OpenSCAP + SCAP Security Guide でポリシー遵守をらくにしよう

レッドハット株式会社
ソリューションアーキテクト 森若和雄
2018.09.10

「セキュリティポリシー遵守してね」「はい」

- システムが **5 台**
  - 「ポリシーを遵守していることを確認してください」
    → (1 日 1 台チェック、**1 週間後**) 「遵守できてました」
- システムが **50 台**
  - 「ポリシーを遵守していることを確認してください」
    → (**2.5 ヶ月後**) 「遵守できてました」
- システムが 50 台
  - 「ポリシーを遵守していることを**毎月**確認してください」
    → .oO(絶対無理だから**人を 3 人に増やして……**)
- **コンテナや VM が 200 台**
  - 「ポリシーを遵守していることを**毎月**確認してください」
    → .oO(**10 人に増員して**……いやさすがに無理だよな……)
    → あきらめる？ 虚偽の報告？**自動化する？**

チェック項目の例

- パスワードのハッシュ関数は SHA512
- 空のパスワードを許可しない
- chmod, chwon などのシステムコールやファイル削除は監査ログに記録する
- 15 分以上何もしないで経過すると ssh のセッションをタイムアウトさせる
- パッケージの GPG 鍵チェックを必須にする
- bluetooth のカーネルモジュールを無効にする

…… などなど

Security Content Automation Protocol

- セキュリティポリシーを遵守しているかチェックする作業を自動化したい！ (手動では無理)
- そのために作られた規格群を SCAP と呼びます
  - 脆弱性の識別 CVE
  - 設定の識別 CCE
  - プラットフォームの識別 CPE
  - 脆弱性の分類と識別 CWE

  識別子をつけて区別できる
  - 脆弱性の深刻さのスコア付け CVSS

  脅威を数値化できる
  - チェック手順の記述言語 OVAL
  - チェックリスト記述言語 XCCDF

  自動実行できるチェックリスト
  - など

SCAP 策定の背景

2002 年 連邦情報セキュリティマネジメント法

- 米国の政府省庁で、**全情報システムにセキュリティ要求事項を反映する**ことが必須に

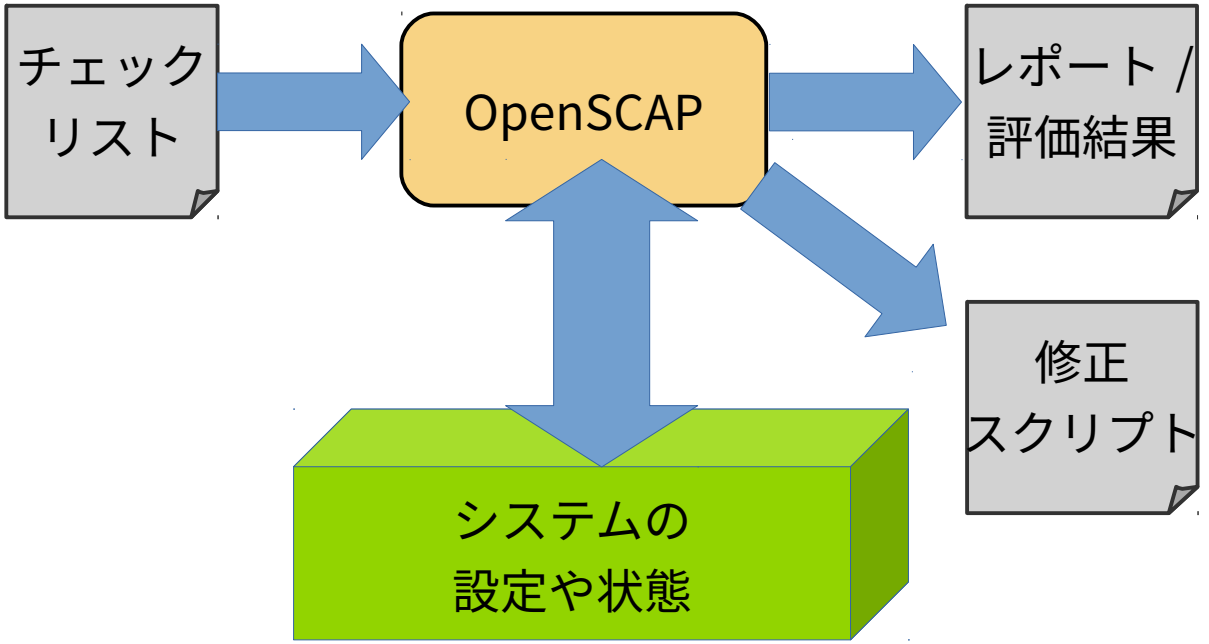
結果として……

- 現場が疲弊
- ミスや判断の相違によるバラつき
- バラバラのツール群による部分的な自動化
  → 標準化された自動化規格の必要性が高まる

2010 年 NIST が SCAP 1.0 をリリース

OpenSCAP

- SCAP の処理系
  - XCCDF で記述されたチェックリストを実行する
  - 評価結果を出力 ,HTML のレポート生成 , 修正スクリプト生成



アンチウイルスや改竄検知等ではない

- OpenSCAP はセキュリティの維持に役立ちますが何でもできるわけではありません
- 代表的なできないこと：
  - イベントドリブンな動作 (「常駐して xx を監視しつづける」のような動作) 全般
  - ポリシーを強制すること (チェックが主な仕事)
  - 過去レポートや他ホストのレポートと比較などのレポート操作
  - ウイルスやマルウェアの検出 (チェックリストが開発されれば部分的には可能)

OpenSCAP で既知の脆弱性を確認する

- チェックリストを取得
  - https://www.redhat.com/security/data/metrics/ にて OVAL および XCCDF 形式で配布
  - 「セキュリティ fix が出荷される前の古いバージョンの rpm が適用されているか」をチェックする
- OpenSCAP でチェックを実施
  - ```
    oscap xccdf eval \
    --results results-xccdf.xml \
    --report report-xccdf.html \
    com.redhat.rhsa-RHEL7.ds.xml
    ```
    ```

SCAP Security Guide(SSG)

SCAP で記述された Linux システム向けのチェックリスト

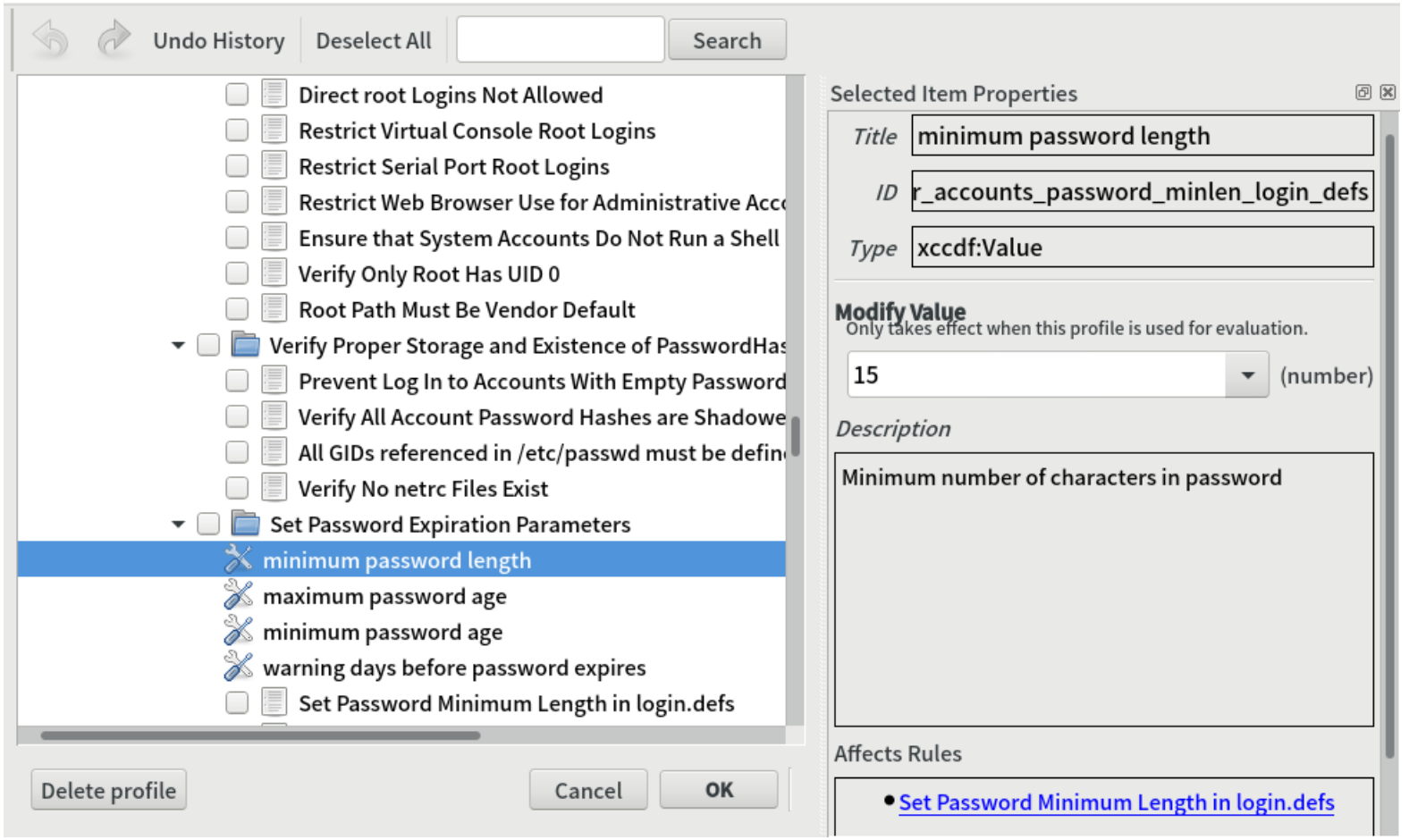
- システム構成や各種設定をチェック
  - ソフトウェアの脆弱性は扱わない
- 各種セキュリティ規格用プロファイル同梱
  - DISA STIG, USGCB, PCI-DSS v3, CIS benchmark(に似たもの) etc.
- 対象ソフトウェアは Fedora, RHEL, CentOS, Debian, Ubuntu, Firefox, Chromium, JRE, OpenStack など多数

SCAP Security Guide の特徴

- NSA, DISA, NIST と Red Hat を中心とするコミュニティで共同開発
  - SSG 以前
    「政府が規格を策定→ベンダが SCAP でチェックリスト作成」 ~3 年
  - SSG 以後
    「共同で SCAP でチェックリストを作成」 ~1 年
  - OVAL でのチェック手順について作成ベンダーによる解釈ブレを排除
- 修正スクリプトを同梱
  - 一部の問題には、 bash または ansible による修正スクリプトを同梱
- カスタマイズして独自プロファイルを作成可
  - チェック項目を取捨選択
  - パスワード長などの項目はパラメータを設定できる

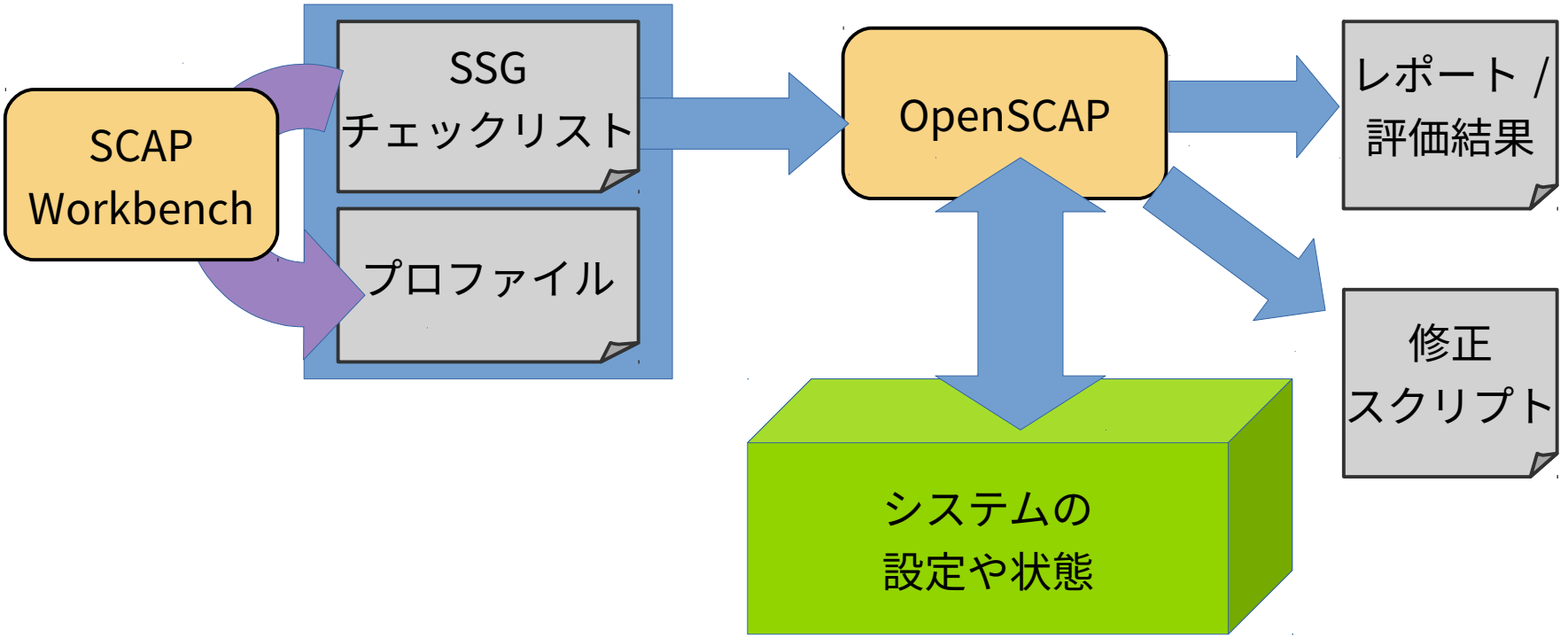
SCAP Workbench

- SSG(XCCDF 文書一般) のプロファイルを作成
  - 既存のチェック項目を GUI で取捨選択、指定



SSG と OpenSCAP でのチェック

- SCAP Workbench で SSG から必要な項目を取捨選択
- 作成したプロファイルを利用して OpenSCAP でチェック



レポート例

- OpenSCAP は XCCDF 形式での結果の他に HTML 形式のレポートも生成

レポート / 評価結果

対象システムのバージョン等 (CPE)

各種アドレス

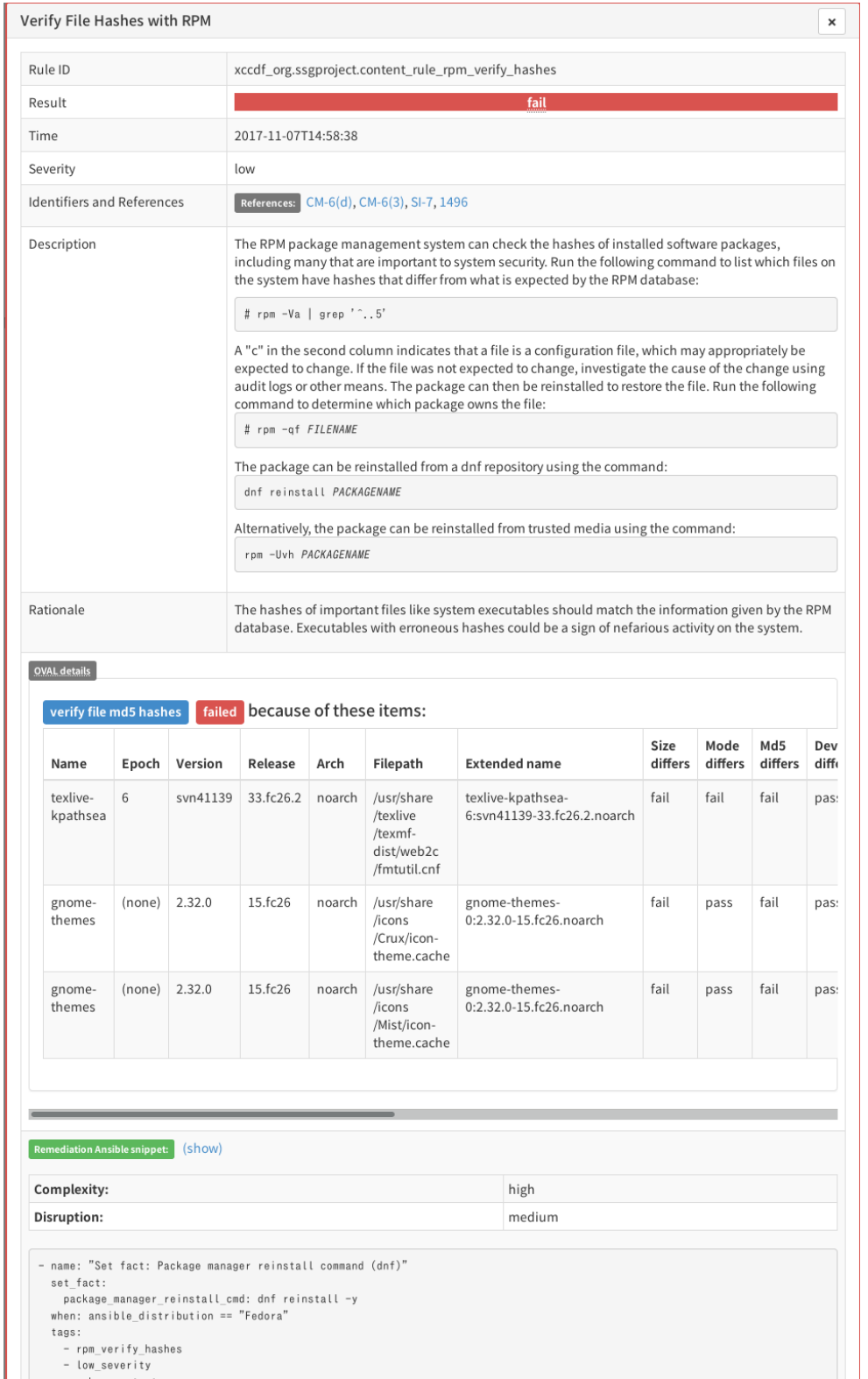
適合状況
不適合ルールの深刻度
重みつきのスコア

各ルール毎の pass/fail

レポート例

レポート / 評価結果

- 各ルールをクリックすると詳細を表示
  - チェック内容
  - 関係する規格
  - 失敗 or 成功した理由
  - 対策用のスクリプト (bash, ansible)



Verify File Hashes with RPM

Rule ID	xccdf_org.ssgproject.content_rule_rpm_verify_hashes
Result	fail
Time	2017-11-07T14:58:38
Severity	low
Identifiers and References	References: CM-6(d), CM-6(3), SI-7, 1496

Description	The RPM package management system can check the hashes of installed software packages, including many that are important to system security. Run the following command to list which files on the system have hashes that differ from what is expected by the RPM database:
	`# rpm -Va | grep '^..5'`
	A "c" in the second column indicates that a file is a configuration file, which may appropriately be expected to change. If the file was not expected to change, investigate the cause of the change using audit logs or other means. The package can then be reinstalled to restore the file. Run the following command to determine which package owns the file:
	`# rpm -qf FILENAME`
	The package can be reinstalled from a dnf repository using the command:
	`dnf reinstall PACKAGENAME`
	Alternatively, the package can be reinstalled from trusted media using the command:
	`rpm -Uvh PACKAGENAME`

Rationale	The hashes of important files like system executables should match the information given by the RPM database. Executables with erroneous hashes could be a sign of nefarious activity on the system.

OVAL details

verify file md5 hashes failed because of these items:

Name	Epoch	Version	Release	Arch	Filepath	Extended name	Size differs	Mode differs	Md5 differs	Dev diff
texlive-kpathsea	6	svn41139	33.fc26.2	noarch	/usr/share /texlive /texmf-dist/web2c /fmtutil.cnf	texlive-kpathsea-6:svn41139-33.fc26.2.noarch	fail	fail	fail	pas
gnome-themes	(none)	2.32.0	15.fc26	noarch	/usr/share /icons /Crux/icon-theme.cache	gnome-themes-0:2.32.0-15.fc26.noarch	fail	pass	fail	pas
gnome-themes	(none)	2.32.0	15.fc26	noarch	/usr/share /icons /Mist/icon-theme.cache	gnome-themes-0:2.32.0-15.fc26.noarch	fail	pass	fail	pas

Remediation Ansible snippet: (show)

Complexity:	high
Disruption:	medium

```
- name: "Set fact: Package manager reinstall command (dnf)"
  set_fact:
    package_manager_reinstall_cmd: dnf reinstall -y
  when: ansible_distribution == "Fedora"
  tags:
    - rpm_verify_hashes
    - low_severity
```

oscap コマンド例

パッケージ導入

```
# yum install openscap scap-security-guide
```

チェックリストの諸元確認

```
# oscap  info /usr/share/xml/scap/ssg/content/ssg-
rhel7-xccdf.xml
```

“common” プロファイルでのチェック実行

```
#oscap  xccdf  eval --profile common \
--results /tmp/results.xml \
/usr/share/xml/scap/ssg/content/ssg-rhel7-xccdf.xml
```

remediate スクリプトの生成

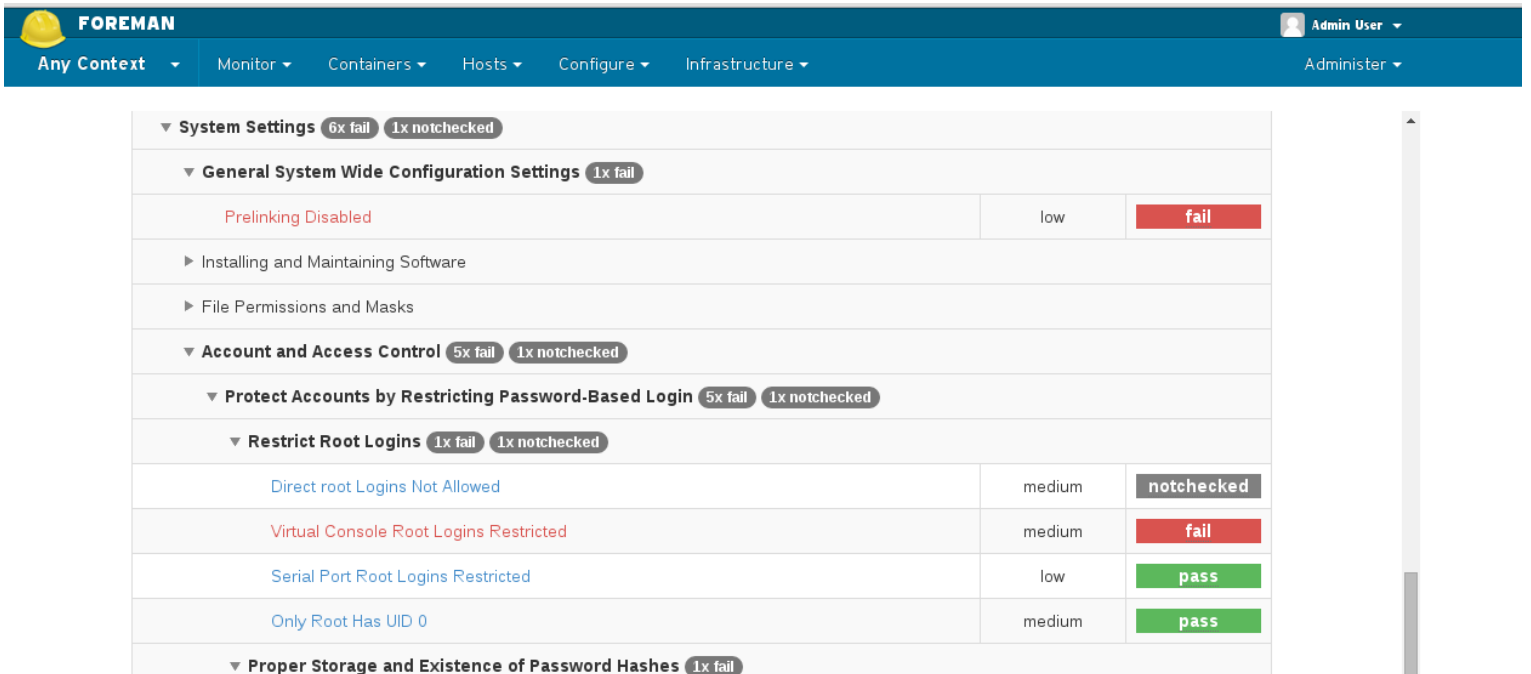
```
#oscap xccdf generate fix --fetch-remote-resources \
--profile common  --output /tmp/remediate.sh \
/tmp/results.xml
```

SSG にない独自のチェックは？

- OVAL でチェック方法を記述するのは大変
  - OVAL は XML でチェックを記述する文法
  - 「あるサービスが起動しているか？」だけで 55 行……
  - やりたいことが既存の OVAL リポジトリにないかを探す
    - OVAL のリポジトリ https://oval.cisecurity.org/repository
  - 独自のチェック追加や変更は Ansible 等で行う方が生産性が高そう
- XCCDF の基本は「名前」「説明」「 OVAL を呼ぶ」くらいなので XCCDF だけ勉強するのはアリ
- 専用エディタを使う
  - VMware Modified Enhanced SCAP Content Editor
    - https://github.com/vmware/vmware-scap-edit

複数台のスキャンは……？

- Spacewalk
  https://spacewalkproject.github.io/
- Foreman OpenSCAP plugin
  https://www.theforeman.org/plugins/foreman_openscap/0.9/
  - チェックリスト配布、 puppet で定期実行、レポート表示



まとめ

- **SCAP**: 基本的なセキュリティチェックを自動的に実行することを目的とした規格群
- **OpenSCAP**: OSS の SCAP 処理系
- **SCAP Security Guide**: SCAP 規格にもとづくチェックリストのひな型
- **SCAP Workbench**: チェックリストのカスタマイズを行いプロファイルを生成

「SSG から必要なところだけ SCAP Workbench で選択して OpenSCAP でチェックする」ことで基本的な確認作業の多くを自動化できる

Appendix

SCAP 関連リンク

- IPA の SCAP 概説
  - https://www.ipa.go.jp/security/vuln/SCAP.html
- RHEL ドキュメント「 Security Guide 」
  - http://red.ht/2yb6Zft (英語)
  - http://red.ht/2zps2yg (和訳)
- OpenSCAP プロジェクト
  - https://www.open-scap.org/
  - https://github.org/OpenSCAP/
  - 今回紹介以外のツール、詳しいドキュメント、各種チュートリアル
- OVAL のリポジトリ https://oval.cisecurity.org/repository
- OpenSCAP Scanning in Satellite 6 and CloudForms
  - http://red.ht/2AwBIFk
  - Red Hat の管理ツールとの連携について紹介

RHEL 特有の話題

- oscap-anaconda-plugin
  - RHEL や Fedora のインストーラ anaconda 用プラグイン
  - インストール時にスキャンを実施し remediation script を実行する
- Red Hat Satellite
  - Foreman をベースとする管理スイート
  - OpenSCAP の定期実行、結果を収集・蓄積・表示
- RHEL 同梱 SSG の対象ソフトウェア
  - RHEL6, RHEL7, JRE, Firefox