
SCAP Security Guide Questions / Answers

Contributor WorkShop

Ján Lieskovský
November 2015

Agenda

- Introductory Notes
 - SSG Repository Structure
 - Contributing To SSG
 - Developer Workflow
-

SSG Q/A WorkShop - Introductory Notes

[SCAP Security Guide \(SSG\):](#)

- delivers security guidance, baselines, and associated validation mechanisms using the SCAP protocol
-

SSG Q/A WorkShop - Introductory Notes

[SCAP Security Guide \(SSG\):](#)

- delivers security guidance, baselines, and associated validation mechanisms using SCAP protocol
 - links security hardening advice for (Red Hat) products with official requirements (NIST SP 800-53, USGCB ..)
-

SSG Q/A WorkShop - Introductory Notes

[SCAP Security Guide \(SSG\):](#)

- delivers security guidance, baselines, and associated validation mechanisms using SCAP protocol
 - links security hardening advice for (Red Hat) products with official requirements (NIST SP 800-53, USGCB ..)
 - bridges the gap between official security requirements and specific security hardening implementation details
-

SSG Q/A WorkShop - Introductory Notes

SCAP -- [Security Content Automation Protocol](#):

- Suite of specifications
 - to standardize format and nomenclature
 - by which software flaw and security configuration information
 - is communicated both to machines and humans
-

SSG Q/A WorkShop - Introductory Notes

SCAP -- [Security Content Automation Protocol](#):

- Latest version 1.2 ([NIST SP 800-126 Rev. 2](#))
 - Work on 1.3 version in progress
 - Design and development comments were [accepted till 28-th Sep 2015](#)
-

SSG Q/A WorkShop - Introductory Notes

SCAP v1.2 set of:

- Languages
- Reporting Formats
- Enumerations
- Measurement / Scoring Systems
- Integrity Model

XCCDF 1.2

OVAL 5.10.1

OCIL 2.0

ARF 1.1

AI 1.1

CCE v5

CPE 2.3

CVE

CVSS 2.0

CCSS 1.0

TMSAD 1.0

SSG Q/A WorkShop - Introductory Notes

SCAP standards relevant to SSG:

[XCCDF 1.2](#)

[OVAL 5.10.1](#)

[OCIL 2.0](#)

[ARF 1.1](#)

[CCE v5](#)

[CPE 2.3](#)

[CVE](#)

SSG Q/A WorkShop - Introductory Notes

SCAP standards we will cover today:



XCCDF 1.2



OVAL 5.10.1

SSG Q/A WorkShop - SSG Repository Structure

Currently supported SSG products:

```
[iankko@dhcppc4 scap-security-guide]$ tree -d -L 1 -C
.
├── Chromium
├── config
├── docs
├── Fedora
├── Firefox
├── JBossEAP5
├── JBossFuse6
├── JRE
├── OpenStack
├── RHEL
├── RHEVM3
├── shared
└── Webmin

13 directories
[iankko@dhcppc4 scap-security-guide]$
```

SSG Q/A WorkShop - SSG Repository Structure

Supported versions of Red Hat Enterprise Linux:

```
[iankko@dhcppc4 scap-security-guide]$ tree -d -L 1 -C RHEL
RHEL
├── 5
├── 6
└── 7

3 directories
[iankko@dhcppc4 scap-security-guide]$
```

SSG Q/A WorkShop - SSG Repository Structure

Directory structure of concrete product (RHEL/6):

```
[iankko@dhcppc4 scap-security-guide]$ tree -L 1 -C RHEL/6
RHEL/6
├── input
├── kickstart
├── Makefile
├── output
├── README
├── tests
├── transforms
└── utils

6 directories, 2 files
[iankko@dhcppc4 scap-security-guide]$
```

- Exceptions are JBossEAP5 and JBossFuse6 products
-

SSG Q/A WorkShop - SSG Repository Structure

Observation #1:

Modifying content of `input/` folder for each product (except JBossEAP5) is **sufficient** to contribute **a new group or rule**.

SSG Q/A WorkShop - SSG Repository Structure

Directory structure of `input/` folder per product (RHEL-6):

```
[iankko@dhcppc4 scap-security-guide]$ tree -L 1 -C RHEL/6/input/  
RHEL/6/input/  
├── auxiliary  
├── guide.xml  
├── guide.xslt  
├── intro  
├── oval  
├── profiles  
├── remediations  
└── xccdf  
  
6 directories, 2 files  
[iankko@dhcppc4 scap-security-guide]$
```

SSG Q/A WorkShop - SSG Repository Structure

Meaning and structure of special `shared/` directory:

```
[iankko@dhcppc4 scap-security-guide]$ tree -d -L 1 -C shared/  
shared/  
├── modules  
├── oval  
├── references  
├── remediations  
├── transforms  
└── utils  
  
6 directories  
[iankko@dhcppc4 scap-security-guide]$
```

- Used to hold content applicable to ≥ 2 products
 - Easier maintenance (avoids duplicated data)
 - Process of unification (still) not finished
-

SSG Q/A WorkShop - SSG Repository Structure

Observation #2:

If particular **OVAL** check or **remediation script** has same form across multiple products it should be placed into **shared/** directory (rather than into concrete product directory).

SSG Q/A WorkShop - SSG Repository Structure

Observation #3:

If by porting an OVAL check or remediation script between the products (e.g. `RHEL/6` => `RHEL/7`) that OVAL check / remediation is recognized as being identical for the two products, it should be placed into the `shared/` directory (rather than into concrete product directory).

SSG Q/A WorkShop - Contributing To SSG

SSG “shorthand” format:

- Applicable to both XCCDF and OVAL
 - Idea:
 - Replace the need to edit one big final XCCDF / OVAL file
 - By editing (bigger) set of smaller files
-

SSG Q/A WorkShop - Contributing To SSG

SSG “shorthand” format -- Idea (continued):

- The content of these files is to be grouped by the relation of the rule to:
 - Higher set of rules for the same OS component (XCCDF),
 - Or to form one OVAL definition
-

SSG Q/A WorkShop - Contributing To SSG

SSG “shorthand” format -- Idea (continued):

- Let the build system to take care for actions like:
 - Expand concrete namespace identification for objects
 - Combine the elements into resulting output (XCCDF ,
OVAL) documents
-

SSG Q/A WorkShop - Contributing To SSG

SSG “shorthand” XCCDF

vs

Official XCCDF:

SSG Q/A WorkShop - Contributing To SSG

SSG “shorthand” OVAL

vs

Official OVAL:



SSG Q/A WorkShop - Contributing To SSG

[XCCDF \(Extensible Configuration Checklist Description Format\)](#): (very informally):

- Organization layer
 - Presentation layer
-

SSG Q/A WorkShop - Contributing To SSG

Contributing new XCCDF rule:

- Relevant SSG repository folders:
`<product>/input/xccdf/system`
`<product>/input/xccdf/services`
 - Identify the **SSG** product the rule is to be applicable for
(`RHEL/6`, `RHEL/7` etc.)
 - Based on that select concrete SSG subfolder the new rule will be added into
-

SSG Q/A WorkShop - Contributing To SSG

Contributing new XCCDF rule (continued):

- Identify the **XCCDF** group the rule is to be applicable for:
 - Is this new rule for system service or other OS component?
 - for service place it into `<product>/input/xccdf/services` folder
 - for other OS component (kernel, networking etc.) place it into `<product>/input/xccdf/system`
-

SSG Q/A WorkShop - Contributing To SSG

Contributing new XCCDF rule (continued):

- Identify the **XCCDF** group the rule is to be applicable for:
 - Does a particular XCCDF group already exists for this functionality?
 - If yes, place the rule into that group of XCCDF shorthand file relevant to that system component, e.g.
`<product>/input/xccdf/system/auditing.xml`

SSG Q/A WorkShop - Contributing To SSG

Contributing new XCCDF rule (continued):

- Identify the **XCCDF** group the rule is to be applicable for:
 - Does a particular XCCDF group already exists for this functionality?
 - If no, create a new XCCDF group for that rule
 - Place the created group into particular XCCDF shorthand document based on “**locality principle**”

SSG Q/A WorkShop - Contributing To SSG

- Identify if the **XCCDF** group for the rule already exists:

```
[root@localhost ~]# rpm -q scap-security-guide
scap-security-guide-0.1.26-1.el6.noarch
[root@localhost ~]# grep '<Group' /usr/share/xml/scap/ssg/content/ssg-rhel6-xccdf.xml | head -10
<Group id="intro">
  <Group id="general-principles">
    <Group id="principle-encrypt-transmitted-data">
    <Group id="principle-minimize-software">
    <Group id="principle-separate-servers">
    <Group id="principle-use-security-tools">
    <Group id="principle-least-privilege">
  <Group id="how-to-use">
    <Group id="intro-read-sections-completely">
    <Group id="intro-test-non-production">
[root@localhost ~]#
```

SSG Q/A WorkShop - Contributing To SSG

- Available `system/` XCCDF groups for `RHEL/6`:

```
<Group id="system">
  <Group id="entropy">
  <Group id="software">
    <Group id="disk_partitioning">
    <Group id="updating">
    <Group id="integrity">
      <Group id="aide">
      <Group id="rpm_verification">
      <Group id="additional_security_software">
    <Group id="permissions">
      <Group id="partitions">
      <Group id="mounting">
      <Group id="files">
        <Group id="permissions_important_account_files">
        <Group id="permissions_within_important_dirs">
      <Group id="restrictions">
        <Group id="daemon_umask">
        <Group id="coredumps">
        <Group id="enable_execshield_settings">
        <Group id="enable_nx">
```

```
<Group id="selinux">
<Group id="accounts">
  <Group id="accounts-restrictions">
    <Group id="root_logins">
    <Group id="password_storage">
    <Group id="password_expiration">
    <Group id="account_expiration">
    <Group id="accounts-pam">
      <Group id="password_quality">
        <Group id="password_quality_pamcracklib">
        <Group id="locking_out_password_attempts">
        <Group id="set_password_hashing_algorithm">
    <Group id="accounts-session">
      <Group id="root_paths">
      <Group id="user_umask">
    <Group id="accounts-physical">
      <Group id="bootloader">
      <Group id="screen_locking">
        <Group id="gui_screen_locking">
        <Group id="console_screen_locking">
```

SSG Q/A WorkShop - Contributing To SSG

Contributing new XCCDF rule:

- LIVE DEMO



SSG Q/A WorkShop - Contributing To SSG

Contributing new OVAL check:

- Relevant repository folders:

`<product>/input/oval`

`shared/oval`

- LIVE DEMO
-

SSG Q/A WorkShop - Contributing To SSG

Contributing new remediation script:

- Relevant repository folders
 <product>/input/remediations/bash
 shared/remediations/bash
 - Provide standalone bash script
-

Contributing new
remediation script

SSG Q/A WorkShop - Contributing To SSG

-- example:

```
[lankko@dhcppe4 scap-security-guide]$ cat RHEL/6/input/remediations/bash/kernel_disable_entropy_contribution_for_solid_state_drives.sh
# platform = Red Hat Enterprise Linux 6

# First obtain the list of block devices present on system into array
#
# Used lsblk options:
# -o NAME      Display only block device name
# -a          Display all devices (including empty ones) in the list
# -d          Don't print device holders or slaves information
# -n          Suppress printing of introductory heading line in the list
SYSTEM_BLOCK_DEVICES=($(/bin/lsblk -o NAME -a -d -n))

# For each SSD block device from that list
# (device where /sys/block/DEVICE/queue/rotation == 0)
for BLOCK_DEVICE in "${SYSTEM_BLOCK_DEVICES[@]}"
do
    # Verify the block device is SSD
    if grep -q "0" /sys/block/${BLOCK_DEVICE}/queue/rotational
    then
        # If particular SSD is configured to contribute to
        # random number entropy pool, disable it
        if grep -q "1" /sys/block/${BLOCK_DEVICE}/queue/add_random
        then
            echo "0" > /sys/block/${BLOCK_DEVICE}/queue/add_random
        fi
    fi
done
```

SSG Q/A WorkShop - Contributing To SSG

Contributing new XCCDF profile:

- LIVE DEMO



SSG Q/A WorkShop - Developer Workflow

- SSG code is [hosted](#) at GitHub
 - We follow the [GitHub](#) flow:
 - Fork the repo
 - Create feature branch in forked repo
 - Perform, test locally & commit the code changes
 - Push the changes to the feature branch
 - Create a pull request (proper description, labels, milestones)
 - Wait till Jenkins finishes the PR testing:
 - If PR testing by Jenkins failed, inspect the failures & fix them (possibly commit additional changes)
 - If PR testing by Jenkins passed, your job is done
-

SSG Q/A WorkShop - Reviewer Workflow

- SSG code is [hosted](#) at GitHub
 - Reviewing existing PRs is **as much as important** as providing new code changes
 - Reviewer Workflow:
 - Check out the [list of existing PRs](#)
 - Apply the proposed code changes from PR to be reviewed on top of your local branch (synchronized with upstream master)
 - Build the content
 - Test the particular code changes (verify announced problem is corrected)
 - If testing succeeded, **ACK** and merge the PR into upstream
-

SSG Q/A WorkShop - Build the SSG Content

- SSG code is [hosted](#) at GitHub
 - Clone upstream master
 - Build RPM from the upstream content:
 - # make clean
 - # make SSG_VERSION_IS_GIT_SNAPSHOT=no rpm
 - Or [install scap-security-guide RPM package](#) according instructions for your distribution
-

SSG Q/A WorkShop - Where to Go Next

- Check out [SSG upstream wiki pages](#)
 - Check out the [OpenSCAP portal pages](#)
 - Check out [SSG OpenSCAP portal pages](#)
-

SSG Q/A WorkShop - That's It!

- Thank you for your attention.
 - Questions / Feedback Welcome !
-