



Continuous Diagnostics and Mitigation (CDM)

CDM Training Modules

July 29th, 2014

Version 1.0



**Homeland
Security**

MODULE 3

CDM Implementation

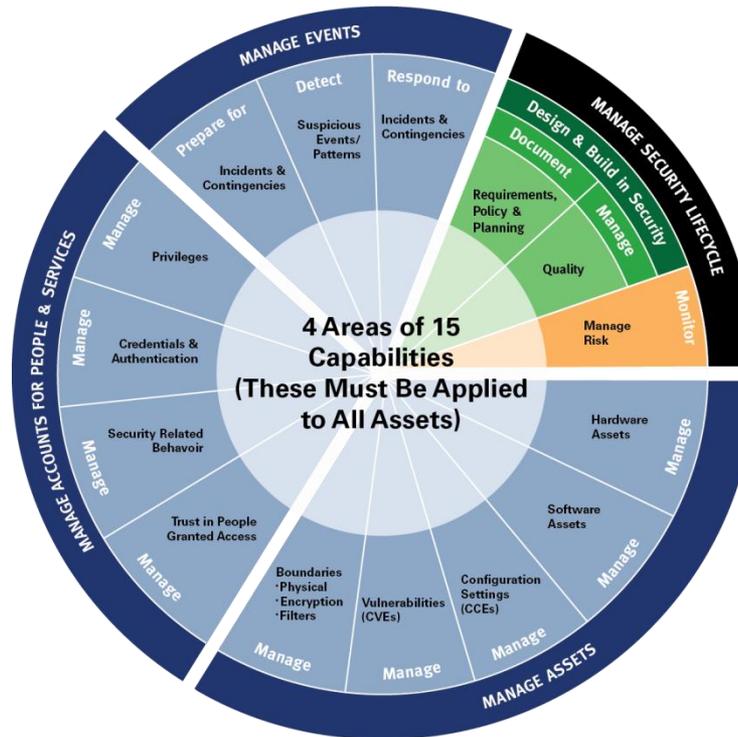


Homeland
Security

3.1



3.3 Software Asset Management (SWAM) Implementation



Topics to be Covered

1. Software Asset Management (SWAM) Definition
2. Making the Paradigm Shift
3. Developing a Desired State Specification
4. Actual State
5. Discussion of D/A Specifics
6. Summary



**Homeland
Security**

Learning Objectives

At the conclusion of this module, the participant will be able to:

1. Describe to management why the capability is important to the security of their networks.
2. Identify the typical steps necessary to be taken by the department or agency (D/A) to implement SWAM and manage software.
3. Identify D/A-specific steps/issues likely to affect SWAM implementation at their D/A.
4. Describe optional ways to achieve those steps and/or address issues.
5. Select the best set of options for implementation at their D/A.



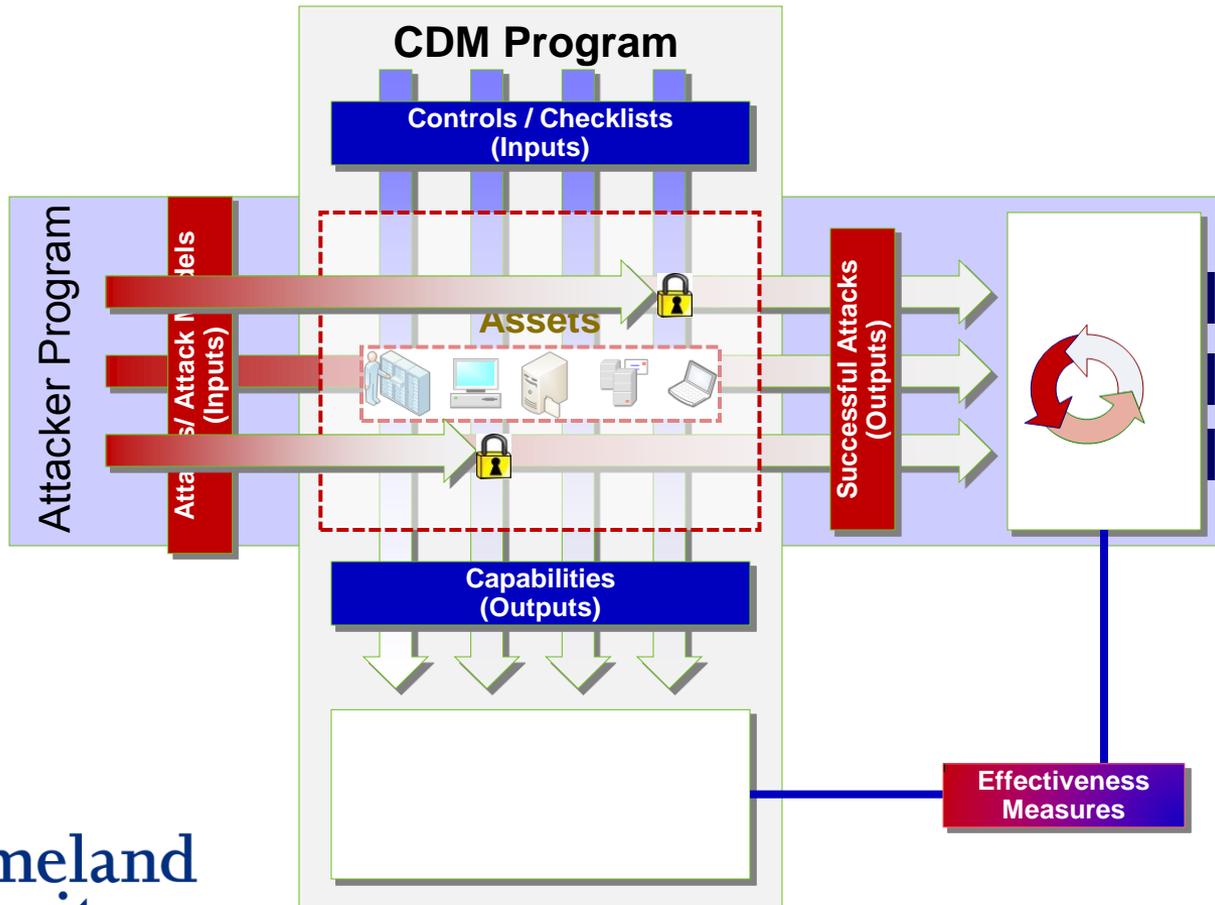
Topic: Software Asset Management (SWAM) Definition



**Homeland
Security**

Why Continuous Diagnostics and Mitigation?

How does CDM Work?



Homeland Security

Why SWAM?

Many cyber attacks today focus on the software used within an organization

- **APT - Advanced Persistent Threats**
 - An adversary that possesses sophisticated levels of expertise and significant resources which allow it to create opportunities to achieve its objectives by using multiple attack vectors (e.g., cyber, physical, and deception). These objectives typically include establishing and extending footholds within the information technology infrastructure of the targeted organizations for purposes of exfiltrating information, undermining or impeding critical aspects of a mission, program, or organization; or positioning itself to carry out these objectives in the future. The advanced persistent threat: (i) pursues its objectives repeatedly over an extended period of time; (ii) adapts to defenders' efforts to resist it; and (iii) is determined to maintain the level of interaction needed to execute its objectives.
- **Zero Day Exploits**
 - An attack on a piece of software that has a vulnerability for which there is no known patch



Why Advanced SWAM?

Advanced SWAM Techniques can block most APTs and Zero Day threats by

- Explicitly tracking “trusted” executables for approved SW; for example, by location, by hash or by certificate.
- Preventing other SW from executing.
- SW for APTs and Zero Day threats won't run because they are not on the approved list.
- Operationally, this is a maturity level that may take a few years to grow into.



SWAM CDM Capability Definition

SWAM addresses attacks that seek to exploit unauthorized and undetected malicious software.

It does this by:

- Identifying all software
- Determining whether the software is authorized and managed
- Taking appropriate action for the software that is not



Blocking unauthorized software can prevent many phishing attacks and zero day exploits.



SWAM is a CDM Capability

- A CDM capability is defined by:
 - Attack scenarios – the threat
 - Objects under attack
 - Concept of Operations (CONOPS)
 - Clarification of how the capabilities work together (differentiation)



SWAM Attack Scenario

- Unauthorized SW product/executable is on a device and vulnerable, placed by:
 - Malicious actor or agent
 - User
 - Administrator
 - Vendor, etc.

- Malicious Actors (threat source)
 - Search for and exploit unauthorized software
 - Leverage unauthorized software to compromise or harm systems and data, such as:
 - Gain control of target machines
 - Exfiltrate data



Software Assets in CDM

Definition: Software

Program or set of programs used to run a computer (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.46)

- **Software Executables***
 - An individual file made up of instructions to the CPU
 - Machine-level instructions or interpreted code
 - Sample of file types: .app (Mac OS), .bat (Windows), .cmd (Windows), .jar (Java), .ksh (Linux), .run (Linux)
- **Software Products***
 - A software product includes the set of software executables within a specific software release
 - May be commercially produced, open source, or locally developed at the D/A
 - A new patch or version is treated as a new product because it modifies its set of software executables
 - Example: Microsoft Word 2010, version 14.0.6129.5000

*Having the right patch level is covered in Vulnerability Management, not SWAM

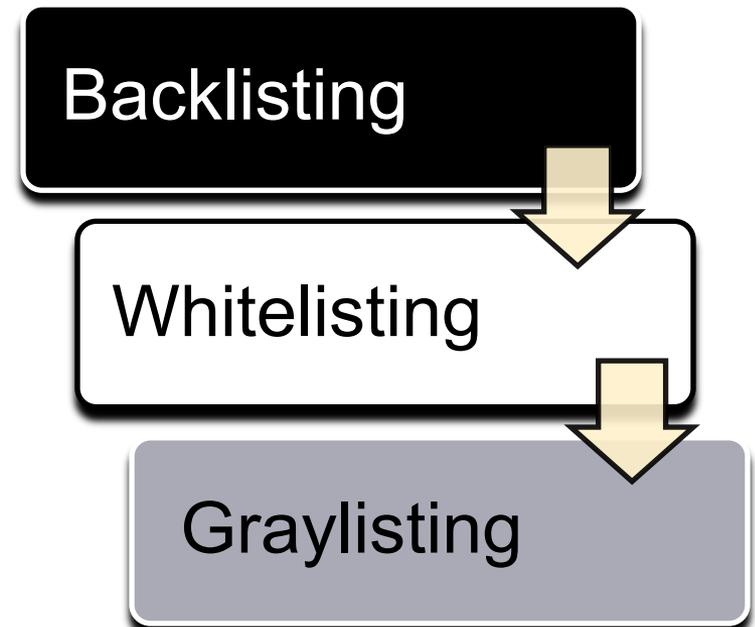


Software executables are the focus of SWAM.



SWAM CONOPS

- SWAM uses the following concepts:
 - Blacklisting
 - Software assets we don't trust
 - Whitelisting
 - Software assets we consider trustworthy
 - Graylisting
 - Software assets we don't know whether we can trust or not



Each software assets is identified as being on one of these lists



SWAM CONOPS

- Option 1 Blacklist Focus

Explicit Blacklisting

Weakest

Implicit Whitelisting

- Option 2 whitelist Focus

Explicit Whitelisting

Strongest

Implicit Blacklisting

*Flexible
Compromise*

- Option 3 Combined Focus

Explicit Blacklisting

Explicit Whitelisting

Implicit Graylisting



SWAM CONOPS

Federal Blacklisted Software Assets

Types:

1. Known bad by reputation
2. Known bad by behavior
3. No known business use
4. Product no longer supported (patching) by vendor
5. SW vendor NOT known to actively report security defects and promptly issue patches to fix them, for its supported products.
6. Not in an acceptable location and/or
7. Unsigned mobile code (including invalid certificates)



SWAM CONOPS

- **Blacklisted Software Assets**
 - Explicitly defined based upon prior criteria
 - Other products should be implicitly found on the whitelist or graylist
- **Examples**
 - Software threats detected by anti-virus programs
 - Known bad

Important point: If SW is on two lists, blacklist trumps whitelist

Weakness of blacklisting: We cannot possibly know all bad software



SWAM CONOPS

Federal Whitelisted Software Assets

Software asset that is not blacklisted and one or more of the following:

1. The agency has approved the SW and established a settings benchmark
2. There is a Federal USGCB, CDM, STIG, etc benchmark
3. Whitelisted by at least 3 CFO Act agencies
4. Used by at least 15 CFO Act agencies
5. Agency has established authorized installer accounts (with no internet or email access) for trusted/trained installers, and SW was installed by them, and subsequently reviewed and approved.

Use crowd Sourcing



SWAM CONOPS

- **Whitelisted Software Assets**
 - Products and executables that are explicitly listed or added based upon prior criteria
- **Examples**
 - Approved for use by the D/A and appropriate configuration settings applied
- **Ways to whitelist** 
 - By product
 - By executable hash
 - By location (i.e.: C:\program files\Microsoft Office
 - By who installed the executable
 - Other



SWAM CONOPS

Graylisted Software Assets

Software asset that is not blacklisted or whitelisted and one or more of the following:

1. Signed mobile code with valid certificate from source not explicitly trusted
2. Other SW not blacklisted and not white-listed



SWAM Scoring Example

		Settings Benchmark	
	Base Score	No Benchmark and no Assessment	Benchmark Present but no Assessment
Blacklisted	100%	N/A	N/A
Graylisted	10%	Add 30%	Add 15%
Whitelisted	0%	Add 30%	Add 15%



SWAM CONOPS

Search for and identify all
software

Collect Actual State

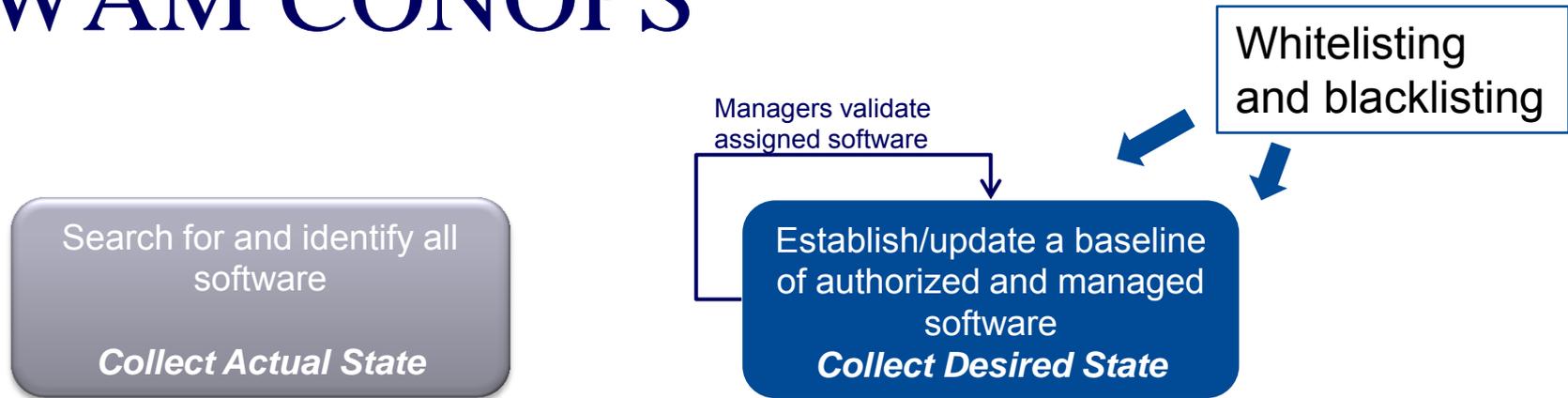


Mostly CMaaS Responsibility

- ✗ Often done, by product for part of the installed software, but seldom done for all.**
- ✗ Seldom done at the executable file level (.exe)**



SWAM CONOPS

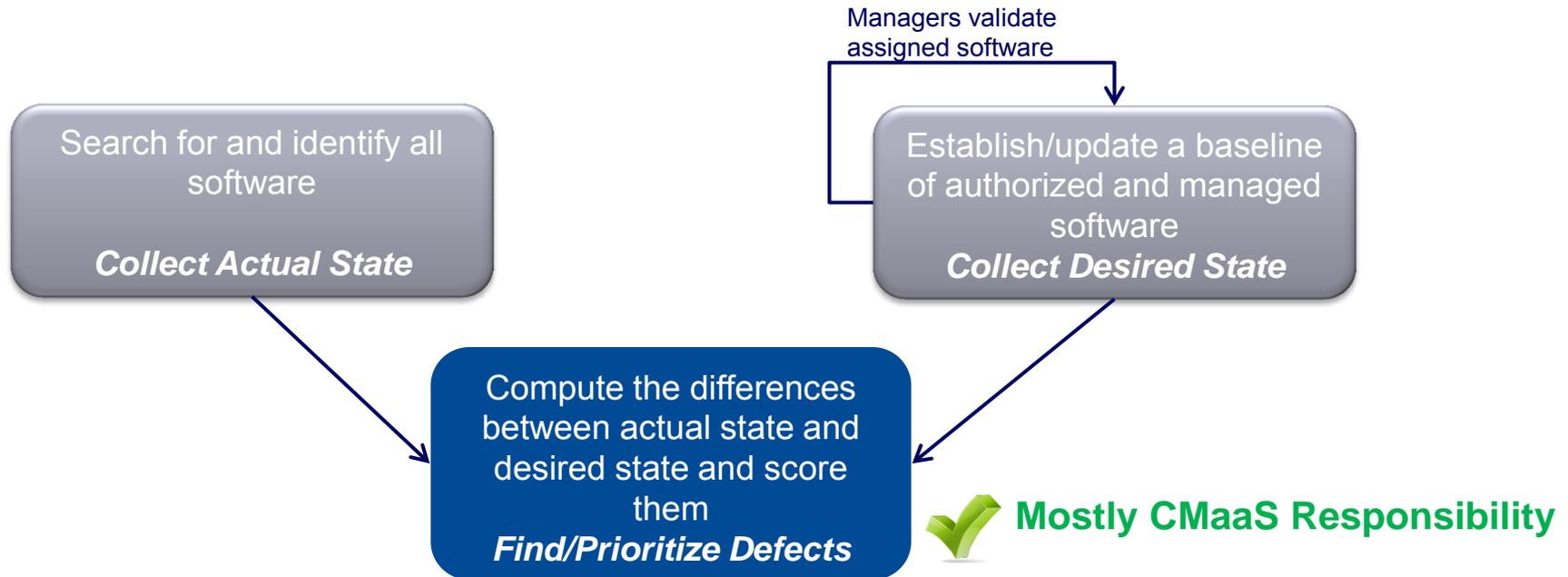


- ✗ This step is seldom done**
- **At all (except for Anti-virus protection)**
 - **Or it is not current**
 - **Or not in a form that can be automatically compared to actual state**

- ✓ Mostly D/A responsibility**
CMaaS provides repository



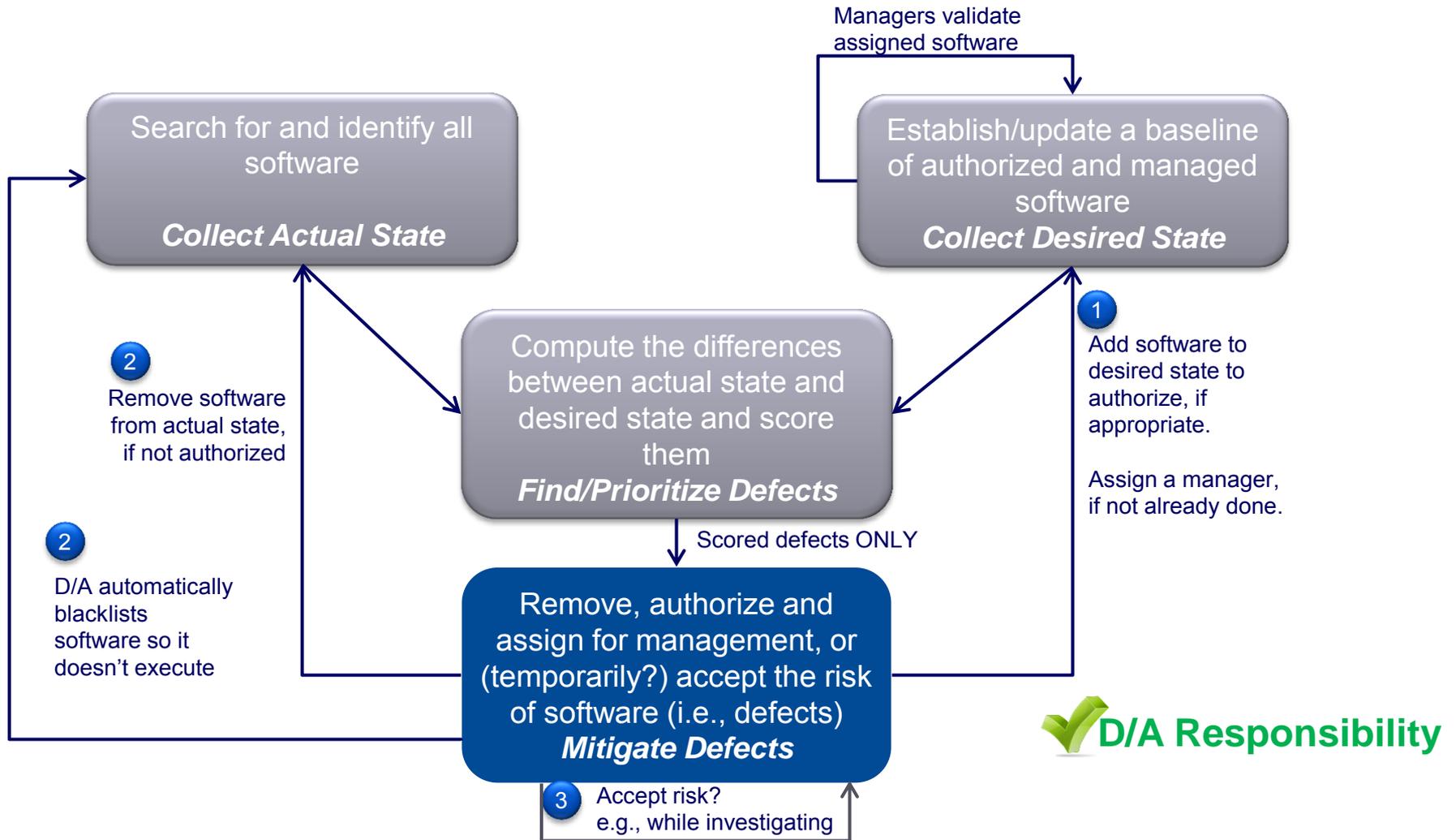
SWAM CONOPS



 **When both Actual and Desired State are automated, timely, and comparable, we can easily compute differences, which represent unauthorized software.**



SWAM CONOPS

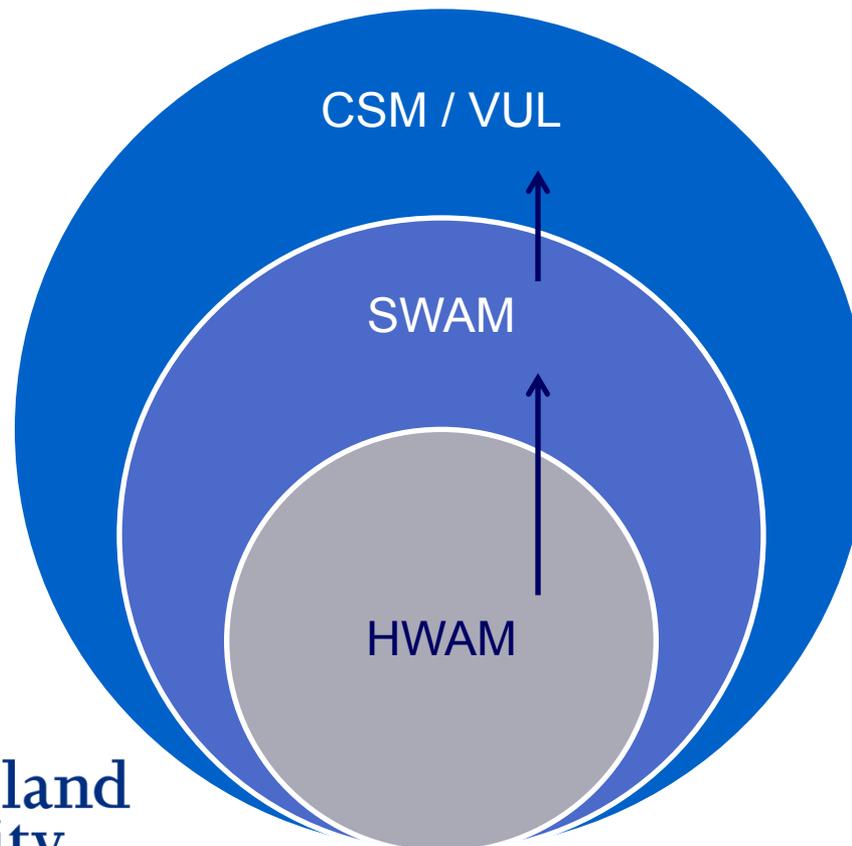


 **Then we can take the appropriate action**



How the Capabilities Work Together

- Hardware Asset Management (HWAM) and Software Asset Management (SWAM) support CSM and VULN by providing a reliable specification of hardware and software assets to check for known issues.



- CSM tells you which settings your software should have.
- VULN tells you what updates your software needs to have.
- SWAM tells you what software are present
- HWAM tells you where to look for SW



How the Capabilities Work Together

- ✓ ■ SWAM makes sure software is
 1. Identified
 2. Authorized
 3. Managed

- ✗ ■ You can't authorize software unless you identify it and make a decision on its trustworthiness.

- ✗ ■ You also can't expect it to be managed if someone isn't assigned that responsibility.

- 💡 ■ SWAM does not address how well software on the device is managed, but only that the software is authorized. How well the software is managed is covered by configuration settings management (CSM) and vulnerability management (VULN).



How the Capabilities Work Together



SWAM makes sure software is

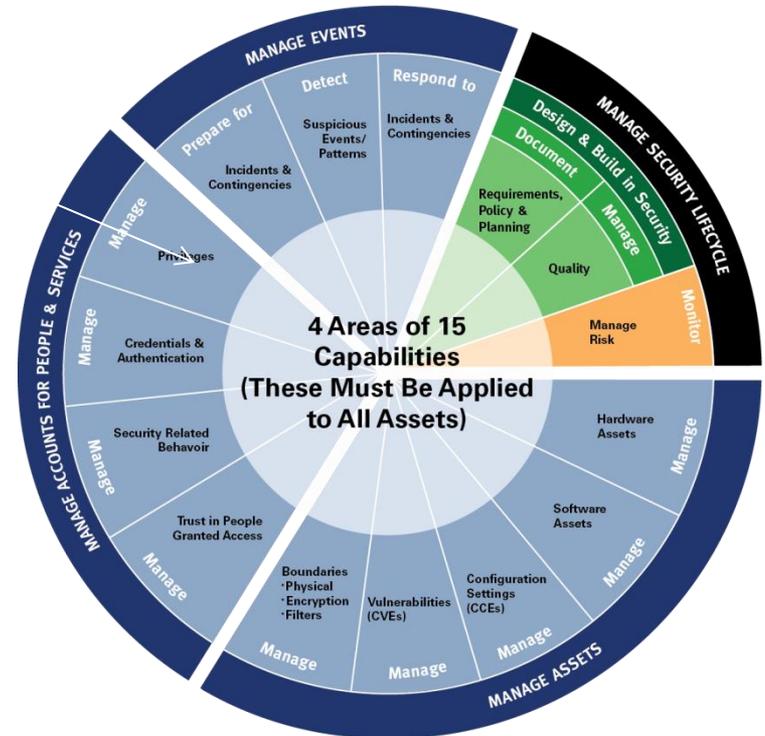
1. Identified
2. Authorized
3. Managed



You can't authorize software unless you identify it and make a decision on its trustworthiness.



You also can't expect it to be managed if someone isn't assigned that responsibility.



Well-Managed vs. Unmanaged

Well-Managed

- Product white/blacklisted
- Authorization process
- Patches up-to-date
- Settings authorized

Unmanaged

- Graylisted
- Product not approved
- Patches out-of-date (**VULN**)
- Unauthorized settings (**CSM**)

- Provides a view of software management responsibility
 - who decides what versions are allowed for the organization **SWAM**
 - who configures the software **CSM**
 - who patches the software **VULN**



SWAM Can Prevent Some Phishing Attacks

“By far, phishing attacks constitute the vast majority of attacks on federal and private sector networks.”¹

- Phishing attacks rely on ability to launch unauthorized and unmanaged software
- Whitelisting limits software the attacker can execute
- Mobile code management
 - Done via current digital certificates issued from a trusted Certificate Authority (CA)



Exercise

- **Describe to management why the capability is important to the security of their networks.**
 - How is my organization compromised because of unauthorized and unmanaged software:
 - We don't know?
 - Mostly by software that was unmanaged?
 - Mostly by unauthorized software being put on the network maliciously?
 - Can your organization easily find and report 99% of the software on your network(s)?
 - Is it easy to find out who manages the software?
 - Do you have a process to know when software should be removed?



Mitigate Defects

- The following list shows the most important defect types and mitigation options for SWAM*:
 - Unauthorized software
 - Device Role Policy Violation
 - Blacklist is out of date
 - Non-reporting

* The full set of Defects and mitigations are documented in the *Software Asset Management Datasheet* at <http://www.us-cert.gov/cdm>.



**Homeland
Security**

Mitigate Defects

Unauthorized Software

 Defect

 Mitigation

Defect Type	Detection Rule	Response Options
Unauthorized Software	In Actual State but not in Desired State	<ul style="list-style-type: none"> Remove software Authorize software OR Accept risk



Mitigate Defects

Device Role Policy Violation

 Defect

 Mitigation

Defect Type	Detection Rule	Response Options
Device Role Policy Violation	Actual State less secure than Desired State	<ul style="list-style-type: none"> • Remove device from incompatible device role, OR • Update policy, OR • Accept risk



Mitigate Defects

Blacklist is out of date

 Defect

 Mitigation

Defect Type	Detection Rule	Response Options
Blacklist is out of date	Actual State less secure than Desired State	<ul style="list-style-type: none"> • Update the blacklist for the device ,OR • Restore updating process, OR • Remove device, OR • Accept risk



Mitigate Defects

Non-reporting

 Defect

 Mitigation

Defect Type	Detection Rule	Response Options
Non-reporting	Actual State data unavailable	<ul style="list-style-type: none">• Deploy collection capability, OR• Restore collection, OR• Remove device, OR• Accept risk



Other Aspects of Software (Not the Focus of SWAM)

- Software suite:
 - A collection of products from one vendor than can be purchased as a unit
 - Might be the only way the get the products
 - The suite might be installed as a unit
 - The products are maintained (patched) individually
 - E.g., Microsoft .Net
- Documentation
 - Electronic files describing the software product and its use
 - E.g., Users manuals, installation guides
- Data files
 - Input/output for the software executables
 - E.g., Databases, raw input
- Non-interpreted source code
 - Languages that are not interpreted
 - E.g., C++ source code, Java source code



Other SWAM Data about Software

Discussion:

- In addition to vendor, name, version, and update, what other software data could be collected?
 - Accountability
 - User(s)
 - Installer(s)
 - Owners(s)
 - Supply chain
 - Licensing data
 - Producer(s)
 - Distributor(s)
 - Settings (may support or be collected by other capabilities)
 - Frequency of use
 - Others?



Exercise

Identify the typical steps necessary to be taken by the D/A to implement SWAM and manage software.

- Can you explain to your management and staff how SWAM works?
 - What are the main CMaaS contractor roles?
 - What are the main D/A roles?
 - How is it different from your current CONOPS?
- Can you explain what risks SWAM protects against and how this compares to your existing system?
 - Knowing what software needs to be managed?
 - Knowing who manages it, so you can keep a list of defects they need to address?
 - Getting those defects mitigated quickly?
- Do all defects found have to be fixed immediately?
- How does risk acceptance work?
- Can you explain why SWAM doesn't deal with how well the software is managed? What does?



Topic: Making the Paradigm Shift



**Homeland
Security**

Paradigm Shift

Definition: Paradigm

A set of assumptions, concepts, values, and practices that constitutes a way of viewing reality for the community that shares them, especially in an intellectual discipline.
(<http://www.ahdictionary.com/word/search.html?q=paradigm>)

Definition: Paradigm Shift

A significant change in the paradigm of any discipline or group.
(<http://dictionary.reference.com/browse/paradigm+shift>)



- Paradigm shifts drastically change the way a subject is approached
- **CDM (and ISCM generally) requires a paradigm shift for information security to allow automation**
- Most (if not all) paradigm shifts encounter resistance from those heavily invested in the old paradigm



Paradigm Shift

Move from
Managing Products
to Managing
Executables



Microsoft Office Suite



winword.exe



How are software executables and products identified?

Software Executables

- Identified by, at a minimum:
 - Executable name (e.g., nameext.dll); and
 - Digital fingerprint (i.e., hash)
- Methods:
 - Compare or match digital fingerprints (or Software Identification [SWID] Tags containing digital fingerprints) to hash library

Software Products

- Identified by, at a minimum:
 - Vendor name (e.g., Microsoft);
 - Product name (product licensed, e.g., Word 2010);
 - Version number, including patch (e.g., 14.0.4763.1000); and
 - Update (e.g., SP 1)
- Methods:
 - Common Platform Enumeration (CPE)
 - SWID Tag
 - Others?



Trust Library

- CMaaS tools have a “Trust Library” for software executable fingerprints
 - The Trust Library may include some pre-whitelisted items (those known to be trustworthy)
 - Custom software must be added to the Trust Library manually



Information in the Trust Library

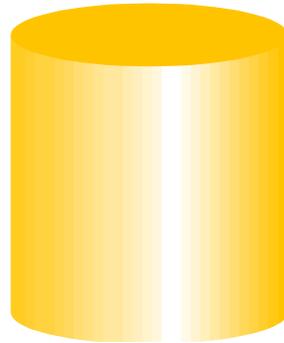
- General information
 - Designated listing (white, gray, or black)
 - Designated by
 - Date designated
- Executable, identified by:
 - Executable name
 - Digital fingerprint
- Product, identified by:
 - Product vendor
 - Product name
 - Version
 - Both build version and licensing version help identify the product
 - Build version is more important; it tells how vulnerable the product is
 - Release
 - Patch level
 - CPE ID, if applicable
 - SWID Tag, if applicable



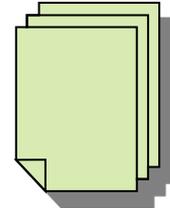
Locational Whitelisting



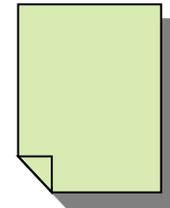
Hard Drive



File System
(C Drive)



C:\Program Files*.*



C:\Temp



- All whitelisted software is defined in approved paths
- Easier to implement, alternative to hash
- Only approved installer accounts have access
 - No email or Internet access by these accounts



Role-based Listing Considerations

**DETERMINED
TRUSTWORTHY**

**TRUSTWORTHINESS
UNDETERMINED**

**DETERMINED
UNTRUSTWORTHY**

	Whitelist	Graylist	Blacklist
Needed for Role	<ul style="list-style-type: none"> Trusted and allowed 	<ul style="list-style-type: none"> Temporarily allowed Move to whitelist or blacklist over time (high priority) 	<ul style="list-style-type: none"> Allowed if need overrides risk (risk accepted) Will require security mitigation(s)
Not Needed for Role	<ul style="list-style-type: none"> Not allowed (even if trusted) 	<ul style="list-style-type: none"> Not allowed Move to whitelist or blacklist over time (low priority) 	<ul style="list-style-type: none"> Not allowed

 **D/As define their own listing policies.**

Creating the Lists (White, Black, and Gray)

- Start with a D/A's software configuration control board (CCB) specification
 - Pros
 - Good source of authorized software
 - Members typically represent each department and consider their various missions
 - Cons
 - Takes too long to make ongoing decisions
 - Labor intensive
 - Often does not work small details such as:
 - Patches
 - Drivers



Whitelist Approvals

Globally

- Example: Every device can run Microsoft Word
- Pros:
 - Less administrative costs (\$)
 - Fast/easy approvals
- Cons:
 - Allows software to run on devices where it is not needed
 - May propagate problems across the network

Allow too much Software

By Device/User

- Example:
 - Jane Smith can run Microsoft Word
 - Server ABC can run Apache
- Pros:
 - Uniquely tailors environment for everyone's needs
- Cons:
 - Prohibitive administrative overhead (\$\$\$\$\$)

Too Specific

By User Role

- Example: All financial analysts can run Microsoft Excel
- Pros:
 - Limits the use of the software to those who need it
 - Relatively low administrative costs (\$\$)
- Cons:
 - Allows software to run on devices where it is not needed

By Device Role

- Example: Database servers can run Oracle database software
- Pros:
 - Limits the use of the software to the devices that need it
 - Relatively low administrative costs (\$\$)
- Cons:
 - Allows users to run software that may not be needed

Combination of User Role and Device Role

- Example: All database administrators can run Oracle database software on database servers
- Pros:
 - Limits the use of the software to those users and devices that need it
- Cons:
 - Moderate administrative costs (\$\$\$)



Software Installer

- Software installer will differ amongst D/As and could be:
 - Person or a service account
 - The device manager as defined under HWAM
 - Enterprise Configuration Management Tool
 - Application Manager
 - ~~End User~~
 - Determination based on privileges
 - Other?

Software Install Attributes

- No email from system
- No Internet from system
- Account is only used to install software



Moving Software from the Whitelist

- Use systematic process for un-authorizing software, e.g.:
 - Sun-setting
 - Software is approved for a specified period of time
 - After the specified period, the software is removed from the whitelist
 - Grace period for outdated version or unsupported product
 - Software is authorized to run until a specified date
 - After the specified date, the outdated version is removed from the whitelist
 - Critical vulnerability is identified
 - Software is temporarily removed from the whitelist until a patch is released and applied
 - Other?

- As software is un-authorized, it is moved to the blacklist



Strive for a Small Graylist

- The graylist should be processed and reduced towards zero
 - Could take months /years
 - D/A should staff it to the size needed to achieve this
 - Start with a repository from the CMaaS provider with trustworthy and untrustworthy software identified (de facto standard)
 - Priority:
 1. New software
 2. Legacy software
 - a. Start with common software
 - b. End with least used software
- If software is dormant (not needed or used), D/As may choose to remove it entirely



New Software



- Decide who or what is authorized to install new software
- All new software installed by an approved installer is either graylisted or whitelisted
 - D/A policy decision
- All new software installed by an unauthorized individual is blacklisted
 - D/A policy decision



Mobile Code

Definition: Mobile Code

Mobile code is software transferred between systems and executed on a local system without explicit installation by the recipient. (http://en.wikipedia.org/wiki/Mobile_code)

- Currently, no good method to verify that the executable code received matches the intended function
 - For now, SWAM can:
 - Allow/disallow mobile code to run
 - Allow only digitally signed mobile code to run
 - Allow code from trusted sources to run
 - Monitor how often mobile code runs if not allowed
 - Assign risk scores appropriately
- The Metrics Working Group (MWG), will decide if/how to monitor mobile code



Common Platform Enumeration (CPE)

Definition: Common Platform Enumeration (CPE)

A structured naming scheme for information technology systems, software, and packages. Based upon the generic syntax for Uniform Resource Identifiers (URI), CPE includes a formal name format, a method for checking names against a system, and a description format for binding text and tests to a name. (NIST, <http://nvd.nist.gov/cpe.cfm>)

- CPEs are typically identified through:
 - Identify the executable and deduce the CPE from the executable
 - Registry settings
 - Software Identification (SWID) Tag
- CPEs are used to identify software products
 - An example is the following name representing Microsoft Internet Explorer 8.0.6001 Beta:

```
wfn:[part="a",vendor="microsoft",product="internet_explorer",  
version="8\0\6001",update="beta"]
```

[Click here to return to "How are software executables and products identified?"](#)



Software Identification (SWID) Tag

Definition: Software Identification (SWID) Tag

Software ID tags provide authoritative identifying information for installed software or other licensable item (such as fonts, or copyrighted papers).

(http://en.wikipedia.org/wiki/ISO/IEC_19770)

- A SWID tag on a computing endpoint (device) provides a high degree of proof that the product is actually installed¹
- SWID tags are typically created/modified:
 - By the software product vendors
 - When software products are installed
 - When software products are patched

■ Sample SWID Tag

[Click here to return to "How are software executables and products identified?"](#)



**Homeland
Security**

¹Cheikes, B., The MITRE Corporation, Auditing and Remediating with CPE Names and Software Identification Tags, 12 January 2012.

Digital Fingerprints

Definition: Software

Digital fingerprinting is the identification of large data files or structures using a hash function. These functions change a larger data set, sometimes known as a key, into a shorter data set, which may be called a hash. (adapted from <http://www.techopedia.com/definition/23370/digital-fingerprinting>)

- Digital fingerprints are typically:
 - Used to identify changes to software executables
 - Generated by a third-party for management purposes
 - Stored in a “hash library”
- Cryptographic hash function, e.g.:
 - MD2 ▪ SHA-0
 - MD4 ▪ SHA-1
 - MD5 ▪ SHA-256/224
- Digital fingerprints come from:
 - CMaaS provider for COTS products
 - D/A for custom products
 - Product vendor (SWID Tag)

[Click here to return to "How are software executables and products identified?"](#)



Using SWID tags and Digital Fingerprints to Manage Supply Chain

- Combination of SWID tag and digital fingerprints can validate supply chain integrity
 - A SWID tag can include the digital fingerprints
 - Validate supply chain by:
 - Computing current hash value of the software executable
 - Comparing the computed value to the original hash value in the hash library
- Supply chain management needs to be worked with the software producer and can be facilitated by the D/A's CMaaS provider

[Click here to return to "How are software executables and products identified?"](#)



CPE vs. SWID Tag

CPE	SWID Tag
Both are currently viable options for tracking software.	
Limited to naming the software products	Could name and document all of the executables that make up a software product
Slightly ahead of SWIDs in market adoption in the U.S	Less traction in the U.S.
NIST standard	International standard

[Click here to return to "How are software executables and products identified?"](#)



Anti-Virus versus SWAM

Definition: Anti-virus

A generic blacklisting tool that is looking for software determined to be untrustworthy.

- Anti-virus can be:
 - A tool to partially implement SWAM
 - May or may not be necessary in the future with a tightly controlled whitelist

- SWAM also includes
 - Whitelisting (authorization)
 - Blacklisting (prohibition)
 - Identifying software administrator (management)
 - Identifying software to be configured and patched



Patches/Versions Effect on Software Products and Executables

- Patches and new software versions change the installed software product and its executables
- In CDM, a different software product version needs to be monitored
- A CPE is created when new software versions are installed, but not patches

cpe:/a:adobe:acrobat:10.0
Vendor: adobe
Product: acrobat
Version: 10.0
View CVEs
cpe:/a:adobe:acrobat:10.0.1
Vendor: adobe
Product: acrobat
Version: 10.0.1
View CVEs
cpe:/a:adobe:acrobat:10.0.1::-:pro
Vendor: adobe
Product: acrobat
Version: 10.0.1
Revision: -
Edition: pro
View CVEs
cpe:/a:adobe:acrobat:10.0.2
Vendor: adobe
Product: acrobat
Version: 10.0.2
View CVEs
cpe:/a:adobe:acrobat:10.0.3



Exercise

Identify how the typical SWAM steps may differ from older ways to managing software.

- Why does SWAM require a desired state specification?
 - Why does it need to be automated?
 - How does the desired state specification relate to what software is authorized and who manages it?
- What is a Software Manager?
 - Is that different from a software owner or user?
 - Why does SWAM require each software to have a manager?
 - Is the software manager better thought of as a person or a group in my organization?
- Why is scoring to prioritize defects an important part of CDM (including SWAM)?



Topic: Developing a Desired State Specification



**Homeland
Security**

Desired State Specification

Definition: Desired State Specification

A listing of each authorized software and its managers.

- **Oops!! What if my D/A doesn't have a complete desired state? Won't getting one be an incredible amount of work?**
- Actually, no. it can evolve naturally out of the actual state, as the D/A
 - Identifies its actual software
 - Adds them to desired state, grandfathering them
 - Assigns appropriate governance roles for each new future software product/executable
 - Works to find known bads and known goods in the grandfathered software, over time
 - Should be clean by the next refresh of each device

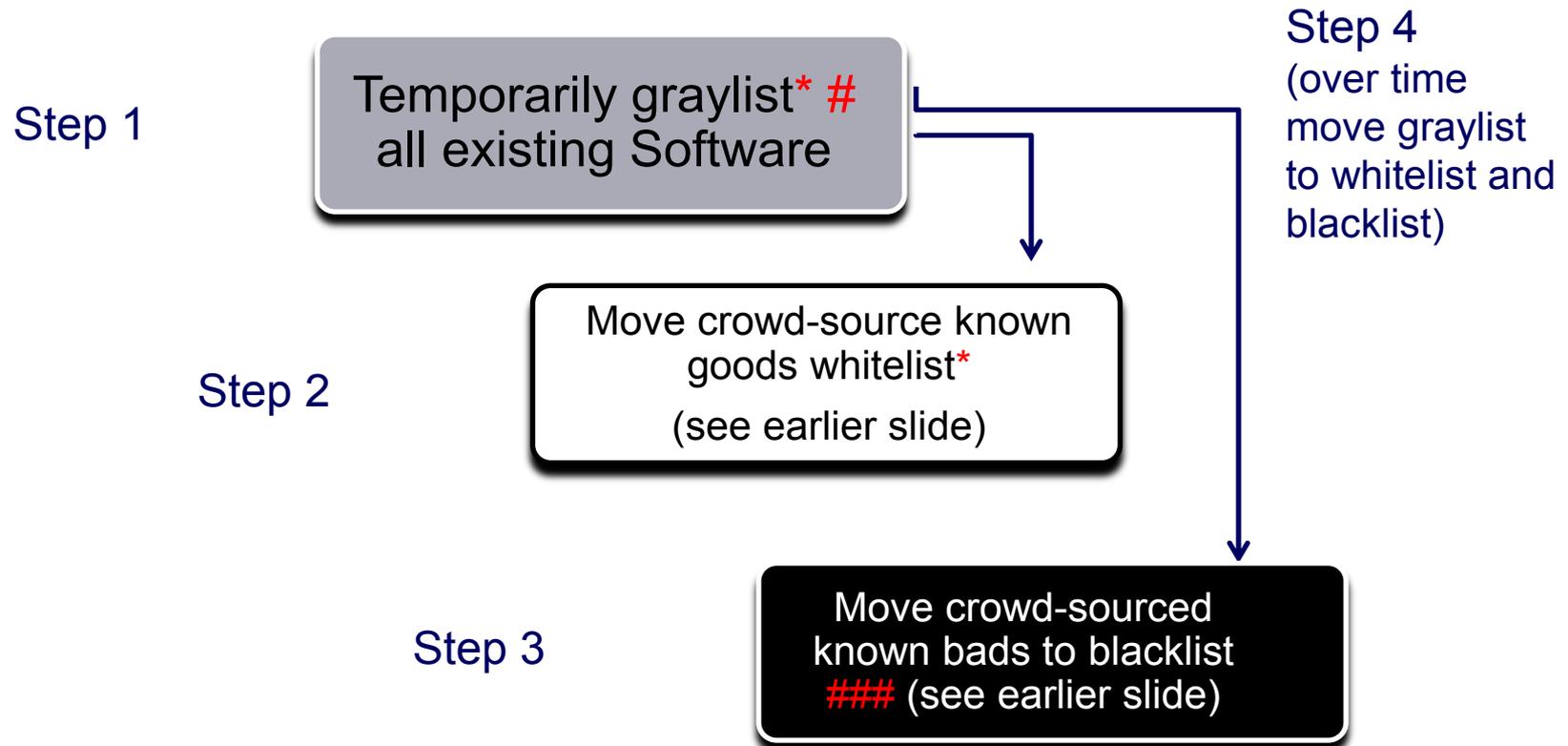


How can my organization do this?



**Homeland
Security**

Initiate Desired State - Legacy Software



* Allowed to Run # Incurs Risk if it runs



Initiate Desired State - New Software

Step 1

Authorized Installer adds Software to whitelist*

Decide to move to graylist* #

Step 2a

or

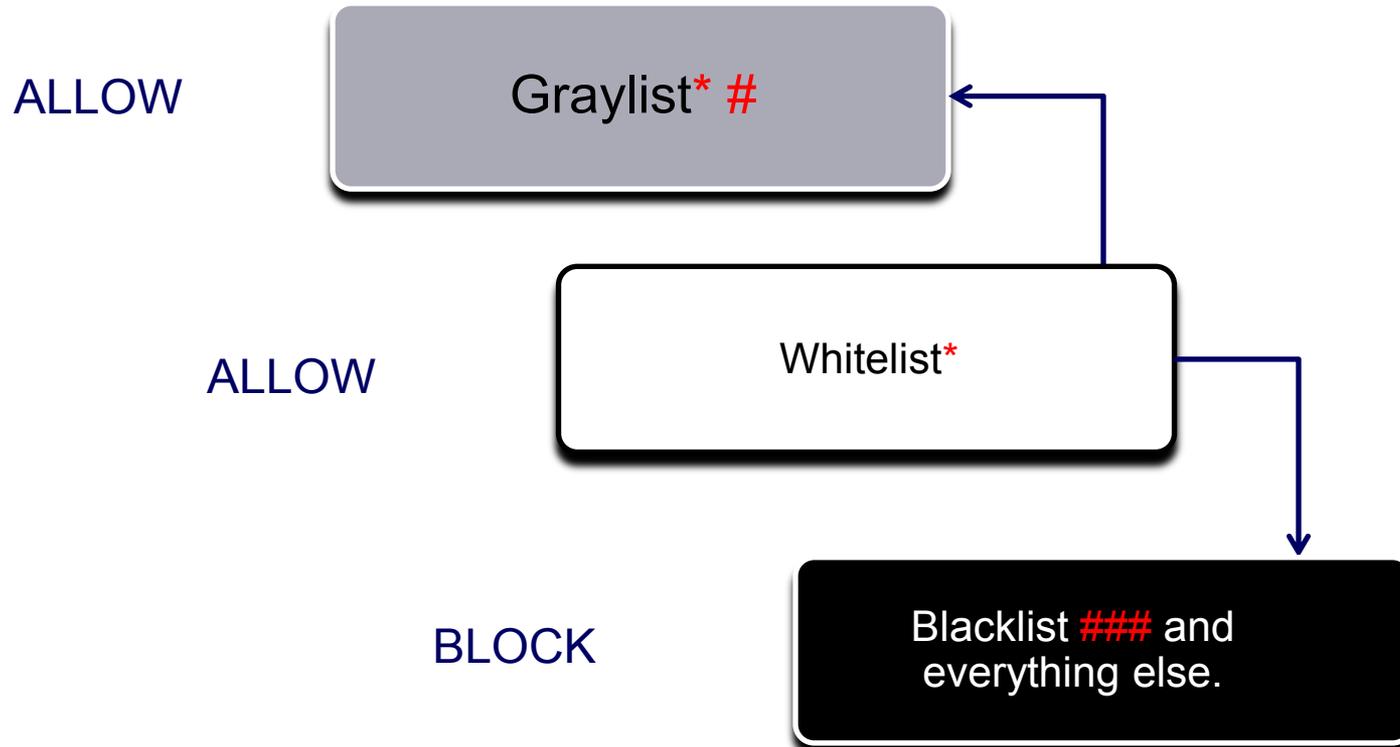
Decide to move to blacklist ###

Step 2b

* Allowed to Run # Incurs Risk if it runs



Initiate Desired State – At Run Time



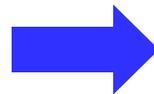
Incurs risk if it runs

New APTs and Zero-days will not be white or graylisted. They are “implicitly” blacklisted. So they won’t run.



Initiate Desired State – Mobile Software

- Keep a reputational list of trusted sources
 - Block mobile from running
 - If not from a trusted source
 - If not signed with a current/valid digital certificate



Reputational List



Approved?



Denied!



Initiate Desired State – Blocking

- If possible have a trust repository
 - For each executable
 - That belongs with each version/patch level of product
 - Sources
 - Trust Library
 - SWID
- Block Scenarios
 - Block software not matching the sources (trust library or SWIDs)
 - Block all software not on the whitelist or graylist from running



Exercise

Identify the typical steps to be taken by the D/A to implement SWAM and manage software. How do we get started?

- Why are software identifiers critical to SWAM operations?
- Why do you need to know about software identifiers, if the CMaaS provider is largely responsible for deciding which to use?
- Which software identifiers might work best given how your network is managed?
- What are the standard desired state data elements about software?
- Are there other DESIRED state data elements my organization might need?
- What way might be the best for assigning software managers in your D/A?



Topic: Actual State



**Homeland
Security**

How to Determine the Actual State

- SWAM scans run at least every 3 days
- Several methods for determining Actual State:
 - Registry/configuration files
 - Scanners (authenticated scan)
 - Agents (software on the systems)
 - Vendor publishing (e.g., SWIDs)
 - Vendor publishing plus the agent



Methods to Determine Actual State

Registry/Configuration Files

- Description: List of configuration parameters for each software product installed
- Pros:
 - Common to all operating systems
- Cons:
 - Software listed may not be on the machine
 - Software on the machine may not be listed
 - May include unaccounted malware

Scanners

- Description: A tool that remotely inventories and/or inspects a device for specific information.
- Pros:
 - No software is needed on an endpoint device
- Cons:
 - Puts load on the network
 - Scanners may exist on the network that are not legitimate and authentication burdens the network

Agents

- Description: Tool installed on end point device to specification/inspect a device for specific information
- Pros:
 - Not dependent on a continuous network connection
 - Can block software from running
- Cons:
 - Puts load on the device

Publishing

- Description: Identification embedded into the software when published by the vendor
- Pros:
 - State of software at install time
 - Method to continuously check trust library
- Cons:
 - Needs automated software identification mechanism
 - SWID Tag must be generated if one is not supplied

Combination of Agents and Publishing

Description: A software specification checked against purchasing and licensing records

Pros:

- Ties together the information from installation to current status
- Not dependent on a continuous network connection
- Can block software from running
- Method to continuously check trust library

Cons:

- Puts load on the device
- Must be installed and managed on endpoint device
- Needs automated software identification mechanism
- SWID Tag must be generated if one is not supplied



Acceptable Rates for False Positives and Negatives

Definition: False Positive

Report of software asset that does not exist.

Definition: False Negative

Failure to report actual software that does exist.

- All CMaaS products are contractually obligated to produce actual state inventories with:
 - False positive rate below 0.1%
 - False negative rate below 0.1%
- D/As will monitor this ratio using a CDM-provided tool
 - To ensure the D/A workload responding to defects is minimized and
 - To ensure the contractor is adhering to its performance specification



Exercise

Identify the typical steps to be taken by the D/A to implement SWAM and manage software. How do we get started?

- How does your organization monitor performance on:
 - Your network?
 - Individual devices?
- How can your organization use instances when CDM is operating (and when it is not) to estimate the network impact of CDM data collection tools?
- Can you hold the CMaaS contractor responsible to not interfere significantly with such performance?
- Why does the CMaaS contractor need “authentication credentials” to collect data? Who must provide those credentials?
- Why does the CMaaS contractor need an appropriate path through firewalls to collect data? Who must provide those paths?
- Who decides whether the CMaaS tools are safe on our network?
- Who assesses the CMaaS contractor tools?



Activity: Discussion of D/A Specifics

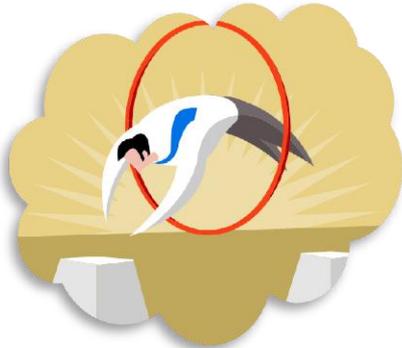


Homeland
Security

How will SWAM

Implementation look in your D/A?

- Discussion:**
- Discuss SWAM implementation as it will apply in a given D/A
 - What capabilities already exist?
 - Will/can they be integrated?
 - What information does the CMaaS provider need to begin implementation?
 - Identify any issues of concern
 - What political challenges need to be overcome?
 - What operational/organizational challenges need to be overcome?
 - What other D/A-specific issues may impact the ability of the CMaaS provider to implement SWAM?
 - Discuss recommendations



SWAM Responsibilities:

CMaaS Contractor

- Determine the actual state using software sensors
- Compare differences between actual state and desired state
- Prioritize defects based on federally approved algorithm
- Maintain the SWAM sensors
- Adhere to performance specifications
- Establish and maintain
 - A Trust Library with digital fingerprints for COTS products
 - Digital fingerprints for custom and other non-COTS software
 - The white, black, and graylists according to D/A listing policies
 - Reporting Groupings
- Support the D/A decision process and make recommendations as appropriate
 - E.g., Recommend the D/A should decide to put all new software on the graylist



SWAM Responsibilities: D/A and CDM PMO

- D/A
 - Define white, black, and graylist policies, e.g., whether:
 - New software is automatically white or graylisted
 - Graylisted software is allowed to run
 - Establish and maintain the desired state specification
 - Assign software to white, black, or graylist
 - Reduce the size of the graylist over time
 - Mitigate defects
- CDM PMO
 - Provides training and mentoring
 - Organizes CDM acquisition(s)
 - Provides oversight and assistance to D/As who are unable to resolve unsatisfactory contractor performance



Impact of SWAM Sensors on Network and Device Performance

- It is the CMaaS contractor's responsibility to measure the impact of SWAM sensors on:
 - Network performance
 - Device performance
- Methods to measure impact could include:
 - Shutting down CDM for a short period of time and compute the difference in performance

Discussion:

- What other methods could be used to measure impact?
- How can the CMaaS contractor minimize the impact of SWAM sensors on network/device performance?



Navigating Secure D/A Networks

- Firewalls, network segmentation, different security policies, etc. pose network navigation challenges for software discovery tools
- The CMaaS provider must work with the D/A to determine how software discovery tools may navigate secure networks, asking and addressing questions such as:
 - What permissions are required?
 - Who is the POC(s) for obtaining permissions?
 - Where on the network should the software discovery tools be located to minimize network navigation challenges?
 - Others?
- D/As are responsible for working with the CMaaS contractor to ensure they have the information needed to successfully navigate the D/A network



Reporting Status based on SWAM Metadata

- SWAM information will be grouped to facilitate generating reports
- The SWAM baseline includes metadata (e.g., software type) that can be used to establish groups (i.e., finance) to report status including defects
- The CMaaS contractor is responsible for establishing and maintaining these groupings
- Group software by, for example:
 - Software role or device role
 - E.g., The status of database management system or database servers are reported to the IT department
 - Software Business Owner
 - E.g., The status of the financial management system is reported to the CFO
 - System supported
 - E.g., The status of web servers are reported to the web administrators
 - Device/CPE/Software
 - E.g., The status of the software executables for Adobe Acrobat on each device is reported to the systems administrator

Each D/A needs to work with their CMaaS contractor to determine the optimal set of SWAM metadata to collect and manage.



Topic: Summary



**Homeland
Security**

Summary of Topics

1. Software Asset Management (SWAM) Definition
 2. Making the Paradigm Shift
 3. Developing a Desired State Specification
 4. Actual State
 5. Discussion of D/A Specifics
 6. Summary
- Question & Answer Session

