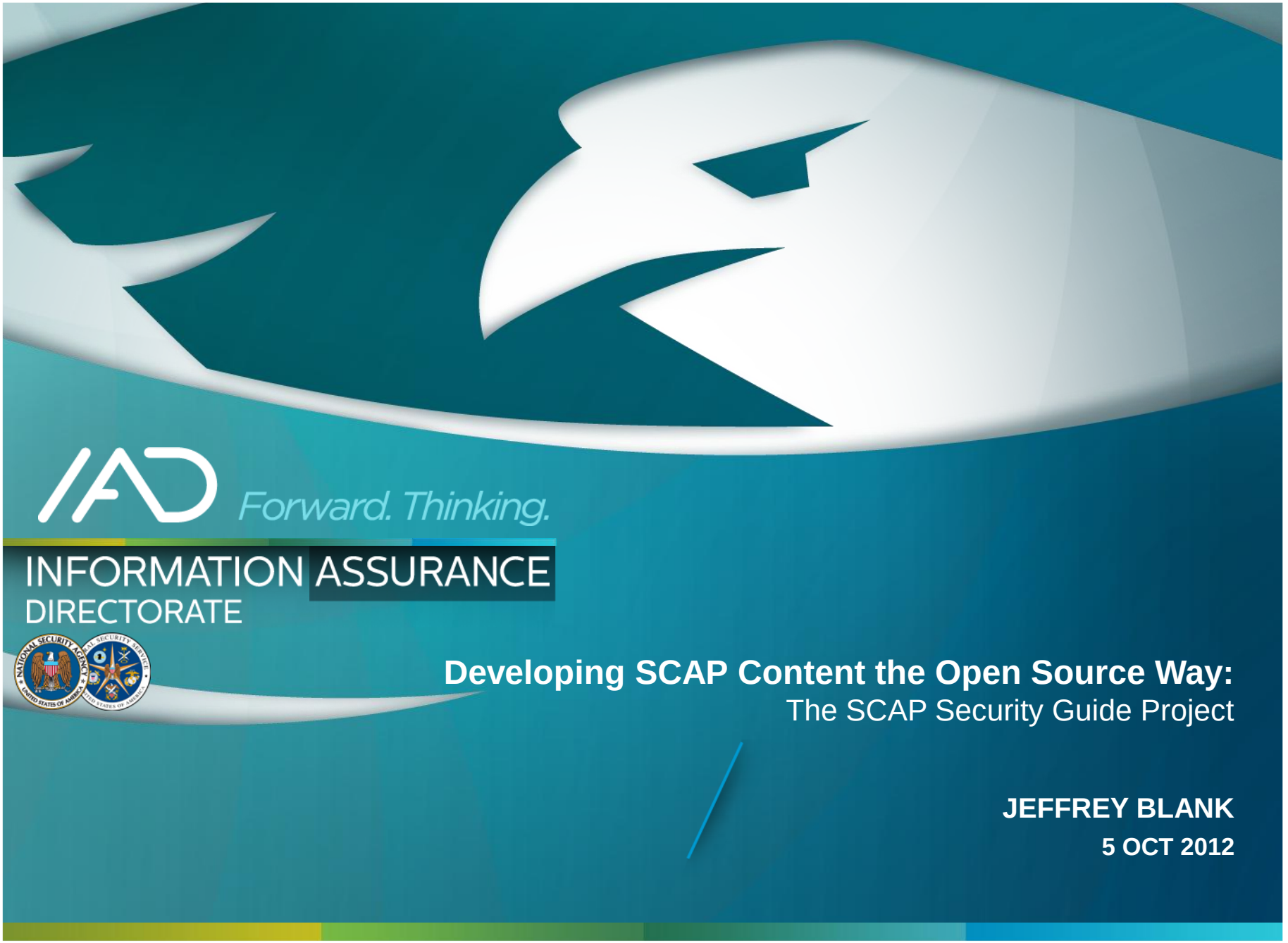

Developing SCAP Content the Open Source Way:

The SCAP Security Guide Project

JEFFREY BLANK

5 OCT 2012

OVERVIEW

- Challenges in SCAP content development
- Benefits and obstacles to open source approach
- Project description
- Goals

CHALLENGES IN SCAP CONTENT DEVELOPMENT

- Existing authoring tools
  - use a GUI approach
  - do not track with current specifications
  - limited support for collaboration
- Existing execution mechanisms
  - OVAL reference interpreter not capable of interpreting OVAL
  - development may be driven by particular tool behavior
  - significant capability diversity on a per-platform, per-tool basis
- Government contract-driven
  - can tend toward isolation

BENEFITS TO OPEN SOURCE APPROACH

- Powerful collaboration tools available
  - wiki, mailing list, ticketing system, versioning systems
  - permit and encourage Internet-wide collaboration
  - change is transparent and accountable
- Enables direct collaboration with vendor
  - and direct communication with user community
- Enables transparent interagency collaboration
  - reduces government waste by pooling resources
  - speeds content development, testing

OBSTACLES TO OPEN SOURCE APPROACH

- Contributor agreements
  - often require assignment or assertion of copyright
  - not compatible with non-copyright status of USG works
    - purely public domain
    - different situation for contractors
  - But it's possible to change these
    - with the assistance of your general counsel – see:

Fedora.

4. Public Domain United States Government Works.

Sections 1 through 3 of this FPCA do not apply to any Contribution to the extent that it is a work of the United States Government for which copyright is unavailable under 17 U.S.C. 105.

5. Acceptance

THE SCAP-SECURITY-GUIDE PROJECT

- An open source project
  - http://www.fedorahosted.org/scap-security-guide
- Government-industry collaboration
- SCAP Content for Red Hat Enterprise Linux 6 (and JBossEAP)
  - XCCDF
  - OVAL
  - OCIL
- Unix-style software development
  - Makefiles, XSLT, and Python
- Uses OpenSCAP for execution capability

RELATED OPEN SOURCE PROJECTS

- OpenSCAP
  - provides execution capability, formatting of results
  - included with platform
    - significantly eases testing and deployment
  - feedback loop between content developers and tool developers
- Aqueduct Project
  - community with tools for hardening/deploying systems
    - for baseline compliance
  - feedback loop between admins and baseline creators
- SecState
  - demonstrates remediation via SCAP
  - expert feedback for content testing

XCCDF CONTENT APPROACH

- Significant overhead reduced using “shorthand” / macros:

```
<Rule id="enable_auditd_service">
<title>Enable auditd Service</title>
<description>The <tt>auditd</tt> service is an essential userspace component of
the Linux Auditing System, as it is responsible for writing audit records to
disk.
<service-enable-macro service="auditd" />
</description>
<rationale>Ensuring that the <tt>auditd</tt> service is active ensures that
audit records generated by the kernel can be written to disk, or that appropriate
actions will be taken if other obstacles exist.
</rationale>
<ident cce="4292-9" />
<oval id="service_auditd_enabled" />
<ref nist="CM-6, CM-7" disa="169,172,174,1353,1462,1487,1115,1454,067,158,831,1123
63,130" />
</Rule>
```


OVAL CONTENT APPROACH

- Templating, minimal transformation to produce real OVAL:

```
<def-group>
 <!-- THIS FILE IS GENERATED by create_package_installed.py.  DO NOT EDIT.  -->
  <definition class="compliance" id="package_rsyslog_installed"
  version="1">
    <metadata>
      <title>Package rsyslog Installed</title>
      <affected family="unix">
        <platform>Red Hat Enterprise Linux 6</platform>
      </affected>
      <description>The RPM package rsyslog should be installed.</description>
    </metadata>
    <criteria>
      <criterion comment="package rsyslog is installed"
      test_ref="test_package_rsyslog_installed" />
    </criteria>
  </definition>
  <linux:rpminfo_test check="all" check_existence="all_exist"
  id="test_package_rsyslog_installed" version="1"
  comment="package rsyslog is installed">
    <linux:object object_ref="obj_package_rsyslog" />
  </linux:rpminfo_test>
  <linux:rpminfo_object id="obj_package_rsyslog" version="1">
    <linux:name>rsyslog</linux:name>
  </linux:rpminfo_object>
</def-group>
```

OCIL CONTENT APPROACH

- Automatic generation from inline XCCDF text
  - for a contrived check system
    - also suitable for use as manual checking text
  - developers simply see <ocil> tags
    - can be generated into boolean question type, and into interrogatory form using its “clause” attribute
- XSLT to extract this from XCCDF
  - to produce standalone OCIL
    - or used to produce tables or guide with such information
  - replace with check-content-ref

GOALS

- Issue “SNAC”-style NSA Security Guide for RHEL 6
- Work with DISA FSO to release STIG from content
- Submit to NIST with DoD as champion agency for USGCB

- Project should enable:
  - unified and coordinated QA activities
  - speedy release of USG baselines for future versions of RHEL

- Encourage this open approach for other platforms

IAD

Forward. Thinking.