TagVault.org

Certified SWID Tag Integration with Common Platform Enumeration names

April 9, 2012

Version 2.0

Steve Klos

stevek@tagvault.org

Contents

1 Executive Summary

This paper describes how software identification (SWID) tags for identifying software installed on computing assets can integrate with and potentially automate the creation of Common Platform Enumeration (CPE) names, which provide hardware and software information about computing assets.

The CPE name is designed to provide the following (from the CPE 2.3 Naming Specification Standard):

> *Common Platform Enumeration (CPE) is a standardized method of describing and identifying classes of applications, operating systems, and hardware devices present among an enterprise's computing assets. CPE can be used as a source of information for enforcing and verifying IT management policies relating to these assets, such as vulnerability, configuration, and remediation policies. IT management tools can collect information about installed products, identify products using their CPE names, and use this standardized information to help make fully or partially automated decisions regarding the assets.*[1]

The CPE name provides a key link among many of the Federal Government Security Content Automation Protocol (SCAP) databases.[2] When originally conceived, the CPE naming process assumed that the software publishers would (for the most part) provide the appropriate CPE name (or names) for inclusion in the Official CPE Dictionary. To date, adoption of CPE naming processes by the software publishing community has been limited, and so the National Institute of Standards and Technology (NIST) and MITRE Corporation have developed a process whereby CPE names are manually created and added to the CPE dictionary.

During the same period that the Security Content Automation Protocol (SCAP) infrastructure was being developed and implemented, software publishers were becoming increasingly motivated to help their customers manage the logistics, compliance and security of the various software products they license and use. This led to the development of an ISO standard, ISO/IEC 19770-2:20093, and the establishment of a registration and certification organization, TagVault.org, to provide the rules, processes and tools required to ensure that end-user organizations can lower the overall cost of managing the logistics, compliance and security of their software products.

TagVault.org provides tools that are used internally within a software publisher's environment to validate normalized and registered names, to validate that a minimum amount of data is included in the software identification (SWID) tag based on certification level, and to digitally sign the certified tag. This

[1] See: http://csrc.nist.gov/publications/nistir/ir7695/NISTIR-7695-CPE-Naming.pdf.

[2] See: http://scap.nist.gov.

[3] See: http://www.iso.org/iso/catalogue_detail.htm?csnumber=53670

ensures that the data is authoritative and that anyone looking at the data can validate the provenance of the data.

With the changes included in the CPE 2.3 specification, the certification processes defined by TagVault.org and the higher level of interest software publishers have in helping customers be more productive in managing the logistics, compliance and security of their software products, it is an ideal time to work towards a publisher-owned CPE naming process. This process would not only automate the creation of CPE names, but would provide authoritative data through the use of digital signatures and the addition of a significant level of additional meta-data to the SCAP infrastructure.

As part of this process, TagVault.org proposes that CPE names be automatically created based on a set of attributes the publisher must provide and a set of rules used to ensure the data conforms to the CPE 2.3 naming standard. These automatically generated names will vary slightly from the naming conventions currently used in the CPE dictionary, however, these differences will be far outweighed by the numerous benefits that come from this approach including:

- **Defined and validated checks** – Ensures that only registered and normalized values are used without putting a burden on the software publisher.

- **Complete CPE dictionary** – All software releases will use the same approach and the CPE name is generated at build time through a consistent rule set. Even if a publisher is not providing CPE names directly to NIST for inclusion in the CPE dictionary, when a title is discovered and/or a vulnerability for a title is discovered, the CPE name has already been created using the rules and can be added to the dictionary.

- **Consistency in identification across IT disciplines** – Security, logistics and compliance processes within an IT operation all need to work effectively together. By using the same reference data elements for all processes, significant automation can occur while lowering costs and increasing reliability.

- **Direct linkage between discovered titles and CPE names** – Having the ability to automatically associate discovered data with CPE names provides a significantly more automated solution to validating vulnerability and checklist operations. Even if software is provided without a SWID tag, a 3rd party SWID tag can be created and applied to the system (either through an internal process, or using 3rd party tools) in order to provide this link.

- **Integration of patch data sets with the CPE dictionary** – Currently, CPE names only apply to primary software releases since the manual application of CPE names to any larger data set would be impossible. However with SWID tags, every release, regardless of whether it's a primary release or a patch release, can be distributed with a SWID tag (and potentially a CPE name) from the publisher, improving the automation and patch applicability capabilities.

- **Additional metadata availability** – The SWID tag provides significantly more data than just a CPE name—data that can be used for vulnerability assessments, compliance and logistics efforts.

For example, the SWID tag can document an application's structure allowing organizations to know exactly what a suite consists of and which patches apply at which levels of a suite.

- **Authoritative data with clear provenance defined** – Every certified SWID tag must be digitally signed by the publisher and must be signed with a digital signature that has been countersigned by a certificate authority. Using this approach, along with the use of a trusted timestamp server in the certification process, ensures that the data can be reliably used regardless of the delivery route. Also note that as digital signatures increase in strength (e.g., moving from 1024-bit signatures to 2048-bit signatures), the certification process can be augmented to support these more stringent requirements.

- **Market level support for legacy applications** – There are already organizations in the market that are working to provide third party support for legacy SWID tags. Companies such as AssetLabs create software identification tags for software discovered using their Application Recognition Library.

The proposal for automated creation of the CPE name based on tag data could result in the following CPE names[4] being generated as part of the SWID tag certification process:

```
cpe:2.3:a:tagvault.org:Tag_Creation_and_Signing_Utility:1.0.0.0:-:-:-:-:-:-:certified_tag

cpe:2.3:a:symantec.com: Enterprise_Vault:10.0.1.0:-:-:-:-:-:-:certified_tag

cpe:2.3:a:microsoft.com: Office _2007:12.0.6607.1000:service_pack_3:-:-: Professional:-:-:certified_tag
```

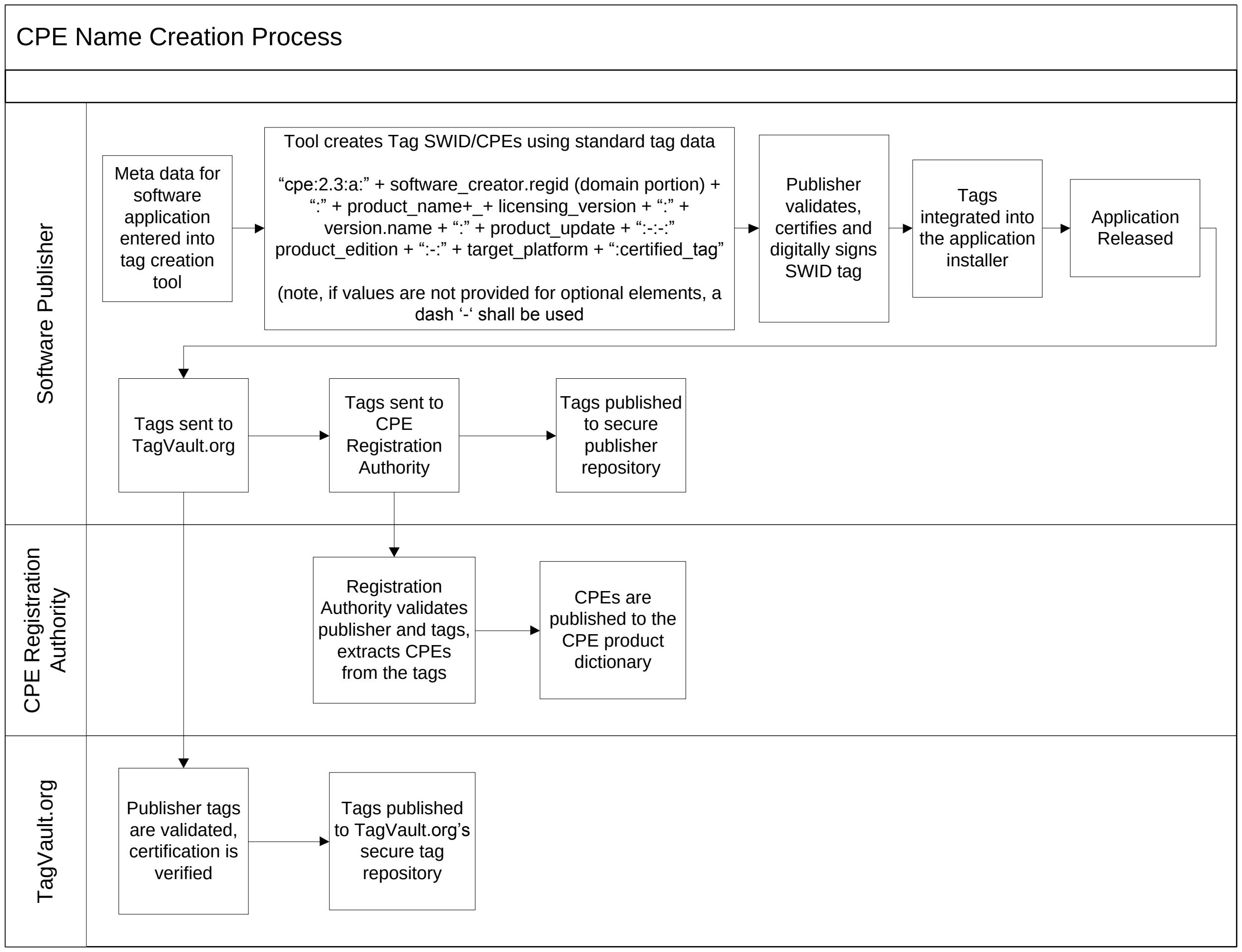
In the above examples, the "other" attribute available in the CPE name is used to indicate that this CPE name comes from a certified tag, indicating that additional metadata is available.

This approach naturally complements the existing process of manually created CPE names, but enables an automated approach to having the industry support CPE names in a manner that will provide authoritative, consistent and timely reference data for the CPE dictionary.

The overall flow for CPE name creation within the SWID tag creation process is illustrated below:

[4] NB: CPE names created by this process will utilize the formatted string binding defined in the CPE 2.3 Naming specification. As described in that specification, formatted string bindings are mechanically translatable to URI bindings that are backward-compatible with CPE 2.2-conformant tools.

Certified SWID integration with CPE


CPE Name Creation Process
Software Publisher
CPE Registration Authority
TagVault.org
Meta data for software application entered into tag creation tool
Tool creates Tag SWID/CPEs using standard tag data
“cpe:2.3:a:” + software_creator.regid (domain portion) + “:” + product_name+_+ licensing_version + “:” + version.name + “:” + product_update + “:-:-:” product_edition + “:-:” + target_platform + “:certified_tag”
(note, if values are not provided for optional elements, a dash ‘-‘ shall be used
Publisher validates, certifies and digitally signs SWID tag
Tags integrated into the application installer
Application Released
Tags sent to TagVault.org
Tags sent to CPE Registration Authority
Tags published to secure publisher repository
Registration Authority validates publisher and tags, extracts CPEs from the tags
CPEs are published to the CPE product dictionary
Publisher tags are validated, certification is verified
Tags published to TagVault.org’s secure tag repository

2 Technical Summary

CPE names are needed to enable automation of software policy and security compliance checking. Despite this clear need, software publishers have not embraced the process of creating CPE names as part of their product release cycles. To compensate, NIST and MITRE have created a manual CPE name assignment process to manage the integration of a variety of SCAP information repositories. This manual process is not scalable, but in the absence of support from the publishing community, it is the only effective method to add CPE names to the dictionary.

Ideally, CPE name assignments should be performed by the publisher as part of the product release process. This allows the data to be validated and the assignment process to become more automated and scalable. The data set should be integrated with other industry efforts to create a standardized software identification scheme that can be used by software discovery products as well as within the software supply chain and for security automation.

TagVault.org is a registration and certification authority for 19770-2 SWID tags. If the verification and certification processes are executed properly, certified and digitally signed SWID tags offer authoritative publisher-assigned CPE names which meet or exceed the requirements of CPE names. Additionally, this proposal will not significantly change CPE definitions as they exist today, that is, CPE names can continue to be defined and used in exactly the same way as they are today for any applications that do not include SWID tags. Finally, because software that includes SWID tags will install the SWID tags on the computing device as part of the standard installation process, the SWID tags will provide an explicit ability to link discovered software with CPE names for these software titles.

The overall goal of this proposal is to enable software publishers to automatically create a CPE name for each released product as part of the process of creating a SWID tag, and to do so without requiring additional resources. TagVault.org proposes to add extended elements to the current version of the ISO/IEC 19770-2:2009 standard, and to define a SWID tag creation process that would be used in order to provide authoritative CPE names that come directly from the software publisher. The expectation is that any new elements added to the TagVault.org certification process will also be incorporated back into the ISO/IEC 19770-2:2009 standard through standard ISO procedures.

2.1 Proposed Integration Between Certified SWID Tags and CPE Names

Certified SWID tags and CPE names each serve a specific purpose. The intent and scope of certified SWID tags was initially a structure to ensure authoritative data that can be used to uniquely identify software titles installed on a computing device. This data may be used for logistics, compliance and security processes. The data is provided as an XML file and resides along with the software on the device.

CPE names, at least in as far as they relate to applications and operating systems, provide a unique identifier for a software item that can be used as a common reference for that item when security-based applicability statements are defined. CPE names also allow for logical groupings of software into sets of related items, where the groupings may be organized by vendor, product, edition, update, etc.

The goal of automatically creating CPE names from SWID tags is to ensure a clean and consistent data set for the overall SCAP as well as to ensure that a single software reference can be used throughout the lifecycle of a software title from installation media, to installation and use, to compliance efforts and associated security procedures. Having a defined set of software identification data that is authoritatively created by the publisher and is consistent across all of these disciplines ensures a much more effective and efficient set of IT processes.

2.2 Initial Integration Points for SWID Tags and CPE Names

Existing procedures for TagVault.org certified tags are formed from a base of seven (7) mandatory fields that the ISO/IEC 19770-2 standard requires for conforming SWID tags. TagVault.org also specifies registration and normalization procedures that must be adhered to in order for the SWID tag to be considered certified at any level. The data provided by even the most basic level of TagVault.org SWID tag certification creates a data set that generally meets the minimum requirements of the SCAP CPE 2.3 standard as well as ensuring that many of the data elements are normalized and validated.

Although CPE 2.3 (or prior) defines no minimum level of data required to create a conforming CPE name, conventional practice defines a “proper” CPE name as having at least four unique attributes: **part**, **vendor**, **product** and **version**. For software items, three of these elements – vendor, product, and version – are provided in the base level of SWID tag certification. For software products, the part attribute will generally be ‘a’ (application) and in special cases, may be ‘o’ (operating system).

There are four other attributes in the CPE 2.3 name specification which will be considered in some depth in this document (note that this document does not include details for target_sw; this attribute requires future review). These four attributes with high level details for each attribute (from the CPE Naming specification) are:

- **Language** – Values for this attribute SHALL be valid language tags as defined by [RFC5646], and SHOULD be used to define the language supported in the user interface of the product being described.

- **SW_Edition** – Values for this attribute SHOULD characterize how the product is tailored to a particular market or class of end users.

- **Target_HW** – Values for this attribute SHOULD characterize the instruction set architecture (e.g., x86) on which the product being described or identified by the WFN operates. Bytecode-intermediate languages, such as Java bytecode for the Java Virtual Machine or Microsoft Common Intermediate Language for the Common Language Runtime virtual machine, SHALL be considered instruction set architectures.

- **Update** – Values for this attribute SHOULD be vendor-specific alphanumeric strings characterizing the particular update, service pack, or point release of the product.

These additional CPE attributes are often used to create applicability statements which identify where software vulnerabilities may be found. For example, an applicability statement might look as follows (this is a portion of the applicability statement which comes from CVE -2011-1989[5]):

2.2.1.1 Vulnerable Software and Versions

Configuration 1

OR

- * cpe:/a:microsoft:excel:2003:sp3
- * cpe:/a:microsoft:excel:2007:sp2
- * cpe:/a:microsoft:office:2007:sp2
- * cpe:/a:microsoft:excel:2010::x32
- * cpe:/a:microsoft:excel:2010:sp1:x32
- * cpe:/a:microsoft:office:2010:sp1:x32
- * cpe:/a:microsoft:office:2010::x32
- * cpe:/a:microsoft:office:2010::x64
- * cpe:/a:microsoft:office:2010:sp1:x64
- * cpe:/a:microsoft:excel:2010::x64
- * cpe:/a:microsoft:excel:2010:sp1:x64
- * cpe:/a:microsoft:office:2004::mac
- * cpe:/a:microsoft:office:2008::mac
- * cpe:/a:microsoft:office:2011::mac

The goal of this proposal is to create a process that can be accepted and used throughout the software ecosystem allowing software publishers to automatically create CPE names that can be used for logistics, compliance and security operations. The next section details how this can be automated with the data sets provided directly from the software publisher.

2.3 Additional Data Elements Required in Certified SWID Tags

To facilitate integration of CPE names with the existing 19770-2 standard and TagVault.org certified tag requirements, TagVault.org proposes to add the following elements (shown associated with their corresponding CPE attributes) to SWID tags:

CPE Attribute	SWID Element
Language	Additional review and community discussion required (many products support multiple languages with a single release).
Other	For CPE names created through a SWID tag validation and certification process, this will include the string “certified_tag”.
Part	None – incorporated as option in the SWID tag verification and registration utility.

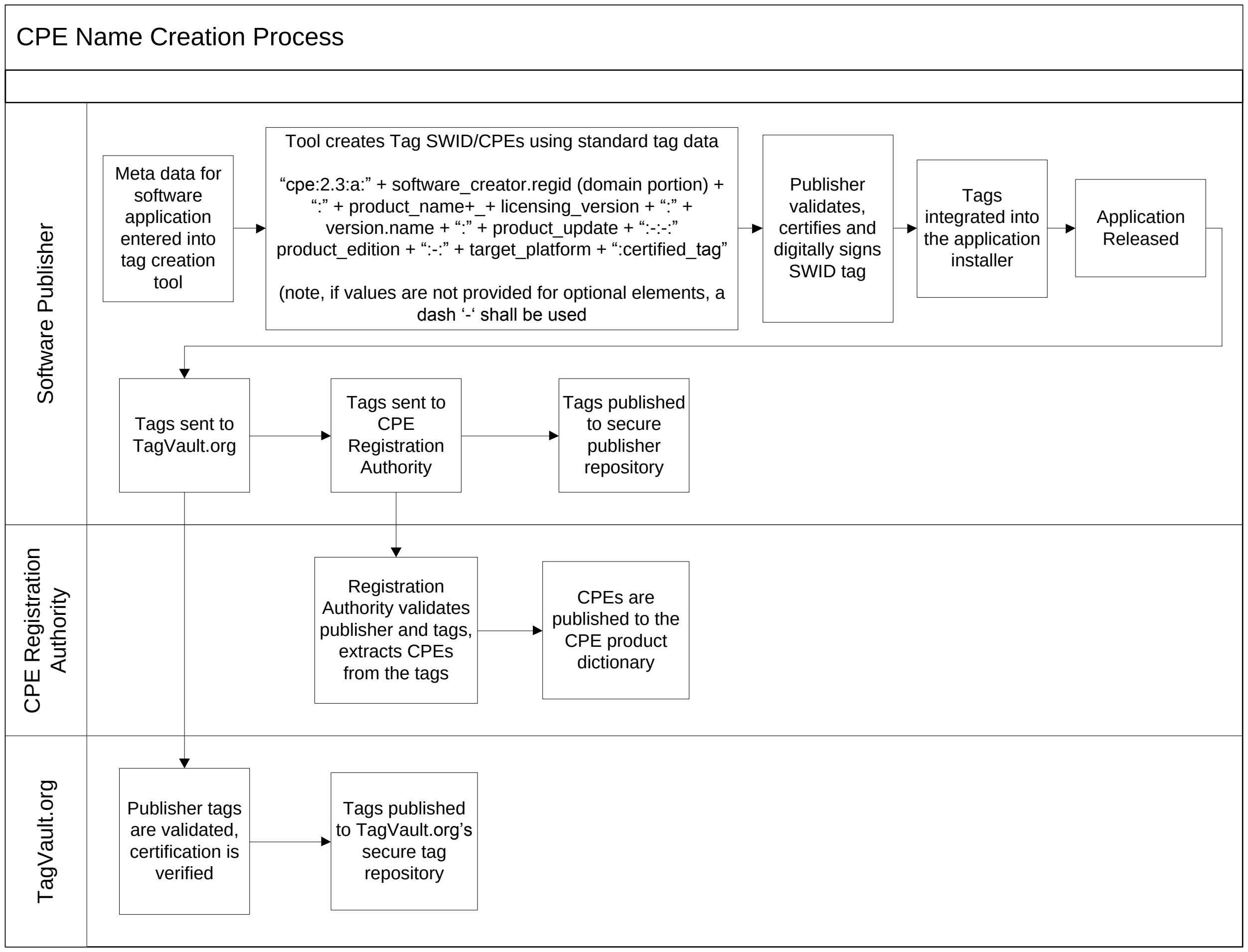
[5] See: http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-1989

Product	product_name (new) + licensing_version (new)
SW_Edition	product_edition (new)
Target_HW	target_platform (new)
Update	product_update (new)
Vendor	Software_creator.regid (existing)
Version	Version.numeric values – Major.minor.build.review (existing)

2.4 Example CPE Names Based on Recommendations

TagVault.org recommends that the cpe_id of a certified SWID tag be created to meet the requirements of a CPE formatted string binding, and be included as part of the certified SWID tag that is installed with a software product. The creation of the cpe_id will follow specified rules as defined in section 3 of this document, such as changing spaces " " in a string to underscores "_".

SWID tags and their corresponding CPE names will be created following a process shown below:


CPE Name Creation Process
Software Publisher
Meta data for software application entered into tag creation tool
Tool creates Tag SWID/CPEs using standard tag data
"cpe:2.3:a:" + software_creator.regid (domain portion) + ":" + product_name+_+ licensing_version + ":" + version.name + ":" + product_update + ":-:-:" product_edition + ":-:" + target_platform + ":certified_tag"
(note, if values are not provided for optional elements, a dash '-' shall be used
Publisher validates, certifies and digitally signs SWID tag
Tags integrated into the application installer
Application Released
Tags sent to TagVault.org
Tags sent to CPE Registration Authority
Tags published to secure publisher repository
CPE Registration Authority
Registration Authority validates publisher and tags, extracts CPEs from the tags
CPEs are published to the CPE product dictionary
TagVault.org
Publisher tags are validated, certification is verified
Tags published to TagVault.org's secure tag repository

The above process would result in the following sample CPE names:

```
cpe:2.3:a:tagvault.org:Tag_Creation_and_Signing_Utility:1.0.0.0:-:-:-:-:-:-:certified_tag

cpe:2.3:a:symantec.com: Enterprise_Vault:10.0.1.0:-:-:-:-:-:-:certified_tag

cpe:2.3:a:microsoft.com: Office _2007:12.0.6607.1000:service_pack_3:-:-: Professional:-:-:certified_tag
```

For CPE name assignment to occur as part of the SWID tag creation process, software publishers must have the ability to "own" the naming process for their own products. Although guidance can (and should) be provided to publishers regarding the preferred structure and conventions of names that will be most useful to security management, the goal should be to minimize requirements for CPE names to be manually entered and/or modified.

3 TagVault.org Process to Automatically Generate CPE Names

The following sections provide specifics on how SWID tags can be used to automatically create the nine (9) key attributes defined in the CPE 2.3 naming structure (with the assumption that the first two attributes, “cpe” and the cpe version number are essentially constants for any particular release). Many of these procedures rely on additional elements that are not yet defined in the ISO/IEC 19770-2:2009 standard. The 19770-2 standard allows for extended data, meaning that SWID tags that include additional data elements are still conformant with the standard. TagVault.org is defining these extensions as well as specifying the rules on how data for the extended elements need to be registered and/or formatted. TagVault.org has already committed to ISO Working Group 21 (WG21) that a revision of the standard will be initiated in 2012 to incorporate these and other changes into the published version of the ISO/IEC 19770-2 standard.

Since the integration with the SCAP infrastructure through a CPE name is an additional use case for the SWID tag, it is proposed that a new extended element be defined for the SWID tag: cpe_id. This ID will be a string and will follow the structure for a CPE 2.3 formatted string binding. The expectation is that a vendor creating a certified SWID tag for a software product will also automatically generate a corresponding CPE with this CPE name included in the SWID tag as the cpe_id. Following this process ensures that data values utilized by the vendor are consistent with the registered and normalized values defined by the certification authority, and also ensures that the CPE names provide a reliable link between installed SWID tags and SCAP data structures. The benefits of this approach cannot be overstated because the CPE name and the SWID tag are managed through all aspects of the software products lifecycle and will enable discovery, compliance, security, patching and many other logistical processes to use exactly the same reference details for a software product.

In addition to the verification steps, the creation of the CPE name (which is used as the certified SWID tag’s unique_id) must follow additional rules to ensure all string values adhere to the requirements of the CPE naming specification. Namely, all strings are filtered using the following rules:

1. Uppercase letter, lowercase letters, digit characters or a period “.”may be used (ASCII x41-x5a, x30-x39, x61-x7a, x2e).

2. The *underscore* (x5f) may be used, and it shall be used in place of whitespace characters (which shall not be used).

3. The backslash (x5c) is designated the *escape character and* shall only be used in that context. It shall be used in a value string when required to modify the interpretation of the character that immediately follows (see below). In these circumstances, the character following the backslash is said to be *quoted.*

4. The hyphen (x2d) may be used as part of a string value, but if used as the only value of an attribute, it represents the logical value Not Applicable (NA). For any value not provided for a CPE name, the unquoted hyphen shall be used.

5. The *asterisk* (x2a) and the *question mark* (x3f) may be used, and are designated *special characters.* However, if these characters are used within the strings specified in other SWID mandatory elements, they must be quoted. A special character may be included at the beginning of an otherwise non-empty value string, and/or at the end of the string. A special character must not be embedded within a value string. A single asterisk must not be used by itself as an attribute value. The asterisk must not be used more than once in sequence. The question mark may be used more than once in sequence.

6. All other *printable non-alphanumeric characters* (i.e., all punctuation marks, brackets, delimiters and other special purpose symbols, except for the special characters defined above) shall be quoted when embedded in value strings.

Each primary CPE attribute used for software naming is detailed in the sections below (note that attributes are in alphabetical and not in the CPE defined attribute order).

3.1 Language

Proper usage of this CPE attribute requires more review with independent software vendors and the security community.

Since SWID tags support a LANG attribute for string values, and CPE references provide only for the product name to be localized, the most obvious approach to Language is to specify that the XML LANG attribute be utilized for the cpe_id SWID tag element and that every localized product name include a matching localized cpe_id value. This is very easy to manage from the SWID tagging perspective, but it is unclear what the use cases for Language are.

Until this attribute has further review from the software community, it will not be directly addressed as part of this proposal.

3.2 Part

Part is an enumerated value defined as:

a = application
o = operating system
h = hardware

For purposes of software definitions, entries from most vendors creating certified SWID tags will use ‘a’ for application, and a few vendors will use ‘o’ for operating system. It is not expected that hardware vendors will be creating certified SWID tags.

For a system that automatically creates CPE entries as part of the validation process, it is recommended that the utility specify ‘a’ for part by default. This should be defined as an option within the tool, so that a vendor which develops and publishes operating systems can override the default value and specify that the part attribute value should be ‘o’.

3.2.1 Part – SWID Equivalent

No SWID element required.

CPE Attribute	SWID Element
Part	None – incorporated as option in the SWID tag verification and registration utility.
	Default option is to specify ‘a’ for application with options to change to ‘o’ for operating system when required.

3.3 Product

The 19770-2 standard only has one data value for product title and this is expected to be populated with the full title as one might find on a windows platform in the add/remove programs utility. However, the information from this product_title does not allow for a breakout of the discreet data values required for security, logistics and compliance operations.

To provide a more useful breakout of discreet data values, Technical Group (TG) 21, the organization is a mirror of Working Group (WG) 21 which is the ISO working group responsible for SAM standards development. TG 21 is part of the US standards review and creation organization and they had a discussion about this issue at length during the March 2012 TAG meeting. During the meeting, it became clear that there would be a relatively wide industry support to add two additional elements to the 19770-2 standard. Specifically, two new attributes were defined:

- **product_name** – which will only contain the name of a product, without the manufacturer or edition included. This element would be a **mandatory** element. Examples would include, “Office”, “Creative Suites”, “Backup Exec”, etc.
- **licensing_version** – this is the version number that is often associated with license entitlements as well as being a common industry reference for a particular major release of an application. This element is proposed to be an **optional** element primarily because not every company uses a licensing_version. Examples include, “2007”, “2010”, “5.5”, etc.

Of particular note, a CPE reference should contain both the licensing_version as well as the product_version. This is due to the fact that product_version allows for much more granular control of applicability statements as well as the ability to identify when a major code base change has occurred.

The proposal for the automated generation of CPE names within SWID tags is to associate the CPE ‘product’ attribute with the product_name plus an underscore ‘_’ followed by the licensing_version as the publisher defines it. Thus, Microsoft Office Professional 2010 would have the Product attribute in the CPE name be “Office _2010”. Including the licensing_version in the CPE product attribute allows us to provide a simple method to specify a wide ranging applicability statement that can easily encompass all licensing versions of a product - i.e. “Office*”, or could encompass a specific major release of a product – i.e. “Office_20010”. As will be seen in the version attribute description, an applicability statement can apply to a very specific point release of a product as well.

The benefit of this approach is that the software publisher owns the product name, and this name is consistently used throughout the purchasing, inventory and discovery, and security processes. This enables much more automation in the process of identifying whether a software product is actually installed or may require a patch or configuration change to ensure a secure system.

Utilizing a single, consistent product attribute enables additional cross-functional automation. For example, if a specific software product is purchased, the name will be used consistently everywhere, permitting configuration, vulnerability and checklist details to be automatically picked up. Also, if a discovery tool or help desk incident report happens to reference a product that is being researched for security issues, the reference information can be associated in an automated fashion.

3.3.1 Product – SWID Equivalent

New SWID element used.

CPE Attribute	SWID Element
Product	**product_name**(new)_**licensing_version**(new)
	Product_name: New SWID tag element defined as an extended element in the TagVault.org certified SWID tag document product_name value will be modified if necessary to meet the requirements of a CPE attribute-value string.
	Product_name will be registered with all registered values being linked to the regid for the owner. This ensures that the validation steps used for a specified vendor are only for that vendor's products.
	licensing_version: New SWID tag element defined as an extended element in the TagVault.org certified SWID tag document licensing_version value will be modified if necessary to meet the requirements of a CPE attribute-value string.
	Licensing_version will be registered with all registered values being linked to the regid for the owner. This ensures that the validation steps used for a specified vendor are only for that vendor's products.
Example: **SWID Tag: product_title = Microsoft Office Professional 2010** **Product_name = Office** **Licensing_version = 2010** **CPE: product = Office_2010**	

3.4 SW_Edition

SW_Edition is a newly defined field in version 2.3 of the CPE specification. It was created to enable edition-specific information to be broken out separately from other information which, over time, also came to be treated as "edition" information. The edition is almost always included in the product_title as well since most software vendors consider different editions to be completely different products from a sales, licensing and support perspective.

Neither the 19770-2 standard nor the TagVault.org extended elements currently define an edition element, however, there is support for adding this element as an extended element in the TagVault.org SWID tag certification document. Our proposal is that a product_edition element be created.

It would not be possible to automate the validation step to know if a product_edition data value is required or not, but if a product_edition is provided, it is a very easy process to validate that the edition is chosen from an enumerated set of values.

This data value should be a registered value, and the registered values should be uniquely defined per vendor. The uniqueness is required since vendors tend to use the same edition naming structures and will generally not want an engineering team to select an edition name that is not used by their organization.

For example, Adobe provides a version of Photoshop called "Photoshop" with no qualifier and a different version called "Photoshop Extended" (where "Extended" is considered the edition). Microsoft, on the other hand, does not presently use the term "Extended" in any of its product editions.

Vendors can register any edition names they need without restriction. However, when the SWID tag is validated against the registered values, and when a product_edition is included in the SWID tag, that product_edition must be among the registered value for that vendor.

3.4.1 SW_Edition – SWID Equivalent

New SWID element required.

CPE Attribute	SWID Element
SW_Edition	product_edition (new)
	New SWID tag element defined as an extended element in the TagVault.org certified SWID tag document.
	Product_edition will be registered with all registered values being linked to the regid for the owner. This ensures that the validation steps used for a specified vendor.
Example: **SWID Tag: product_edition = Professional** **CPE: SW_Edition = Professional**	

3.5 Target_HW

Target hardware defines the type of operating platform the software was compiled for. In the case of a Windows application, this would likely be x32 or x64. However, in the case of applications created for other platforms, this could be x86, it64, SPARC, RISC, etc.

Target_HW must be consistently used across multiple different publishers to ensure that the same values mean the same things to different entities. This means that Target_HW must be a commonly managed value and must go through at least some level of community based normalization. TagVault.org has a process for registering new names that are commonly used across multiple vendors and this process includes an escalation process up to the board of directors.

Neither the 19770-2 standard, nor the TagVault.org extended elements currently define a Target_HW element, however, there is support for adding this element as an extended element in the TagVault.org SWID Tag Certification document. The proposal is that a new element - target_platform be created in the TagVault.org extended elements.

The values for target_platform will be registered and normalized to ensure consistency across multiple different publishers and tool providers.

3.5.1 Target_HW – SWID Equivalent

New SWID element required.

CPE Attribute	SWID Element
Target_HW	target_platform (new)
	New SWID tag element defined as an extended element in the TagVault.org certified SWID tag document.
	target_platform will be registered as a normalized value list. This normalized list will be consistently verified for every certified SWID tags regardless of publisher.
Example: **SWID Tag: target_platform = x64** **CPE: target_HW = x64**	

3.6 Update

This attribute is used to specify major point releases for a product. An example of this is a Microsoft Service Pack which is a collection of updates, fixes or enhancements for a specific product, such as "Windows SQL Server 2008 Service Pack 3".

Service packs can be identified by the version number of the underlying component, but a service pack (or other update) is often used as minimum criteria for installation dependencies and is something that IT professionals are used to working with.

Neither the 19770-2 standard, nor the TagVault.org extended elements currently defines an element that is related to the update attribute, however, an extended element could be incorporated into the TagVault.org SWID tag certification document. The proposal is that product_update be created.

This data value will be a registered value, and the registered values will be uniquely defined per vendor. The uniqueness is required since vendors tend to use the same update naming structures and will generally not want an engineering team to select an update that is not used by their organization.

Vendors can register any update names they need without restriction. However, when the SWID tag is validated against the registered values, and when a product_update is included in the SWID tag, that product_update must be among the registered values for that vendor.

3.6.1 Update – SWID Equivalent

New SWID element required.

CPE Attribute	SWID Element
Update	product_update (new)
	New SWID tag element defined as an extended element in the TagVault.org certified SWID tag document.
	Product_update will be registered with all registered values being linked to the regid for the owner. This ensures that the validation steps used for a specified vendor.
Example: **SWID Tag: product_update = service pack 2** **CPE: Update = service_pack_2**	

3.7 Vendor

Version 2.3 of the CPE naming specification does not include prescriptive guidance on requirements for the vendor name value. Version 2.2 of the CPE naming specification, however, indicates that the vendor name should be related to the primary DNS name used for that vendor without the top level domain qualification (e.g., "hp" instead of "hp.com"). Due to the fact that the SWID tag includes a regid which is based on the domain name and the reasonable potential that two different publishers may share the same organization-specific label but have a different top level domain, this proposal recommends that the fully qualified domain name be used for the vendor value.

Further, the SWID tag has potentially three different regid values – one for the tag creator, one for the software creator and one for the software licensor. It is recommended that the domain specified in the regid of the software creator be utilized for the value of the vendor attribute.

A regid for a SWID tag follows a rigid structure as shown:

- The string "regid"
- A dot "."
- Date code for domain ownership YYYY-MM format. MM is the first month the owner owned the domain on the 1st day of the month at 00:01 GMT.
- A dot "."
- The reverse notation domain (i.e. tagvault.org would be represented as org.tagvault)
- A comma ","
- An optional suffix

Thus, a regid for a company that uses the domain sampleco.com is:

regid.1992-11.com.sampleco

In order to minimize issues with backwards compatibility, it is recommended that the full domain name (with the top level domain) be used in the normal reference direction (that is with the top level domain portion at the end).

Using the above example, the vendor name based on the certified SWID tag is:

sampleco.com

3.7.1 Vendor – SWID Equivalent

Existing SWID element used.

CPE Attribute	SWID Element
Vendor	Software_creator.regid (fully qualified domain name in proper order – with the top level domain on the right hand side).
Example: **SWID Tag: software_creator.regid = regid.1992-12.com.symantec** **CPE: Vendor = symantec.com**	

3.8 Version

The Version used by most software products typically follows a four-part decimal notation. There are exceptions to this rule and there are products that display one version to their users with limited detail, while providing more specific version details in underlying software definitions. Finally, there is at least one vendor (Cisco) that has an eight-part version string, but in this case, Cisco has a process to specify a four-part version equivalent.

Certified SWID tags allow for both a string notation of version (as is typically seen by the end-user) as well as a four-part decimal notation of the version. By requiring a four-part decimal version identifier, the versions can be unequivocally sorted in a proper rank order allowing end-users and tool providers to have a consistent view of version ordering for any product from any publisher.

The recommendation is that the version used in the CPE is based on the integer versions concatenated together with the period "." character.

3.8.1 Version – SWID Equivalent

Existing SWID element used.

CPE Attribute	SWID Element
Version	Version.numeric values – Major.minor.build.review
Example: **SWID Tag: Version.numeric** **Major = 5** **Minor = 6** **Build = 3** **Review = 0** **CPE: Version = 5.6.3.0**	

3.9 Other

The "other" attribute that is included in the CPE 2.3 name can be utilized to indicate that a CPE name is created from an authoritative publisher. By indicating that the CPE is associated with a certified SWID tag, automated systems can reference this information and identify that the software identified by the CPE name can be identified on computing systems through the SWID tag. Alternatively, if a SWID tag includes a cpe_id element, a discovery and/or software identification utility can recognize that additional information about threats and configuration issues related to this title can potentially be found in the SCAP data repositories.

This proposal recommends that when a SWID tag containing a cpe_id is created through a certification process, the cpe_id include the string "certified_tag" as the value of the "other" field in the CPE name.

4 Potential Implementation Issues

Depending on how existing tools are implemented, there are at least three existing potential issues and one longer term potential implementation issue that need to be understood regarding this change and its potential impact on tools that support and/or use the existing CPE names.

1. The product name will change from a shorthand form that is currently set and defined by NIST to the software publisher's product name plus the licensing version.

   Example – Instead of a product attribute of "office", the product attribute would be "Office_2007".

   Potential Impact – It isn't expected that this change will have much of an impact on tools due to the fact that most tools will be using the product to match the string values. The primary impact is likely to be seen in the applicability statements that are required to define a group of software titles that are related to each other. Since CPE 2.3 allows for wildcard text characters and has the Applicability Language Specification, it is expected that using the software publisher's product name plus the licensing version will have no long-term negative impacts.

2. The vendor, instead of being just the primary portion of the domain name will be the entire domain name. This gracefully deals with cases where two vendors have the same primary name, but their domains are in different top level domains (i.e. .com, .net, .org).

   Example – Instead of "Microsoft", the value "microsoft.com" would be used.

   Potential Impact – As domains are transitioned from being just the primary company name portion of the domain to the full domain including the top level domain, applicability statements may need to be adjusted to include a wild card character to ensure all references are identified.

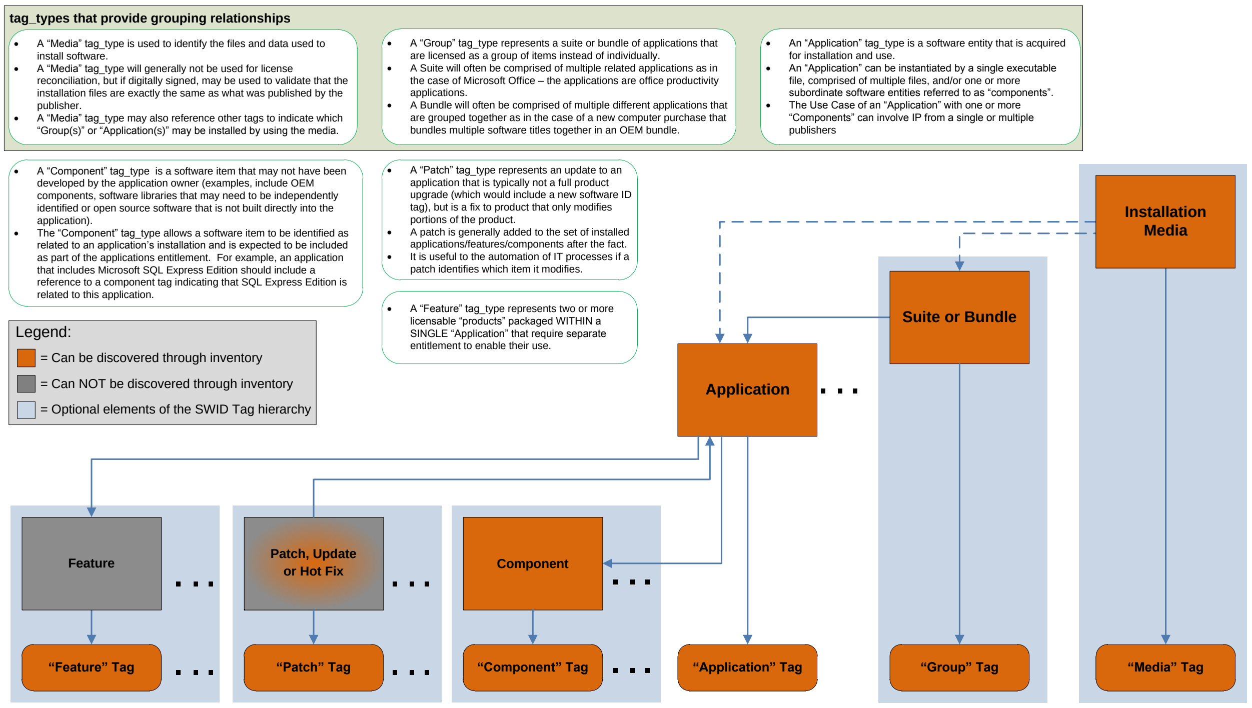
3. Version will be the real "version" of the application and not the marketing version. The marketing version of a product may sometimes be found in the product name attribute.

   Example – If the product is Microsoft_Office_2010_Professional, there will be multiple versions of this product. One valid version for example is 14.0.6029.1000 – it is the version number that represents the code base used for that product, so the "real" development version number should be used.

   Potential Impact – In many cases, current applicability statements do not address a level of resolution below the marketing version of one of the office products. If a patch is required for a specific version of an Office product (e.g., version 14.0.6029.1000), it is currently necessary to specify the applicability statement as applying to every copy of Office 2010.

   There will be a need for a different manner of applicability statements to be utilized. If the marketing version (2010) is required, the product will contain this information and if a specific version (or versions) of a product or suite is required, the version (or a portion of the version with wild-cards included) should be used.

4. **Future Consideration** – SWID tags include structural information to identify how software packages are related to one another. This relationship includes both parent to child and child to parent relationships and it also includes an element that indicates tag_type. Multiple levels of relationships can be specified. The high level perspective of this structure is shown below:



tag_types that provide grouping relationships

- A “Media” tag_type is used to identify the files and data used to install software.
- A “Media” tag_type will generally not be used for license reconciliation, but if digitally signed, may be used to validate that the installation files are exactly the same as what was published by the publisher.
- A “Media” tag_type may also reference other tags to indicate which “Group(s)” or “Application(s)” may be installed by using the media.

- A “Group” tag_type represents a suite or bundle of applications that are licensed as a group of items instead of individually.
- A Suite will often be comprised of multiple related applications as in the case of Microsoft Office – the applications are office productivity applications.
- A Bundle will often be comprised of multiple different applications that are grouped together as in the case of a new computer purchase that bundles multiple software titles together in an OEM bundle.

- An “Application” tag_type is a software entity that is acquired for installation and use.
- An “Application” can be instantiated by a single executable file, comprised of multiple files, and/or one or more subordinate software entities referred to as “components”.
- The Use Case of an “Application” with one or more “Components” can involve IP from a single or multiple publishers

- A “Component” tag_type is a software item that may not have been developed by the application owner (examples, include OEM components, software libraries that may need to be independently identified or open source software that is not built directly into the application).
- The “Component” tag_type allows a software item to be identified as related to an application’s installation and is expected to be included as part of the applications entitlement. For example, an application that includes Microsoft SQL Express Edition should include a reference to a component tag indicating that SQL Express Edition is related to this application.

- A “Patch” tag_type represents an update to an application that is typically not a full product upgrade (which would include a new software ID tag), but is a fix to product that only modifies portions of the product.
- A patch is generally added to the set of installed applications/features/components after the fact.
- It is useful to the automation of IT processes if a patch identifies which item it modifies.

- A “Feature” tag_type represents two or more licensable “products” packaged WITHIN a SINGLE “Application” that require separate entitlement to enable their use.

Legend:

🟧 = Can be discovered through inventory

⬛ = Can NOT be discovered through inventory

⬜ = Optional elements of the SWID Tag hierarchy

Potential Future Impact – Using this structure, it’s possible to identify software at the level required for the specific task. In security content automation, this may entail finding a suite or a bundle and identifying all installed applications that belong to that bundle. It may also entail the identification of a patch (that identifies the application or component it applies to) and automatically receiving an exception when a patch becomes available and there are components, applications or suites that exist within an infrastructure, but have not yet been patched.

These capabilities are provided directly within a certified SWID tag, but they can also be integrated into the CPE and SCAP infrastructures relatively easily. If and when security automation is ready to implement further automation for software patching, these built-in capabilities will be available for use. Using the rules applied to SWID tags for applications and suites, it will be possible to generate CPE references for patches in an automated fashion as well.