Bootstrapping the LSM policies in RPM

Elena Reshetova, Intel OTC

INTEL OPEN SOURCE TECHNOLOGY CENTER

Agenda

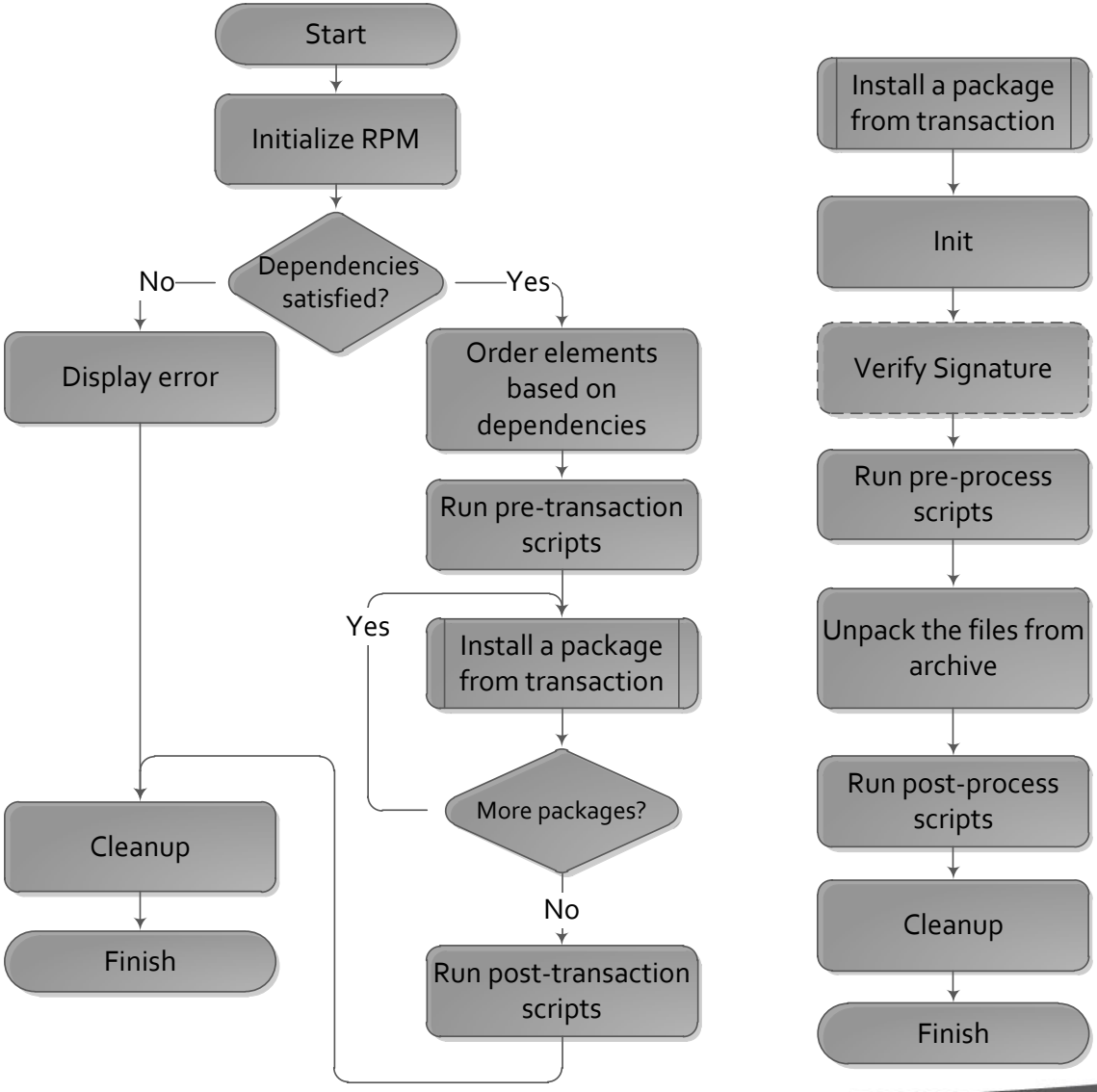
- RPM introduction
- SELinux in RPM
- Goals for RPM security plug-in
- Proposal
- Beyond native applications
- Conclusions & QA

RPM Introduction

- Package Management System
- .rpm binary packages
- http://www.rpm.org/
- Distributions:
  - Red Hat Enterprise Linux, the Fedora Project, SUSE Linux Enterprise, openSUSE, CentOS, Mageia, Tizen and many others.

RPM Installation cycle

- **Transaction**
  - Set of rpms to be installed/removed
- **Transaction element**
  - A single rpm package
- **Optional ”Verify Signature”**
  - Signature check can be skipped

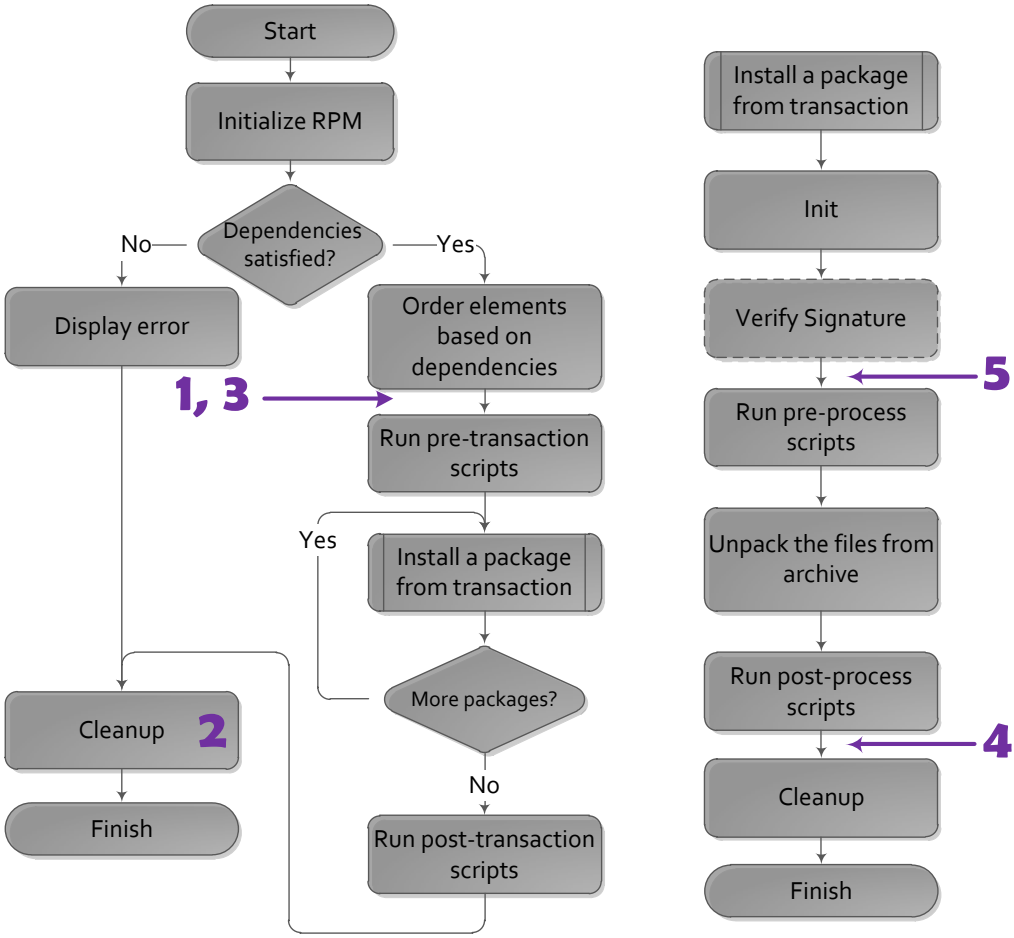


SELinux in RPM

- Goal: setup SELinux policies and label files
- Implementation:
  - mostly inside the sepolicy plugin
  - Hooks
    - PLUGINHOOK_INIT
    - PLUGINHOOK_CLEANUP
    - PLUGINHOOK_OPENTE
    - PLUGINHOOK_COLL_POST_ADD
    - PLUGINHOOK_COLL_PRE_REMOVE
  - Some code is in rpm itself
    - Executing maintainer scripts (rpm_execcon())
    - Setting up flags
    - Various labelling-related tasks (rpmtsSELabelInit(), rpmtsSELabelFini())

SELinux in RPM: Hooks

1. PLUGINHOOK_INIT
   - Initialization hook
2. PLUGINHOOK_CLEANUP
   - Cleaning up hook
3. PLUGINHOOK_OPENTE
   - Extract the policy from package
   - Add to the list of currently processed policies
4. PLUGINHOOK_COLL_POST_ADD
5. PLUGINHOOK_COLL_PRE_REMOVE
   - Load the policy, label filesystem

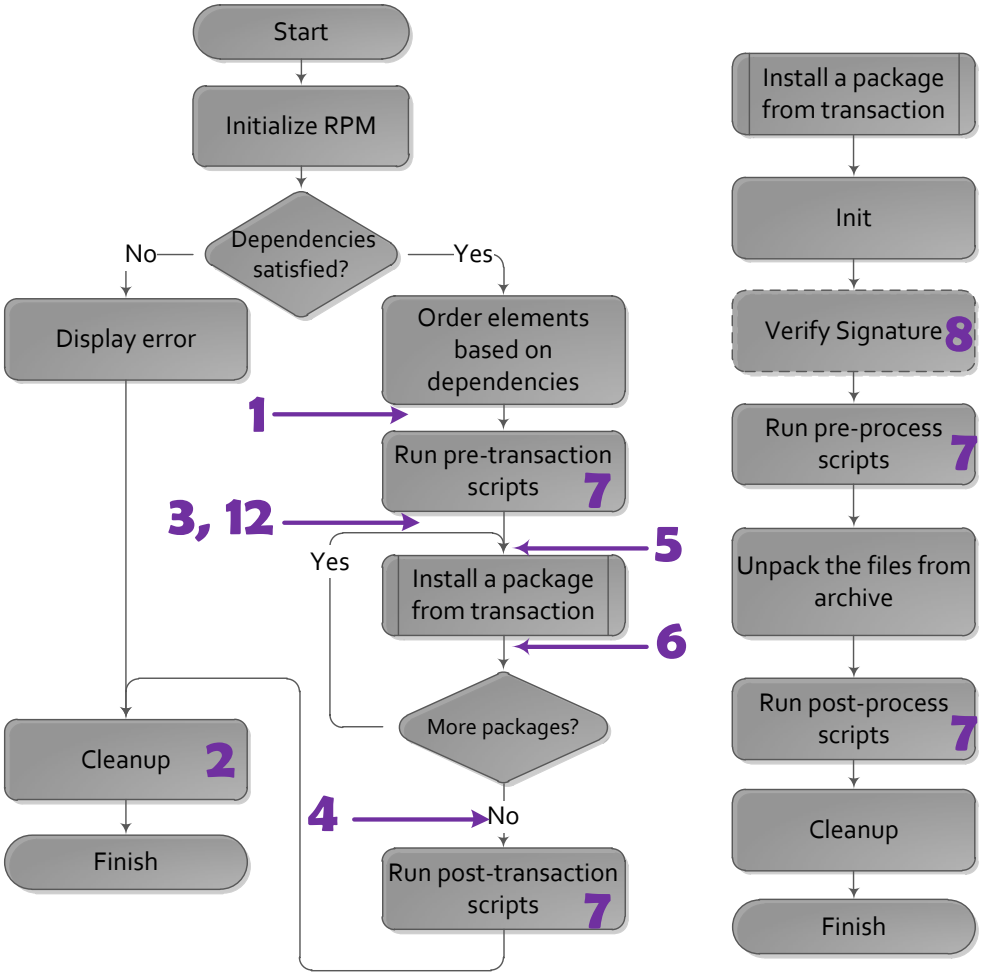


Our goals for RPM security plug-in

- **Smack AC**
  - Manifest per package
    - AC domain definitions and rules
    - Filesystem objects labeling
  - Limitation of privilegde for scripts
- **Device Security Policy Management**
  - Defining set of trusted sw sources and their privilegdes
- **Integrity Protection**
  - Bootstraping IMA reference hashes
- **...**

Hooks proposal – 1/2

1. SECURITYHOOK_INIT
2. SECURITYHOOK_CLEANUP
3. SECURITYHOOK_PRE_TSM
4. SECURITYHOOK_POST_TSM
   - Before and after transaction
5. SECURITYHOOK_PRE_PSM
6. SECURITYHOOK_POST_PSM
   - Before and after package installation
7. SECURITYHOOK_SCRIPT_EXEC
   - Wrapper around script execution



Hooks proposal – 2/2

8. SECURITYHOOK_VERIFY

- Point of enforcement and control

9. SECURITYHOOK_FSM_OPENED

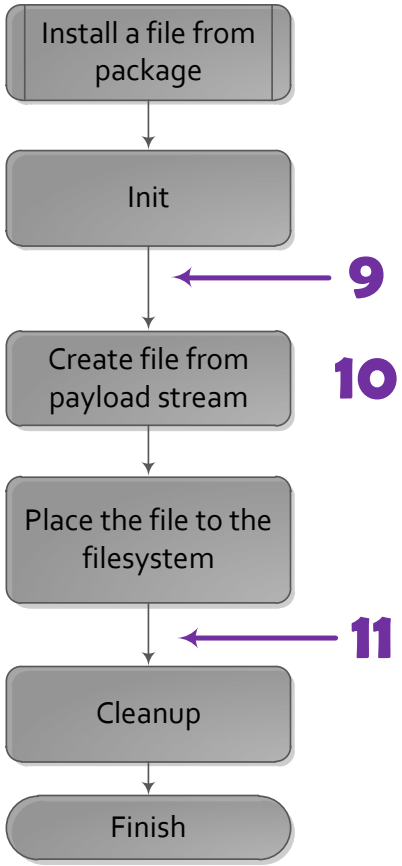
10. SECURITYHOOK_FSM_UPDATED

11. SECURITYHOOK_FSM_CLOSED

- Per file hooks
- Allow to label the file, setup and calculate reference hashes

12. SECURITYHOOK_FILE_CONFLICT

- Control over the conflict situations



Beyond native applications

- Non-native components uprise
  - Usually use a separate installer
- Manifests multiplying
- Many duplicated functionality
  - Creating rules for LSM
  - Labelling files

- Solution: Security plugin functionality as separated shared library

Conclusions

- **RPM**
  - A unified layer of security hooks
  - Easy to integrate any LSM or security component
  - Keeping implementation and interface separated

- **Future**
  - Support of different simultaneous security plug-ins

Questions?

Credits:

- Tero Aho & Ilhan Gurel
  - Implementation of MSSF rpm plug-in and initial design of hooks

Code repo:

https://github.com/ereshetova/rpm/tree/security-changes

Wiki:

https://wiki.tizen.org/wiki/Security/Application_installation_and_Manifest

Contact:

elena.reshetova@intel.com