Overview

This talk represents a 2+ year investment in developing automated strategies and approaches for cyber hardening infrastructure of mission management systems.

Topics we will cover include:

– Mission Management System Commonalities
– But Security and STIGs are Hard!
– Why Automate?
– Multiple Programs – Multiple Solutions
– Our Solution – STIGLER
– Equal Press - Other Solutions
– Recommendations
Mission Management System Commonalities

Merges sophisticated cutting-edge mission planning software

To specialized, long-life (and often ancient) command hardware

Hardening Requirements

- 7x24 uptime,
- stable technologies & environments
- Cross Platform (Windows & Linux)

- Continual Patching
- Short Maintenance Windows
- Enterprise Scale (100s – 1,000s of hosts)

Mission Management Systems have Specialized Security Requirements
But Security and STIGs are Hard!

- So many products go into our Systems!
  - Everyone loves Commercial Off the Shelf (e.g. Oracle, RedHat, Microsoft)
  - Open Source is equally prevalent (Linux, Apache, MySQL and others)

- So many rules!
  - DISA now has over 400 STIGS or SRGs, with thousands of settings

- So many roles!
  - RMF calls for Role Based Access Control, meaning an Operator workstation has a different posture than network administrator workstation, even though the same products are installed.

- So many baselines!
  - Software patches & upgrades mean one or more baseline per machine role
  - Baseline-specific exceptions, exemptions, and risk profiles
  - With quarterly updates, and continuous integration the job is never done

- And then there’s the Cloud
  - How do you deploy to hundreds of servers on AWS or Azure?
Why did Raytheon Automate Hardening?

- **Speed**
  - Machines are faster than people
  - Support short maintenance windows for patches and updates.
  - Reduce build and checkout times for Complex environments from weeks to days

- **Cost**
  - Allows us to leverage senior Systems Administrators for multiple systems
  - Let machines do the simple stuff. Save the people for more challenging tasks.

- **Consistency**
  - **Human involvement** = human error
  - **Test as you build** – Adopt SW “Unit Test” approach to test a STIG setting in the same way that you set it
Our Approach

- Test on environments of hundreds or machines
  - Mix of Windows, Linux, COTS and Open Source (FOSS)
  - Mix of real and virtualized assets/Mix of Servers and Workstations
  - Hundreds of different Apps

- Demonstrate STIG Compliance
  - A “typical” baseline includes > 50 STIGS/SRGs and > 2,500 separate STIG Checks
  - Assume use of ACAS/Nessus for STIG verification (DoD standard)
  - Scripts can be used for “check out” of configuration

- Flex Multiple Deployment Scenarios
  - Periodic Patch Windows, Full Rebuild of environments, Point Rebuilds

- Automate the Humans
  - Develop detailed documentation so everyone knows what to do

Through Automation, we reduced deployment/checkout times from 3 weeks to 4 hours!!
Our Approach (2) – The details

- Collect Scripts from programs
  - Every SA has a “home grown” solution

- Trade Studies
  - Evaluated dozens of deployment, cyber and enforcement platforms

- Wrote Lots of Scripts
  - Learned that college interns excel at this!
  - Focused on DevOps
  - Learned how to manage Infrastructure Deployment scripts as Code
    - Lots of testing, peer reviews
    - CM managing and versioning these scripts

- Tested
  - Took advantage of DevOps and Cloud to deploy, test and repeat
Effectiveness

Infrastructure as Code

Treat those hardening scripts as Code!

We had to manage it as code, with CM (git, Clearcase), use development Methodologies (Agile) and SW Developers to develop custom hardening scripts (Chef, Ruby) and Security Enforcement Scripts (Nessus)

Scales VERY WELL – But lots of Work!

Enterprise Deployment Platforms –

Deploy via Chef, Puppet, IBM BigIron and Test with Nessus, ACAS, Security Center

Better – But don’t support multiple security baselines

COTS Solutions

~Windows Deployment Services for Microsoft & Kickstart & Scripts for Linux (RedHat)

OK – but didn’t help with hardening

Commonality

Reduced Deployment/Checkout times from 3 weeks to 4 hours

Nirvana! – Real-time conversion of STIGS to hardening code

Dozens of developers to write Bash, Ruby, Powershell and Nessus (Interns and SAs, actually)

Requires lots of customization and great SAs

Where most Programs are…
Our Solution – STIGLER

“To STIG” (v): the process of automatically deploying STIGs to the Enterprise and make System Administrator’s lives easier

- Ingest STIG Data from DISA
  - No manual input

- Export to Enterprise Tools
  - Powershell, Group Policy, Bash, Ruby (Chef), Nessus (Proprietary)

- Automated correlation of STIG rules with SCAP/OVAL data
  - Know the setting and desired value for each rule right away.

- Per-rule management
  - Modify or exclude rules quickly.

- Baselines Management
  - Exemptions and exclusions applied to a system role, not the whole system.
Equal Press - Other Solutions

- **The OpenSCAP Project** – a collection of Open Source Security Policies, tools and Standards

- **SIMP** – an open source fully automated framework, built on Puppet – released by the NSA

- **Chef Compliance** – assess an enterprise’s adherence to compliance requirements

- **SteelCloud** – “Config OS” – COTS product that Automates STIG deployment
Conclusion

Through use of automation, we were able to reduce rebuilds from weeks down to hours
- We leveraged automation to improve deployment products (Chef/Puppet) and STIG verification (ACAS)
- Adopted DevOps principles to deploy continuously to learn how to do it fast and repeatable
- Complemented automation with Build Orchestration Books ("BOBs") to make human actions repeatable
- Created scripts to harden COTS and FOSS Operating Systems, Application Platforms and Applications

Buy all the STIGLERs. They make great stocking stuffers.
Thank You!!

Co-Authors

Austin Garrett
Principal Systems Engineer
adgarrett@raytheon.com

Mike Worden
Engineering Fellow
michael.worden@raytheon.com