IBM IT Education Services

# SE Linux Implementation LINUX20

Russell Coker

# Topic Objectives

**In this topic students will learn :**

- The basic concepts of Security Enhanced Linux
- The difference between MAC and DAC
- The concepts of Domain-Type access control
- The aims of configuring SE Linux

# What is SE Linux?

- A system for Mandatory Access Control (MAC) based on the Linux Security Modules (LSM) framework
- Uses features of role-based and domain-type access control
- Tracks user identity through all operations
- Removes the power of UID 0, I have run several machines with root as the guest account

# LSM – Linux Security Modules

- Framework for security enhancements to the Linux kernel
- Devised at the request of Linus so that one kernel source tree can have multiple security modules available (Linus did not want to choose between the available modules)
- Restrictive controls only (apart from capabilities which could be used in a non-restrictive manner although this is not done in SE Linux)
- SE Linux is based on LSM and is included in the standard 2.6.x kernel
- Other LSM security modules have been written, but SE Linux is the only usable one (and the only serious one included in the standard kernel)

# What is wrong with Unix security?

- Programs have full control over the access given to files they create (DAC)
- Therefore no protection against malicious software, “social engineering”, and bugs in privileged software which may result in the software granting inappropriate access to files (EG creating a mode 777 file in /tmp)
- Too coarse grained - root vs non-root gives boolean security model for many cases
- Security model does not allow tracking of identity across change of UID

# Domain Type access control 1/2

- Every process has a domain, every object (file, directory, socket, etc) has a type. The domains are a sub-set of the types (IE types that can apply to processes).
- The /proc files related to a process have the same type as the process itself
- The domain of a process will be used as a target context for operations such as sending signals
- The domain of a process may be changed at exec time either automatically through policy or through code in **login** type programs
- Different domains have different access rights, no domain is required to have a superset of the access of other domains

# Domain Type Access Control 2/2

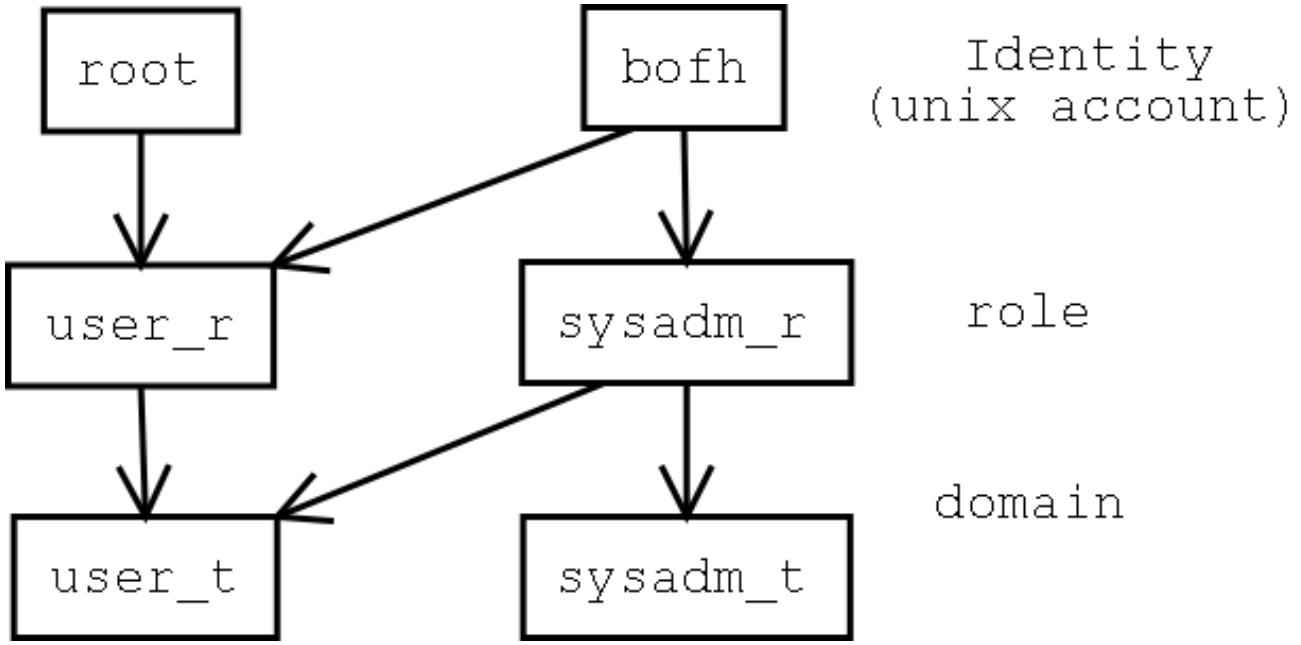
- The user can re-authenticate at any time to change domains with **newrole** and some other similar programs, or the domain can be changed automatically by file execution according to policy
- Each object that a process may act on has a type
- Policy rules determine what access every domain has to each type
- The number of domains can be varied according to needs, having a single domain would give the same result as a non-SE system. The more domains the mode detailed the control you have over security and the more work to set it up.
- The new **Targetted** policy for Fedora will give less control and therefore less configuration work

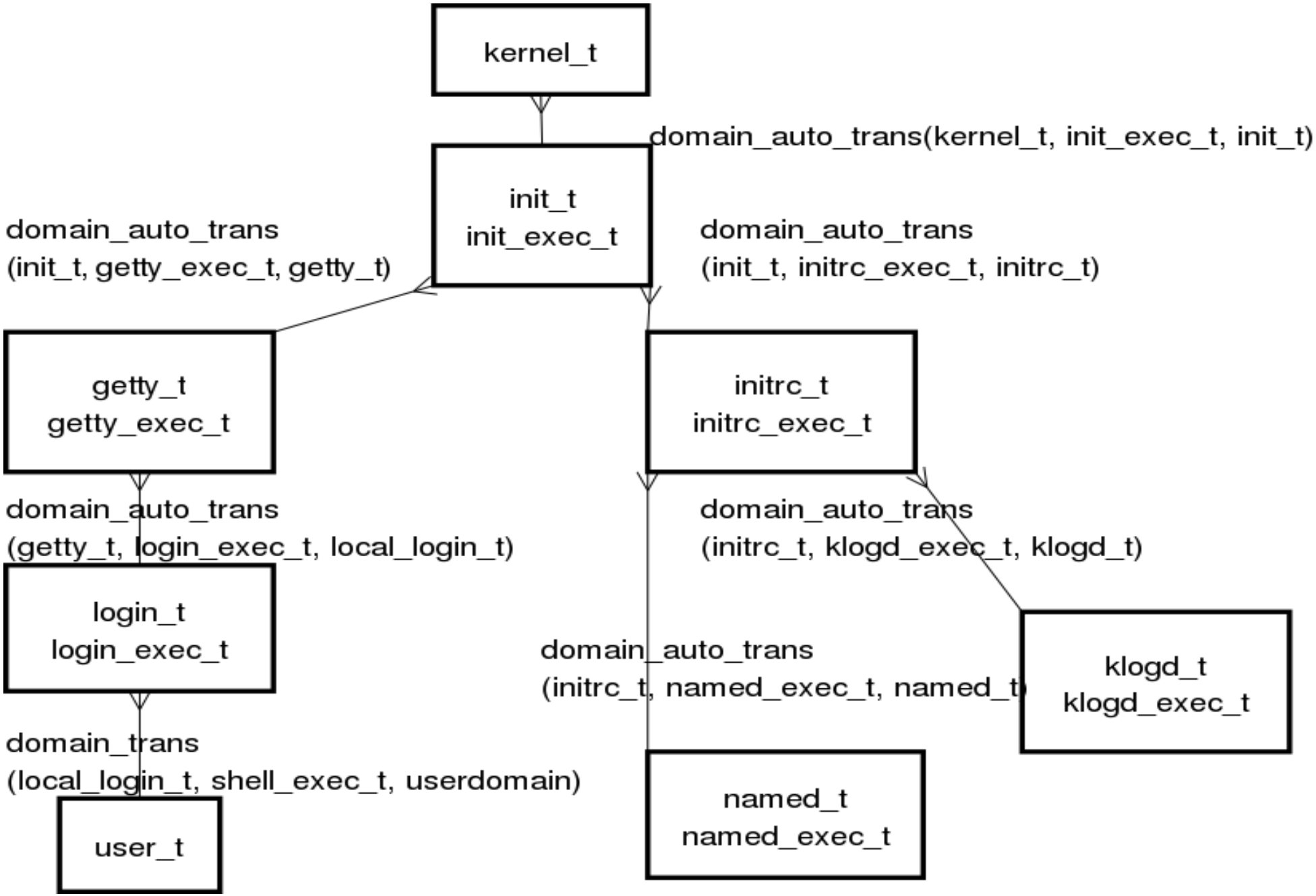
# Role Based Access Control

- Each role has a list of domains that may exist in it
- At login time the security context is changed (identity, role, and domain), also the **newrole** program may be used to change roles (comparable to an **su** operation)
- A role doesn't often change, unlike the domain which may change often automatically without the user noticing
- The role determines which domains are permitted
- Roles may also be changed through role_transition rules in some situations, this is currently only used for the administrator to launch daemons

# Identities

- The Identity is usually the Unix account name and is compiled into the policy database
- Identity controls the available roles which controls the available domains – but this level of control is not used much in the **Targetted** policy





# Policy

- Written at a high level with M4 macros
- Compiled into a binary form that is understood by the kernel
- Loaded by /sbin/init at the start of the boot process before any other programs are executed
- A modified policy can be loaded at any time by the administrator

# Kernel interfaces

- **selinuxfs** file system (almost always mounted on /selinux) used for loading policy, enabling/disabling SE Linux, and querying the kernel policy database
- /proc/PID/attr/* files are used for discovering the current and previous contexts of a process and for a process to request that a non-default context be used for a program it executes or a file it creates
- Setting the context of a file system object involves setting the value of the **security.selinux** XATTR
- Querying the context of a file system object requires reading the **security.selinux** XATTR. The XATTR interface is supported by **devpts** and may be added to other pseudo file systems in future. No support on /proc at the moment.

# Distinguishing characteristics of SE Linux

- In SE Linux access is based on file type not name. Creating a hard-link to a file will not give a different level of access to the file.
- File type can be assigned to a file via an XATTR (not specified in configuration). At mount time all files on a file system can be assigned a file type.
- Requires modified utilities and applications to use full functionality
- A simpler policy could be written which does not require modified utilities, but no-one has published such a policy yet
- Is fully configurable by policy, changes to security requirements do not require recompiling applications

# Q/A

- Those of you who are staying for the lab should start the Fedora Core 2 installation process on their machines now so that it can happen during the break.

| | |
|---|---|
| http://www.nsa.gov/selinux/ | Main NSA SE Linux site |
| http://www.coker.com.au/selinux/ | My SE Linux web pages |

Russell Coker <russell@coker.com.au>