS / Fedora

  …also on Android since 4.3 (Jelly Bean)

*kernel.org/doc/Documentation/security/LSM.txt
**www.cs.utah.edu/flux/fluke/html/flask.html

Why Use SELinux?

- Confine services

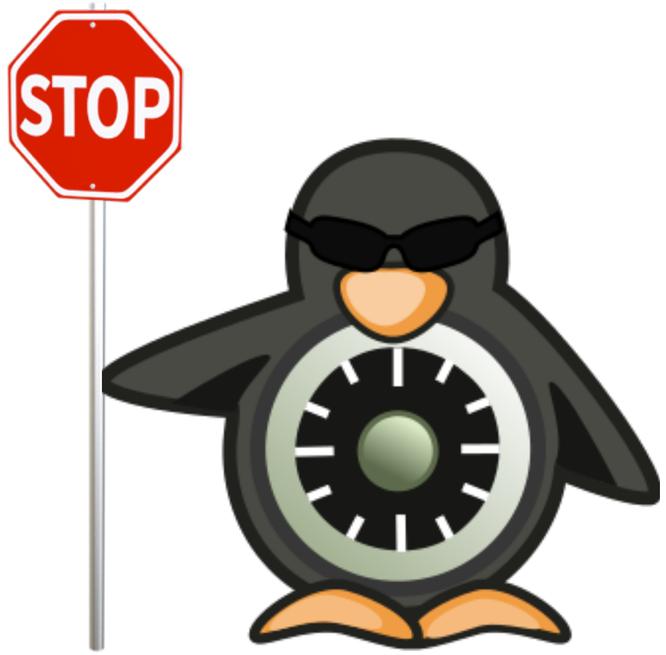
  …limit the impact from compromised services
  …instance isolation in multi-tenant environments

- Restrict users

  …users with limited scopes
  …administrators for specific services

- Control access to sensitive information

  …using confidentiality levels (e.g. *public* ⇢ *top secret*)



Why Use SELinux?

- Confine services

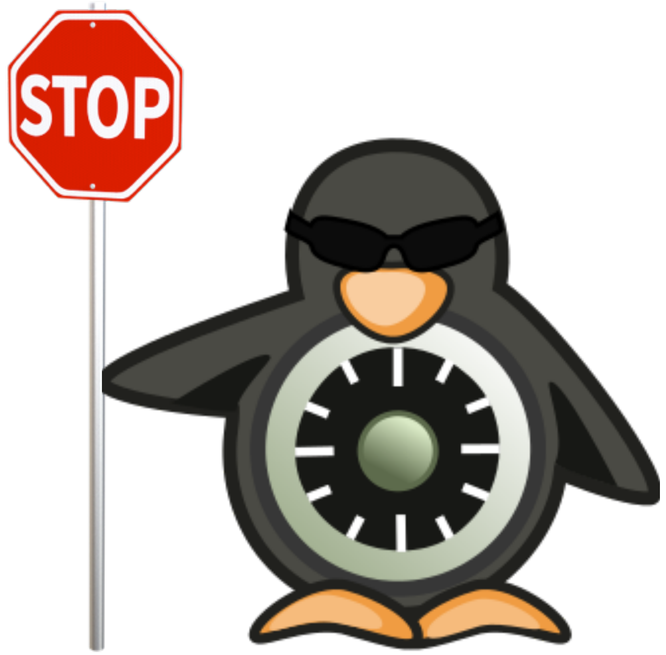
  …limit the impact from compromised services
  …instance isolation in multi-tenant environments

- Restrict users

  …users with limited scopes
  …administrators for specific services

- Control access to sensitive information

  …using confidentiality levels (e.g. *public* ⇢ *top secret*)



Traditional Access Model

- The `root` user has **total** control over the system

  …processes running as `root` have no restrictions
  …*capabilities* can be delegated in part or in full

- Users **decide** on permissions for their own files*

  …they can provide access to other users or groups (or everybody)
  …they just can't transfer this right to other users (i.e. change ownership)

*DAC - **Discretionary** Access Control

Traditional Access Model

- The `root` user has **total** control over the system

  …processes running as `root` have no restrictions
  …*capabilities* can be delegated in part or in full

- Users **decide** on permissions for their own files*

  …they can provide access to other users or groups (or everybody)
  …they just can't transfer this right to other users (i.e. change ownership)

*DAC - **Discretionary** Access Control

SELinux Access Model

- System-wide access policy

  …defined by the administrator and **immutable** to users*
  …where **everything** has to be authorized **explicitly** (*deny by default*)

- Control is based on *type enforcement*

  …processes, files, sockets, etc. have an associated **type** (context)
  …the policy defines authorized type **interactions** and **transitions**

*MAC - **Mandatory** Access Control

SELinux Access Model

- System-wide access policy

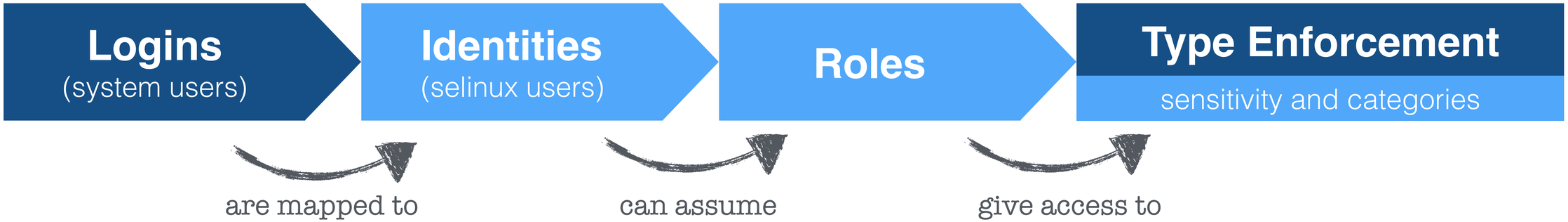
  …defined by the administrator and **immutable** to users*
  …where **everything** has to be authorized **explicitly** (*deny by default*)

- Control is based on *type enforcement*

  …processes, files, sockets, etc. have an associated **type** (context)
  …the policy defines authorized type **interactions** and **transitions**

*MAC - **Mandatory** Access Control

SELinux Access Model



- Identities and Roles **have no permissions** by themselves*

  …they're just ways to reach sets of *type enforcement* rules

- Objects can (optionally) have sensitivity levels** and categories***

  …sensitivity levels follow a *read down* and *write up* model (Bell — La Padula)
  …categories follow dominance rules (set intersections)

*RBAC - **Role-Based** Access Control **MLS - **Multi-Level** Security ***MCS - **Multi-Category** Security

This was just **theory**, you can forget all about it now…

The Reference Policy

- Confines a subset of services (*targeted policy*)

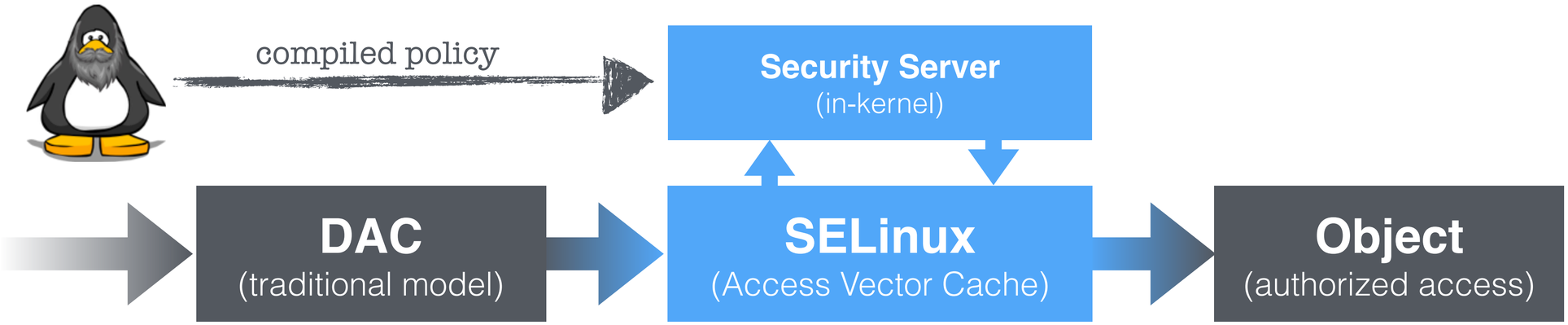
  …**all** services without an **explicit policy** run **unconfined**
  …**all** users are **unconfined by default**

- Additionally…

  …a service may run in *permissive* mode within an *enforcing* system
  …unconfined domains can be disabled (*strict policy*)

github.com/TresysTechnology/refpolicy/wiki

Access Control



- SELinux gets involved only when the DAC **already allows** access

  …so, it can only give permissions that the user **already has** from the traditional model

- Decisions are stored in the *Access Vector Cache* (**AVC**)

  …that's why `audit.log` messages are called AVCs

Security Context

`system_u:object_r:user_home_t:s0-s15:c0.c1023`

identity (selinux user) | **role** (active entities) | **type** | **sensitivity** (list or range) | **categories** (list or range)

- With the targeted policy, *identity* and *role* are mostly **irrelevant**

  …and can be safely **ignored**, even when writing policy rules for new services
  …*sensitivity* (always "s0") and *categories* (empty) can also be safely **ignored**

- For passive entities (e.g. files), the role is **always** `object_r`

  …where `object_r` is just a placeholder for entities that can never assume a role

— DEMO —

Reading List

- Where to start:

  SELinux Intro (Concepts) — www.trust.rub.de/media/ei/attachments/files/2009/02/selinux_intro_0.1_.pdf
  User's and Administrator's Guide (RHEL 7) — red.ht/1VmGJ9j
  CentOS Wiki — wiki.centos.org/HowTos/SELinux
  Gentoo Wiki — wiki.gentoo.org/wiki/SELinux

- Writing your own policy modules:

  Dan Walsh (Red Hat) — danwalsh.livejournal.com/35127.html
  Reference Policy — selinuxproject.org/page/NB_RefPolicy
  Reference Policy API — oss.tresys.com/docs/refpolicy/api/system.html

Thanks!

Questions?

Carlos Rodrigues
cer@brpx.com

brpx Bright Pixel

You can find these slides at speakerdeck.com/carlosefr