SUSE Security Process

An overview on technical level

Marcus Meißner
Teamlead SUSE Security
meissner@suse.de

Novell

SUSE Security Team

- Tasks:
  - Incident handling
  - Proactive work (auditing, design reviews)
  - Research and Integration of new technologies
- Focus on OpenSource parts of the Linux product lines
- Tightly cooperating with:
  - R&D, QA, NTS, Maintenance, Customers

Security Work – what is it?

- not cool
- no fun
- does not make you popular
- tiring work

Security Problems over time

- Buffer overflows
- Format string problems
- Integer overflows (Buffer overflows strike back)
- Last 2 years:
  - image processing libraries
  - problems in web applications
- This year:
  - problems in web applications
- Problem:
  - more and more code operates on data from the Internet
  - applications grow and grow and grow

Non Incident Work

Audit security relevant packages

- network and system daemons, setuid binaries
- design of new technologies like D-BUS
- other security critical packages

Deploy automated measures

Develop new technologies

Educate

- write papers
- hold lectures on security topics

Research

- research into new technologies and attack vectors

Making code harder to exploit

- Overflow checking / mitigation:
  - -D_FORTIFY_SOURCE=2 (default on 10.0, 10.1...)
  - -fstack-protector
  - heap structure validation
  - mangling of pointers that live in dangerous areas
  - randomizing address space
- Automated code checking
  - Annoying gcc warnings
  - 3rd party tools
- Force^WTeach people to write better code

Confinement

- No SELinux here
  - nice idea and formal approach
  - too complicated to setup for both user and admin
- AppArmor
  - access restrictions on application level
  - confines file access, capabilities, program starts
  - globbing and wildcards possible
  - no all-or-nothing approach like SELinux
  - light version on 10.0, full OpenSource now
  - LSM soon in mainline kernel

Product lines

- SUSE Linux (Retail , Box)
  - 2 years supported, gets security and critical bugfixes
  - released every 6 months
  - 4 - 5 active at every time
- SUSE Linux Enterprise Server
  - 5 years regular maintenance (+2 years extended)
  - longer release cycles
  - currently: SLES 8, SLES 9, SLD 1, NLD 9, OES
  - Soon: SLES 10, SLED 10
  - Active: 2 major products, 3 derivated products
- 5 different main codestreams (+ derivates)

Incident Handling – Entering SUSE

Getting knowledge of security problem

- public mailinglists
- closed forums (cross vendor coordination)
- new package releases
- our own security audits
- reports to contact address (security@suse.de)

Tracking

- discard, if affected package is not in active products
- discard, if affected package version is not in active products
- open a Bugzilla entry

Incident Handling – Tracking

Bugzilla

- Is our incident tracking tool
- Security Team adds initial information to new bugreports:
  - detailed description
  - Vulnerability IDs (CVE, VU#, ...)
  - affected package versions and products
  - patch(es) to fix issue (if any)
  - sample exploit(s) (if any)
  - decision on whether to fix for older products or not
- Assigned to packager
- Assisting with finding patches, fixing and priority

Incident Handling – Fixed Packages

Package maintainer work

- Reviews fixes and affected products
- Submits fixed packages (source) for buildsystem
- Source level patch review is done by Buildsystem Team
- Buildsystem Team checks package into package repository of old products

Buildsystem

- Consistency checks during build
- Automated rebuilding all dependend packages
- No fixed (bitwise same) binaries due to rebuilds

Incident Handling – Patchset Creation

Creating the patch set:

- accompanies fixed package up to release
- tracked by SWAMP (SUSE Workflow management tool)
- created by Security Team
  - what packages, what distributions
  - description
  - optional pre or post installation messages
  - links back to Bugzilla and SWAMP
- meta patchfile gets checked into buildsystem
  - collects RPMs out of current state of buildsystem and fixates them
  - prepares the patchset the customer will see for QA

Incident Handling – QA

QA

- Uses created patchset
- Check reproducability of available exploits
- Applies patches just like customer would, from
  - YaST Online Update for SUSE Linux and SLES
  - Red Carpet / ZLM for OES and NLD
- System integration QA (checking RPM dependencies)
- Component Integration QA
  - Package testcases are run (automated and manual)
  - rerun exploit
- process goes back to packager if QA fails

Incident Handling – Release

Not before:

- coordinated disclosure date
- QA approval

On approval:

- patch is copied to staging infrastructure in the same way as for QA
- no further manual steps
- NTS reviews documentation and publishes TID article

Security advisory released

How can you help?

- User / Administrators
  - Install Security Updates
  - Report crashes in Applications
  - Monitor your servers
- Developer
  - Program safely
  - use better languages
  - security conscious design

Its all about certification.

- Security - not a feature, but a process
- Certification describes configurations:
  - profiles defining scenarios of users and attackers
  - versions of installed software
  - content of configurations files
  - hardware
- and processes:
  - security handling during the product lifecycle
  - documentation
  - physical security

Languages

- C
- C++
- Managed Languages and Environments
  - Java
  - C#
- Script
  - perl
  - php