Common Platform Enumeration (CPE) Technical Use Case Analysis

The MITRE Corporation
November, 2008

Executive Summary

A common theme taken from discussions at the Common Platform Enumeration (CPE) Developer Day (April 30th, 2008) was that differences in how and why CPE is being utilized across the community create conflicts among members of the CPE community, and thus create obstacles to CPE's success. In its non-competitive FFRDC role as moderator of CPE, MITRE has conducted a series of interviews with representatives of the CPE community (under a combination of implied and formal NDAs) and has performed an analysis of the different technical use cases for CPE. This white paper presents the results of that analysis.

Summarizing:

- *Four general use cases were identified including: Software Inventory, Network-Based Discovery, Forensic Analysis/System Architecture and IT Management.*
- *Universal agreement exists on the need to support the Software Inventory use case as a "must-have". No other use case was universally identified as a "must-have".*
- *Universal agreement exists that a) "concrete" names are needed to support the Software Inventory use case and b) these names can be created using current CPE 2.1 structures. But...*
- *Disagreement remains on how to best use CPE 2.1 to support the Software Inventory use case and the need for "abstract" names.*

Introduction

A common theme taken from discussions at the Common Platform Enumeration (CPE) Developer Days (April 30th, 2008) was that differences in how and why CPE is being utilized across the community create conflicts among members of the CPE community, and thus create obstacles to CPE's success. The CPE community includes vendors of information security products, operating system and application vendors, large enterprise users of IT and information security products, and government and commercial organizations seeking to support those users by promoting standardization, information sharing, and common practices.

Points of conflict, which are visible in the discussions on the publicly accessible CPE Discussion email list[1] and in the CPE Developer Days minutes[2], include, for example:

- How completely should a CPE Name be stated? The CPE 2.1 Specification defines seven components of a CPE Name, and allows empty components. In practice, which components are left empty, why, and what consequences does this have for possible uses of the CPE Name?

- How should a CPE Name be interpreted? Does it represent an actual piece of software or a collection of software? If a name represents a collection of software, does the use of the CPE Name imply a relationship to all software in that set or only to some?

- What is a "platform" – and what is not? The CPE 2.1 Specification states that information technology (IT) platforms are hardware, operating systems (OSs), and applications. However, many operating systems include software that could be viewed as distinct applications; libraries can be shared by applications and OSs. What distinction, if any, can the CPE community make between platforms, products, and systems? Based on this distinction, what may or may not be given a CPE Name?

These issues are stated in technical terms, but the differences in opinion arise from variations in intended uses.

Data Collection

To help better understand these differences, the CPE team at MITRE conducted a series of semi-structured interviews with representatives of various government and commercial CPE users who are familiar with CPE-related issues. These users included a) producers of operating systems and software applications, b) producers of software management and security products and c) users and integrators of software management and security products. The affiliations of the interviewees spanned a) US Government, b) US Government contractors, c) commercial software vendors and d) non-commercial/open-source groups. In terms of their adoption of CPE, the participants ranged from those that rely heavily on CPE within their implementation to those that are trying to determine if CPE is a worthwhile investment.

The interview questions were designed to gain insight into the specifics of how CPE was being implemented by the various participants. This required detailed discussions of sensitive, proprietary information including internal implementation techniques and strategies, product roadmaps and technical and business challenges. For this reason, the data was collected with the promise of non-disclosure and non-attribution. Some organizations have existing non-disclosure agreements with MITRE and agreed to participate in the interviews under those conditions. Some organizations noted that their willingness to discuss these issues at this level of detail was

[1] The email list can be joined from http://cpe.mitre.org/community/discussion_list.html

[2] Available at http://cpe.mitre.org/community/CPE_dev_days_minutes.pdf

directly tied to MITRE's non-commercial, non-compete charter, which is based in its status as a Federally Funded Research and Development Center (FFRDC).

This document illustrates the outcome of these interviews and the collective analysis of the CPE team as a result of data collected from interviewees.

Summary of Analysis

Based on these interviews, the email discussions, and the discussions at CPE Developer Days, MITRE has performed an analysis of the different technical uses of CPE. One use of CPE – to facilitate software inventory management (also referred to as credentialed endpoint management) – was viewed as a "must-have" across the CPE community, and is largely supported by CPE Version 2.1. There remains some amount of lingering disagreement on how to best to support this use case with CPE 2.1, but the over-riding consensus is that certain aspects of CPE 2.1 are well matched to this use case.

CPE 2.1 supports other uses incompletely at best. These include: network-based discovery, forensic analysis, and IT management. While those uses are important to the enterprise users of CPE-based tools and databases, vendor members of the CPE community frequently prefer to meet those needs in proprietary or enterprise-specific ways. The CPE community, which includes the Government organizations that fund the maintenance of the CPE Specification and the Official CPE Dictionary along with the commercial vendor organizations that participate in the CPE Working Group, must decide on two issues:

1. **Restrict Scope of CPE to Software Inventory Use Case? -** The community can choose to restrict CPE to the technical use case that is universally agreed to be a "must-have" (and possibly seek other solutions outside of CPE for the other technical use cases), or to expand CPE into these other areas (and risk diluting the effectiveness of CPE with its main audience).

2. **How to Best Use CPE for the Software Inventory Use Case? -** While there is broad agreement that CPE 2.1 is sufficient for producing the kinds of names needed to support the Software Inventory use case, there remains some disagreement on how the use case should be supported and if or how CPE should be modified.

This white paper is intended to inform these decisions.

Technical Use Cases

In the context of the Making Security Measurable effort, a Technical Use Case defines an intended "best-practice" usage of a standard.[3] A Technical Use Case, generally presented in one or a few paragraphs, identifies

[3] See The MITRE Adoption Programs and the NIST SCAP Validation Program, August 2008, available at http://cve.mitre.org/adoption/Adoption_and_Validation_August2008.pdf. Examples are taken from the OVAL Use Cases (http://oval.mitre.org/language/about/use_cases.html).

- the type of organization that uses the standard (e.g., organizations that develop vulnerability assessment tools, patch management vendors, configuration management tool vendors, enterprise users of such tools),

- the purposes for which it uses the standard, and

- expectations on those uses (in particular, expectations of product interoperability and/or of reporting).

A variety of possible uses for CPE have been identified, including software inventory management (also referred to as credentialed endpoint management), network-based discovery, forensic analysis, and IT management. The following sections summarize the results of our analysis of these possible use cases.

Supported Use Case

Based on these interviews, the email discussions, and the discussions at CPE Developer Days, MITRE has performed an analysis of the different technical uses of CPE. One use of CPE – to facilitate software inventory management (also referred to as credentialed endpoint management) – was viewed as a "must-have" across the CPE community, and is largely supported by CPE Version 2.1.

Software Inventory Management

Software inventory management products include configuration audit, endpoint management, and asset inventory tools. Consistent with the use cases for OVAL, such products have credentialed access to end systems.[4]

The Technical Use Case is as follows: A software inventory management product vendor uses CPE Names to tag data elements within their product's data model. These data elements may directly represent the individual software products that exist on an end system (e.g., a laptop, desktop, or server), in which case the CPE Name represents a standard identifier for instances of that record type. Or, the data elements may represent some other object (e.g., a configuration check, a vulnerability check, a patch check, a configuration control change, or a patch), in which case the CPE Name implies a relationship to a software product as identified by the CPE Name.

With this tagging, the product vendor can develop, or can enable their product to interoperate with, different tools that share information about the individual software products on the end systems. Whether those tools perform asset management, vulnerability management, configuration assessments, or tactical descriptions of a given network, they have a common need to share software inventory information. The tools are expected to use CPE Names for this purpose.

---

[4] Credential access to an end system can be implemented in one of two ways. A software agent that runs with full administrative privileges can be installed the end system. The agent then reports to and is controlled by a central management system. Or, the central management system can connect to the end system over the network using credentials that give it full administrative privileges.

Issues to Consider

Several issues are raised by this use case. **First**, the statement of the technical use case given above raises the question: "What constitutes a software product?" Information gathered during interviews revealed the following characteristics of a software product as distinctive:

- A user can download or buy it.
- There is a vendor/organization that produces it.
- An enterprise IT administrator can push it out over the enterprise network and install it into their environment.
- It is (or can be) recorded by an asset management tool.

Thus, for purposes of this use case, a software product – whether identified as an operating system or as an application in the CPE naming scheme – has those four characteristics. We emphasize that this definition is a functional definition relative to the meaning that must be made across the boundaries of different tools and organizations. It is **not** a technical definition of what constitutes an individual software product. We highlight the functional definition with an example. Consider a distribution of Linux. From a technical point of view, it can be argued that the version of the Apache web server that ships with that distribution should be considered as a component of the larger Linux operating system. However, those participants who were actively involved in directly supporting Software Inventory management were unanimous that their customers have an expectation to view Apache as a piece of managed software. For this functional reason, they insist that the distribution of Linux and Apache both be given (peer) CPE Names. Additionally, consider a different Linux distribution that ships with the "same" version of Apache. Despite the fact that these two versions of Apache might be considered to be the "same" from a technical point of view, the users considered these to be different in that they are distributed by different organizations. In this way, they are looking for the CPE Name to be similar to the construction of Vehicle Identification Numbers (VINs) for cars and International Standardized Book Numbers (ISBNs) for books in that the naming scheme is (partially) based on who manufacturers or publishes the product. The functional relationships such as who should be contacted for a patch are more important to the Software Inventory use case than technical facts about the code base.

Second, the software inventory management technical use case is based solely on credentialed access to the end system, as opposed to unauthenticated network scanning. Asset identification based on port scanning and finger printing often requires a very different form of product naming and categorization than that which is possible via credentialed access. The following section on unsupported technical use cases will discuss this in greater detail.

Third, the software inventory management technical use case is about machine-to-machine communication. In other words, it is focused around the needs of two different tools trying to share information. Interviews revealed no need to standardize communication with human users, as humans can make the comparison between non-standard terms. This is very different than with CVE, where it is common practice for a human user to use a CVE identifier in a query, report, or email discussion. But with CPE, a human user would never be expected to enter in a CPE name into a search box or to view asset reports displaying CPE Names. Rather a user

would interact with a set of drop down menus, or would enter in search terms related to their desired product. Even defined titles are unnecessary, since tools will often use their own title that better align with the message or format it is trying to deliver.

All interviewees agreed that CPE must support the software inventory management technical use case – which is to say, they all agreed that CPE must be used to create unique names for individual software products. They also agreed that the current Version 2.1 of CPE is largely sufficient to fill this need. However, the analysis revealed a significant disagreement on how CPE Names should be created and understood, even within the single use case of Software Inventory Management.

The Outstanding Question

There is universal agreement that CPE Names can and should be used to identify specific software products as defined above. Further, there is universal agreement that CPE 2.1 creates a namespace that is adequate to uniquely identify the software products to meet the functional communication and coordination needs of this use case. Putting it another way, if there is a specific, individual software product represented in a software inventory management tool, there is broad agreement that a proper and unique name can be constructed for that product based on CPE 2.1. However, there remains some disagreement on which CPE Names should be legitimate and are in fact necessary for the software inventory use case.

In fulfilling the software inventory technical use case, the specification for CPE 2.1 aspires to do more than to create unique names for software products defined at a single level. It also aspires to create names for "categories", "ranges" or "buckets" of software products. The creation of these names is (partially) defined by the syntax rules and examples within the specification. The relationship between the categories or buckets is made explicit through the matching algorithm and prefix structure of the names.

The distinction between names associated with specific software products and names associated with larger categories of software products was revealed in the how interviewees referred to different kinds of CPE Names. CPE Names associated with specific software products were sometimes called "real", "concrete" or "fully qualified". Names associated with categories of software products were sometime called "roll-up" or "higher-level" names.

How CPE handles this distinction and how vendors use and interact with these different names still needs to be answered. This outstanding issue is at the core of the differences found between members of the CPE community.

Unsupported Technical Use Cases

The interview process, together with discussions on the CPE email list and at CPE Developer Days, revealed a number of other technical uses cases for which CPE almost satisfies their needs. These are described below. One common thread is that for each technical use case below, the current version 2.1 of CPE does not exactly work and needs to be fundamentally altered to do so.

As will be discussed in the Conclusion, those alterations might be mutually inconsistent and could potentially break the Software Inventory Management technical use case.

Network-Based Discovery

Some enterprise users and tool developers are involved with network-based discovery of information that is performed without credentialed access to end systems. Their desire is to use CPE to tag the assets found and thus enable sharing of information with other information data sources. Unfortunately, unauthenticated network-based discovery often results in only partial information. Sometimes, full details cannot be determined in this way, but can only be obtained by a credentialed access to the end system. This results for the need for terms like "linux" or "printer" when the discovery algorithms can determine this level of information but nothing more. To support this, some tool developers have implemented a higher level roll-up capability as part of their user interface. That capability incorporates proprietary categorizations of network functionality and reflects the developer's perspective on discoverable assets. Our interviews did not reveal a consistent perspective, (which would have suggested the feasibility of achieving a consensus, in order to extend or modify the standard), and in fact indicated that attempts to define a common approach to rolling up partial information would meet with some resistance. Even if such agreement could be reached, this would produce a categorization scheme (or taxonomy) based on network functionality which would conflict with CPE's producer-based taxonomy.

Forensic Analysis/Software Architecture

In the forensic analysis technical use case, tools are looking to tag things that are of interest to the forensic analysis being done. For example, when dealing with vulnerabilities, one respondent suggested everything that gets a CVE should also get a CPE. This need is driven by the fact that information about a specific vulnerability needs to be associated with the "thing" that it applies to. Unfortunately, many of the "things" that have vulnerabilities are artifacts or components contained within software products and are not products in and of them selves. Examples include drivers and individual dll files. In addition, the current name format used by CPE probably would not make sense for naming these types of things. The current CPE name structure (like ISBNs and VINs) reflects who produced a given software product, an approach that is well suited for supporting the software inventory use case. In contrast, a naming scheme that would support the forensic analysis/software architecture use case would need to reflect universally accepted concepts of operating system and application components. In short, we can't currently talk about these things in CPE, as Version 2.1 of CPE only can name software products. Thus, we anticipate that support of this use case would be in conflict with the software inventory use case.

IT Management

In the IT management view, platforms play functional roles (e.g., server). Some IT managers would like to have lower level CPE names roll up to these functional roles. This is currently outside the scope of CPE. Unfortunately, the naming conventions for functional roles do not align with the current CPE convention for naming software products, which is based on the "who produced it?" perspective, not the "what is it used for?" perspective. In addition, it is unclear whether a common set of functional roles for platforms could be defined in an enterprise-

independent way, or whether each enterprise would define functional roles in a way that reflects enterprise business practices.

Conclusion

Our analysis indicates that the software inventory management technical use case – viewed as a "must-have" across the CPE community – is well supported by CPE Version 2.1. However, CPE 2.1 supports other uses – network-based discovery, forensic analysis, and IT management – incompletely at best. While those uses are important to the enterprise users of CPE-based tools and databases, vendor members of the CPE community frequently prefer to meet those needs in proprietary or enterprise-specific ways.

In hindsight, the development of CPE was focused on the single "must-have" technical use case, software inventory management. Thus, it is not surprising that the specification, and its underlying taxonomy, is working effectively for that use case. Unfortunately, the taxonomy chosen for CPE does not effectively support the use cases that require roll-ups to functional roles or general asset types. The conflicts in the community have arisen as members for whom those use cases are paramount try to use CPE for those cases, because it exists and because it comes close to meeting their needs. We believe that considerable effort will be required to achieve consensus on taxonomies for the unsupported use cases.

We believe that attempts to modify the specification – to enable roll-up of more fully qualified CPE Names into names for functional roles or general asset types – will break the specification in one of two ways. First, the specification could become indeterminate: different tool vendors will roll up more fully qualified CPE Names into higher level names in different ways, thus failing to support interoperability or comparison of results. Second, members of the CPE community could drop out, finding that the taxonomy of functional roles or asset types used to define the roll-up is incompatible with the view they have incorporated into their proprietary user interfaces, tools, or databases.

The CPE community, which includes the Government organizations that fund the maintenance of the CPE Specification and the Official CPE Dictionary along with the commercial vendor organizations that participate in the CPE Working Group, needs to decide whether to restrict CPE to the single technical use case that is most needed (and create additional standards for the other technical use cases), or to expand or modify the CPE Specification and Dictionary to support those other use cases. This decision will help us solve the outstanding issue related to the software inventory management technical use case.

Appendix A – Interview Questions

Ten interviews were performed from July 28 through September 18. The following questions were provided to interviewees to guide conversations about their current or intended use of CPE.

- CPE AND YOUR PRODUCTS
    - How does your company classify or describe your product in terms of the product category it is a member of? How is it classified by IDG or Gartner?
    - Can you describe the basic functionality of your product in general terms? What kind of work is being done by a user when he or she is using your product?
    - What kind of internal data elements contain or are related to CPE Names? e.g. configuration checks? vulnerability checks? remediations? systems?
    - For each type of data element that is associated with a CPE name, can you provide examples of actual CPE names that you use? That is, how deep into the name structure will you process when processing against this data element?
    - What are the relationships between your data elements and the CPE Names and are these relationships implicit or explicitly defined? e.g. "applies to"
    - Can you describe any use of matching or level-of-abstraction translation?
- CPE AND YOUR USERS
    - How/When do your users interact with CPE in your product / GUI?
    - Will users be presented with CPE Names or different display names?
    - For each occasion that a user interacts with a platform name, can you give examples of actual CPE names that are being processed, even if the display name is different?
    - Will users be able to search based on platform name and if so, how does this relate to your use of CPE names?
    - Can you describe any use of matching or level-of-abstraction translation?
    - Are there ever occasions where the user will interact with "platform" information in which CPEs are not used and if so, why not?
- CPE AND PRODUCT INTEROPERABILITY
    - Will your product or system interact with any other products or systems (either internal to your organization or external) and share platform information?
    - For all such interactions, can you describe the data structures involved and how CPEs will be used?
    - In those cases where CPE names are used, can you specific examples of CPE names that are used?
    - Can you describe the (implicit or explicit) relationships defined by the CPE associations?
    - Can you describe any use of matching or level-of-abstraction translation?
    - In those cases where CPE names are not used, can you describe why not?
- OTHER QUESTIONS
    - Do you use either of the CPE XML schemas? If so, why and how?
    - Do you interact with the Official CPE Dictionary? If so, why and how?
    - If no new CPE Names were produced, how would this affect your product?

Appendix B – Interviews

July 28 - McAfee
July 28 - NIST (National Software Reference Library)
July 30 - nCircle
July 31 - Skybox Security
July 31 - Palmadia
August 11 - Red Hat
August 12 - HP
August 14 - BigFix
August 21 - NIST (Nation Vulnerability Database)
September 18 - NSA (data pilot)