Common Platform Enumeration (CPE) – Name Format and Description

Andrew Buttner, MITRE Corporation
Todd Wittbold, MITRE Corporation
Neal Ziring, National Security Agency

Version 1.1 - 13 March 2007

Contents

Approved for Public Release; Distribution Unlimited.
07-0030

©2007 The MITRE Corporation. All rights reserved.

1. Introduction

Following security best practices are essential to maintaining the security of IT systems. To this end, several specification languages exist for describing vulnerabilities, testing system state, and expressing security checklists. But descriptions of vulnerabilities and configuration best practices have greater utility when all participants share common names for the entities described. Further, use of consistent and meaningful names can speed application development, foster interoperability, improve correlation of test results, and ease gathering of metrics.

Today, a popular and widespread naming scheme exists for vulnerabilities; the Common Vulnerabilities and Exposures (CVE) naming scheme is widely used for identifying and describing IT system vulnerabilities. A somewhat similar scheme has been recently introduced for secure configuration best practices: the Common Configuration Enumeration (CCE).

All vulnerability and configuration information items have an important distinction that affects their use: they apply only to a particular range of IT systems, platforms, or applications. This is so obvious that IT managers and security administrators sometimes forget about how critical it can be. When a new vulnerability is announced, the first question most practitioners will ask is: "which systems are vulnerable?" In prose vulnerability descriptions, informal or colloquial names for IT platforms are adequate. Experienced system administrators and security analysts can understand and use ad hoc names.

There is a strong trend toward automation in security practice. Automated systems cannot work with informal or ad hoc names. To foster effective automation, the community needs a more formal naming scheme, consistent and uniform, that allows tools (as well as human analysts and authors) to clearly identify the IT platforms to which a vulnerability or element of guidance applies.

This note describes a structured naming scheme for IT systems, platforms, and packages: the Common Platform Enumeration (CPE). It is based on the generic syntax for Uniform Resource Identifiers [2]. The CPE specification includes the naming syntax, conventions for constructing CPE Names from product information, a matching algorithm, and an XML schema for binding descriptive and diagnostic information to a name. For the list of official CPE Names and for the complete list of abbreviations and formatted names to be used, please visit the CPE web site at:

`http://cpe.mitre.org/`

Using a clear and uniform naming specification, community members will be able to generate names for new IT platforms in a consistent and predictable way.

©2007 The MITRE Corporation. All rights reserved.

2. Requirements

For naming IT platforms subject to vulnerability and configuration guidance, we need to be able to describe three distinct facets.

- *Hardware Platform* – the physical platform supporting the IT system. The type and model of hardware can be relevant for some guidance and vulnerabilities.

- *Operating System Platform* – the operating system controls and manages the IT hardware and supports applications. The operating system type, version, edition, and upgrade status are almost always relevant for vulnerability descriptions and guidance.

- *Application Environment* – software systems, servers, and packages installed on the system are often relevant for vulnerability and guidance. The diversity of applications that may be installed on a modern IT platform is very great, but typically a specific piece of guidance or a specific vulnerability description depend on only one or two applications.

A CPE Name must be able to express any or all of these three facets. If a facet is irrelevant, then the name format must allow it to be omitted.

Security guidance information and vulnerability information apply to a wide range of product and system categories. For example, some vulnerabilities affect an entire product line, while others affect only a particular product release or version. Some guidance applies to entire product family or system type, but other guidance can apply only to a particular version of an application running on a particular OS release. CPE must be able to express platform information across a wide range of specificity.

Along with expressive names, CPE must allow some means to specify concrete diagnostic tests. In the current specification, a CPE Name can include a link to a check written in the Open Vulnerability and Assessment Language (OVAL [3]); this check can be used, with respect to an IT system, to determine whether the system is an instance of the named platform.

A general requirement for the naming structure is that the set of platforms identified by a long name should be a subset of the set of platforms identified by a shorter initial portion of that same name. This is called the "prefix property". Each facet of a CPE Name should exhibit the prefix property.

Additionally, CPE Names must be formalized such that there is a single naming convention producing a single abbreviation for relevant terms. In other words, CPE must enforce the creation of a single, unique name for a given platform instead of allowing multiple different names that all mean the same thing.

Finally, CPE Names do **not** express aspects of system configuration. Many IT vulnerabilities and guidance items depend on configuration, but that is not part of what CPE provides.

©2007 The MITRE Corporation. All rights reserved.

3. CPE Names

This section explains the syntax and usage for CPE Names, and provides several illustrative examples.

3.1. Basic Structure

A CPE Name is a URI [2]. Each name consists of the prefix "cpe:" and up to three major parts. Each major part begins with a slash, and may contain 0 or more elements. (Each element is a named hardware or software constituent of the defined platform.) Multiple elements within a part must be separated by semicolons.

Figure 1 shows the basic syntax of a CPE Name. The first slash must appear, the second and third may be omitted if their respective parts are empty.

Figure 1 – CPE Name Structure

```
cpe:/ {hardware-part} [ / {OS-part} [ / {application-part} ] ]
```

The three parts are:

- *Hardware Facet Specification* - This part identifies hardware aspects of the IT platform. Each element in the hardware part denotes a chassis, module, card, or other physical aspect of a specified platform. This can include virtualized or shared hardware. If a CPE Name's hardware part is empty, then the name corresponds to all hardware platforms.

- *Operating System Facet Specification* - This part identifies operating system aspects of the IT platform. Each element in the OS part describes an operating system running in the platform. For multi-processors devices and distributed systems that incorporate multiple operating systems, multiple elements may appear to indicate the various operating systems used. If a CPE Name's OS part is empty, then the Name corresponds to all operating systems.

- *Application Environment Facet Specification* - This part identifies relevant applications, services, and packages installed on the IT platform. Each element indicates a particular relevant software package. If a CPE Name's application part is empty or absent, then the name corresponds to any combination of applications.

Each element in a part consists of one or more components, separated by colons. The first component names the vendor or supplier of the element; subsequent components provide additional information. Section 3.2 defines the syntax of elements and components fully. The order of elements in a part is not significant; although the order of the components that make-up an element is significant. Duplicate elements may not appear.

©2007 The MITRE Corporation. All rights reserved.

Examples

The simplest CPE Name consists of an empty hardware part, and no OS or application part. The example name below refers to all IT systems.

```
cpe:/
```

A typical CPE Name might specify only an operating system platform, and leave the hardware part empty. The example name below refers to the Microsoft Windows™ 2000 operating system, all editions and update levels.

```
cpe://microsoft:windows:2000
```

A more specific CPE Name might indicate an operating system and a server application. The example name below refers to RedHat Linux Application Server 3, running the Apache Foundation HTTP server version 2.0.52.

```
cpe://redhat:enterprise_linux:3:as/apache:httpd:2.0.52
```

The CPE Name for a network device might include both a hardware part and an operating system part. The example name below refers to a Cisco model 3825 integrated services router running Internet Operating System™ (IOS) version 12.3, enterprise edition.

```
cpe:/cisco::3825/cisco:ios:12.3:enterprise
```

It is quite possible that a vendor, perhaps for marketing purposes, may use different names for identical hardware and software platforms. This could occur for regional versions with different languages, or different vertical markets each with a need for the same tool. In these cases, a different CPE name will be used for each of the different vendor names. CPE in general tries to mimic the product naming structures used by the vendors, while imposing uniform structure and matching semantics.

A CPE Name may be used for an application alone; in that case the hardware and OS parts will be empty. The example name below refers to a particular web browser, regardless of any hardware or particular OS on which it is running.

```
cpe:///microsoft:ie:6.0
```

A CPE Name can specify multiple alternative constituents inside a component by using the exclamation point (the OR operator). The example below specifies a platform with an operating system of either Microsoft Windows XP or Vista.

```
cpe://microsoft:windows:xp!vista
```

It is also possible to use the tilde character '~' to designate every constituent except the one specified (the NOT operator). For example, the following CPE Name identifies every Microsoft Windows system except those running XP.

```
cpe://microsoft:windows:~xp
```

©2007 The MITRE Corporation. All rights reserved.

Note that the OR and NOT operators may not be used together; at most one of them may be used in any single component.

Each part of a CPE Name may include multiple name elements, separated by semicolons. The order of elements in a name part is not significant; elements may appear in any order. The example name below refers to a Sun Solaris™ 9 system with both a database manager and a web application server installed.

```
cpe://sun:sunos:5.9/bea:weblogic:8.1;mysql:server:5.0
```

3.2. Element and Component Syntax

The syntax of CPE Names is tightly controlled to keep enable unique names to be maintained. One of the requirements of CPE is that there be only one way to state specific platform name and to accomplish this, the exact syntax and abbreviations used must be strictly enforced. This section tries to explain the logic that is followed when creating a new CPE Name. It is intended to help keep new CPE Names following a consistent naming convention. Unfortunately, due to the sheer size of possible CPE Names, the entire list of CPE Names can not be included. Please visit the CPE web site for an up to date list of official CPE Names.

Syntax

An element within a CPE Name identifies a particular piece of a platform. Each element consists of one or more components separated by colons. The first component in an element always identifies the supplier or vendor of the platform element. Subsequent components generally express model, version, release, edition, and update information. (Note: the order of components within an element is significant.)

The vendor component of an element might be a source of ambiguity in CPE Names, because there are many ways to express the names of companies and other organizations. For CPE, the name used for a supplier should be the highest organization-specific label of the organization's DNS name [4]. Even if the domain name is different than the company name, it is still recommended to use the domain name for the CPE Name. The table below shows some representative examples.

Organization Full Name	DNS Domain	CPE component
Cisco Systems, Inc.	cisco.com	cisco
The Mozilla Foundation	mozilla.org	mozilla
University of Oxford	oxford.ac.uk	oxford

CPE names are case-insensitive. The components “WebLogic” and “weblogic” are equivalent. To reduce potential for confusion, all CPE Names should be written in lowercase.

Components themselves can contain one of two different special characters that help specify the exact platform intended by the CPE Name. The exclamation point '!' can be used similar to an

©2007 The MITRE Corporation. All rights reserved.

OR operator. A component that includes the exclamation point will match either of the constituents on either side of the exclamation point. The tilde '~' can be used to specify every constituent except the one specified. See the examples in the previous section.

Empty components are legal; they designate a part of the platform name for which any aspect of the platform is valid. For example, the CPE Name below designates all editions of Microsoft Windows 2000, with service pack 4 installed.

```
cpe://microsoft:windows:2000::sp4
```

Abbreviations

Some words and phrases appear frequently in IT product designations. Including these common words in CPE Names would be profligate, and would hinder the objective of making CPE Names compact and easy to use. To help shorten some of the longer CPE Names, abbreviations are used where appropriate. Note that each name should only be associated with a single abbreviation, but a specific abbreviation may apply to multiple names. For example, both 'service pack' and 'support pack' are abbreviated 'sp', but that is the only abbreviation that is allowed for both of these names.

The table below lists abbreviations for multi-word constructions that should be applied in selecting CPE Name components. For an up to date list of abbreviations, please visit the CPE web site at http://cpe.mitre.org/.

Full Designation	CPE Abbreviation	Remarks
version 3.4	**3.4**	Version numbers have special semantics for matching.
release 3	**r3**	Vendor releases are typically abbreviated this way.
release candidate 2	**rc2**	Release candidates are common for open source software.
service pack 4	**sp4**	Service pack designations are usually depicted in this way.
support pack 2	**sptp2**	The word "support" should be abbreviated with "spt".
home edition	**home**	The word “edition” may be dropped when it appears last.
professional edition	**pro**	This word appears in the names of many IT products, and is commonly abbreviated.

©2007 The MITRE Corporation. All rights reserved.

Full Designation	CPE Abbreviation	Remarks
service release 2	**sr2**	Service releases are typically abbreviated in this way.
security rollup	**sr**	This abbreviation seems to be specific to Microsoft Windows.
advanced server	**advanced**	The word "server" may be dropped when it appears last.
standard edition	**std**	This abbreviation is very common.

Multi-word product names may be abbreviated when doing so would not make the CPE Name ambiguous and when the vendor has designated a particular "official" abbreviation in product descriptions. For example, "Internet Explorer" should be abbreviated as "ie", and "Java Runtime Environment" should be abbreviated as "jre". A list of community product name abbreviations will be maintained at the CPE web site.

Otherwise, multi-word product names and designations should be spelled out in full, replacing spaces with underscores. The example below shows how this would look for the Zone Labs ZoneAlarm Internet Security Suite version 7.0.

```
cpe:///zonelabs:zonealarm_internet_security_suite:7.0
```

Except for those listed in Appendix C, single-word name components should not be abbreviated.

Prohibited Characters

A component may not include the following characters:

- colon (**:**) – not permitted in a component because it is used to separate the components in a name element.
- semicolon (**;**) – not permitted in a component because it is used to separate the name elements in a CPE Name part.
- slash (/) – not permitted in a component because it is used to separate CPE Name parts.
- percent-sign (**%**) – not permitted because it is used for character encoding in URIs.

For more information, see Appendix B.

©2007 The MITRE Corporation. All rights reserved.

3.3. Hardware Part Element Names

An element of the hardware part consists of one or more components. The meanings of the first three components are defined by this specification; additional components may be added after the pre-defined ones.

Figure 2 – Hardware Part Element Structure

```
{vendor}:{family}:{model}
```

The first component must be the vendor or supplier of the hardware element. This should be expressed as the organization-specific label from the DNS domain name for the organization or company.

The second component should be the hardware family designation. Many vendors identify product lines or product families with a distinctive name; that name should be used here. If a particular product does not have such a name, this component may be left empty.

The third component should be the hardware model designation. Most hardware vendors use some internally consistent system of model numbers; the full model designation should be used here.

Additional components can be used to describe specifics related to an individual piece of hardware. These components are not defined within CPE as they may be different for each type of hardware.

Hardware Examples

The example name below identifies a particular laptop computer hardware platform. The supplier is Dell Computer, the product line name is "Inspiron", and the model number is 8500.

```
cpe:/dell:inspiron:8500
```

Multiple hardware elements may be used when multiple distinct hardware items are relevant. The example CPE Name below identifies a network router with a cryptographic accelerator module. Note that order of elements is not significant.

```
cpe:/juniper:m-series:m7i;juniper:es-pic
```

Hardware elements may also be used to identify virtual hardware, in the context where the virtual hardware platform serves the same role as a similar real hardware platform. In such cases, the software version number of the virtual hardware system may be used as the model designation. The CPE Name below shows an example for a virtual hardware platform.

```
cpe:/emc:vmware:esx:2.5
```

©2007 The MITRE Corporation. All rights reserved.

3.4. OS Part Element Names

An element of the OS part consists of one or more components. The meanings of the first three components are defined by this specification. Additional components may be added after the pre-defined ones to define additional OS-specific information. Operating systems are particularly difficult to identify with names, because each vendor follows a distinct scheme for naming their product releases.

Figure 3 – OS Part Element Structure

```
{vendor}:{family}:{version}
```

The first component must be the vendor or supplier of the operating system element.

The second component must be the operating system family name or generic name.

The third component must be the operating system version number or the major release designator.

Additional components can be used to describe specifics related to an individual OS. These components are not defined within CPE as they may be different for each type of OS. For example, some operating systems may distinguish between different editions, and others might have update levels or sub-release distinctions. A component might also be used to identify the particular language version of an OS.

OS Examples

The example CPE Name below identifies Microsoft's Windows XP operating system, Professional Edition, update level "Service Pack 2".

```
cpe://microsoft:windows:xp:pro:sp2
```

The example name below identifies Apple's "Tiger" MacOS X release 4.

```
cpe://apple:macos:10.4
```

The example name below identifies the Fedora Project "Fedora Core" Linux distribution, release 6.

```
cpe://fedoraproject:fedora_core:6
```

The example name below identifies any edition of Microsoft Windows 2000, but with the update level of "Service Pack 4".

```
cpe://microsoft:windows:2000::sp4
```

©2007 The MITRE Corporation. All rights reserved.

3.5. Application Part Element Names

An element of the application part consists of two or more components. The meanings of the first four components are defined by this specification. Additional components may be added after the pre-defined ones.

Figure 3 – Application Part Structure

```
{vendor}:{name}:{edition}
```

The first component must be the vendor or supplier of the application, server, or software package. In cases where an application is an extension by vendor A of a system created by developer B, then domain name for vendor A should be used as the basis for the first component of the element.

The second component must be the application product family name or specific product name. Some small organizations use the same organization name and product name. In such cases, the name should appear in both components.

The third component, must be an edition designation, version number, release designation, or update level.

Additional components may be used to describe specifics related to an individual applications.

Application Examples

The example name below identifies the JBoss 4.0.4 application server.

```
cpe:///jboss:jboss:4.0.4
```

The example name below identifies the Oracle Database server 10g, release 2.

```
cpe:///oracle:database:10g:r2
```

The example name below identifies the Adobe Acrobat™ version 6.0 document generation application, Standard edition or Professional edition.

```
cpe:///adobe:acrobat:6.0:std!pro
```

©2007 The MITRE Corporation. All rights reserved.

4. Matching

When a checking or analysis tool tests a real IT target system, it must be able to decide whether any given CPE Name corresponds to that system, and vice-versa. This is called *matching*. If a CPE Name has an associated OVAL Definition, and the definition evaluates to true, then the system is said to be an *instance* of that name. But not every CPE Name will have an OVAL Definition associated with it. This is especially true if a CPE Name has been created by an individual author or application to identify a complex platform. For these cases, there is a defined process for how to match a CPE Name against an actual system based on other known CPE Names (ones that have been matched via an OVAL Definition).

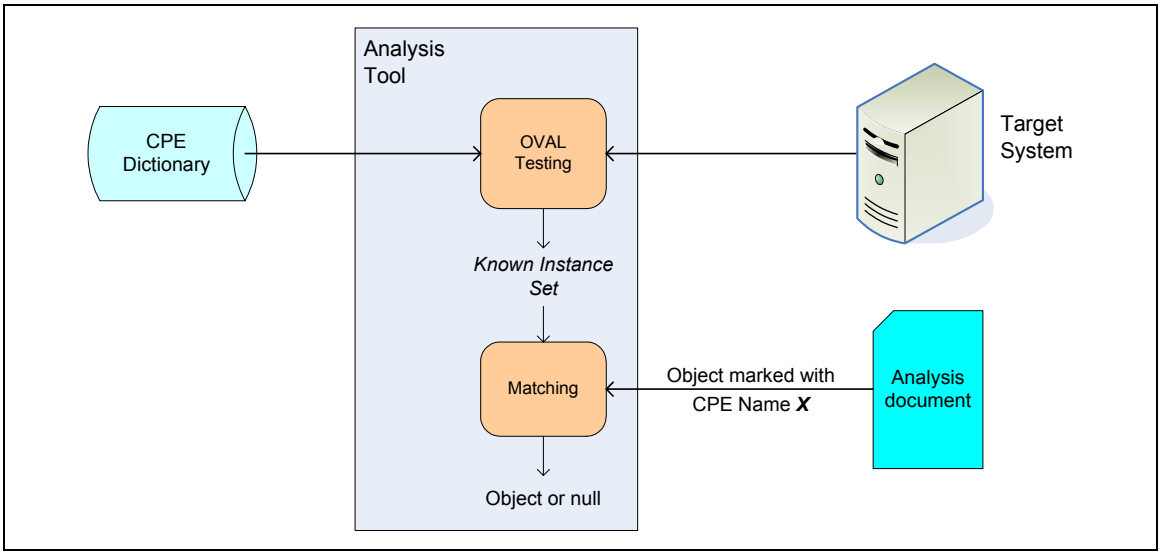
The conceptual model for matching consists of two steps:

1. Make a list of all the elements in the CPE Name.

2. For each element, check whether the target system has the hardware, software, or operating system indicated by the element. If the check succeeds for all elements, then the target is an instance of the CPE Name.

While the conceptual model is very simple, it will not be practical for all analysis tools. It is very difficult, in general, to check the constituent parts of an IT system. In more realistic situations, analysis tools will be supplied with standard dictionaries of CPE Names for common hardware, OS, and application products. Using these dictionaries, and the OVAL Definitions included in them, an analysis tool can assemble a list of CPE Names of which the target platform is known to be an instance. Then, given any arbitrary CPE Name *X*, the tool can ascertain whether or not the target is an instance of *X*. This is called known instance based matching. Figure 4 illustrates the basic concept of known instance based matching; a more formal description is given below.

Figure 4 – Conceptual Model for Known Instance Based Matching



Note: there is no mechanism in CPE to stipulate platform qualification based on the absence of a system constituent. All matching is positive, not negative. All CPE Names are converted to

©2007 The MITRE Corporation. All rights reserved.

lowercase before matching. Further, in the algorithm below, the CPE Names that belong to the set of know instance CPE Names may not use the OR operator (!) or the NOT operator (~).

4.1. Matching Algorithm: Known Instance Based Matching

This algorithm accepts a set of known instance CPE Names and a candidate CPE Name, and delivers the answer 'true' if the candidate can be shown to be an instance based on the content of the known instances. Otherwise, it returns 'false'.

Inputs:

- A set of *m* known CPE Names $\boldsymbol{K} = \{K_1, K_2, \ldots, K_m\}$.
- A candidate CPE Name *X*.

Output:

- True if *X* matches ***K***, false otherwise.

Algorithm:

```
answer ← true.
FOREACH  k  IN  set K  DO
    IF k = X THEN RETURN answer.
END.
P ← set {hardware, os, application}.
FOREACH  p  IN  set P  DO
    E ← set of elements in part p of name X.
    FOREACH  e  IN  set E  DO
         found ← false.
         C ← vector of components in e.
         FOREACH  k  IN  set K  DO
             F ← set of elements in part p of name k.
             FOREACH  f  IN  set F  DO
                    D ← vector of components in f.
                    match ← true.
                    FOREACH  c,d  IN  sets C,D  DO
                        match ← false.
                        IF  c is empty  THEN match ← true.
                        ELSE IF c is a singleton AND c = d  THEN match ← true.
                        ELSE IF  c has form ~v AND v != d THEN match ← true.
                        ELSE IF  c has form v1!v2!..!vn  AND  v = d for some v  THEN
                        match ← true.
                        ENDIF.
                    END.
                    IF  match = true  THEN
                       found ← true.
                    ENDIF.
             END.
         END.
    END.
    IF  found = false  THEN
          answer ← false.
    ENDIF.
END.
RETURN answer.
```

©2007 The MITRE Corporation. All rights reserved.

The comparisons in the CPE matching algorithm are all string equality tests. The known CPE Names and the candidate CPE Name must be converted to lowercase before matching.

4.2. Matching Examples

The examples below illustrate known instance based matching.

In the first example, testing OVAL definitions from a CPE dictionary against a target system results in three CPE Names. These CPE Names are collected into the set ***K*** of known instances.

K = {"cpe://microsoft:windows:2000",
"cpe://microsoft:windows:2000:pro:sp1",
"cpe:///microsoft:ie:5.5"}

A rule in a security guidance checklist describes some settings to check on a system running Microsoft Windows 2000 with service pack 1 and Internet Explorer version 5.5. The rule is marked with a CPE Name indicating those two aspects of the platform to which the rule applies. That name is called the candidate name *X*.

X = "cpe://microsoft:windows:2000::sp1/microsoft:ie:5.5"

There are two elements in *X*, e_1="microsoft:windows:2000::sp1" and e_2="microsoft:ie:5.5". Stepping through the matching algorithm, we see that e_1 matches the second instance in ***K***, and *e*2 matches the third instance in ***K***. So, the algorithm returns true and the rule can be applied to the target system.

In the second example, testing OVAL definitions from a CPE dictionary against a target system results in two CPE Names.

K = {"cpe://sun:sunos:5.9",
"cpe://bea:weblogic:8.1"}

A rule in a security vulnerability report designates the vulnerability as applicable for BEA WebLogic application server 8.0 running on Solaris 8 or 9 (SunOS 5.8 or 5.9).

X = "cpe://sun:sunos:5.8!5.9/bea:weblogic:8.0"

There are two elements in *X*, e_1="sun:sunos:5.8!5.9" and e_2="bea:weblogic:8.0". Stepping through the matching algorithm, we see that e_1 matches the first instance in ***K***, but that *e*2 matches nothing in ***K***. So, the algorithm returns false and the vulnerability does not apply to the target system. Element e_2 does not match the second instance in ***K***, because the instance indicates that the platform has WebLogic 8.1, while the vulnerability depends on version 8.0.

©2007 The MITRE Corporation. All rights reserved.

5. CPE Description Format

The CPE description format is an XML document format for binding descriptive prose and diagnostic tests to a CPE Name. The format is defined using a XML Schema[1]; the full schema appears in Appendix A. The schema defines two data types:

- CPE Name list – a list of CPE items.
- CPE item – a CPE Name, with a full name (title), a list of zero or more descriptive notes, a list of zero or more references, and zero or more diagnostic tests.

A CPE description document may be used by IT testing and report generation tools to get friendly names and descriptions for the compact CPE Names used in machine-readable guidance items and vulnerability definitions.

Figure 5, below, shows a short sample CPE description file.

Figure 5 – A Short CPE Description File

```
<?xml version="1.0">

<cpe-list xmlns="http://cpe.mitre.org/XMLSchema/cpe/1.0"
          xmlns:cpe="http://cpe.mitre.org/XMLSchema/cpe/1.0">

   <cpe-item name="cpe://redhat:enterprise_linux:3">
       <title>Red Hat Enterprise Linux 3</title>
   </cpe-item>

   <cpe-item name="cpe://sun:sunos:5.8">
       <title>Sun Microsystems SunOS 5.8</title>
       <notes>
          <note>Also known as Solaris 8</note>
       </notes>
   </cpe-item>

   <cpe-item name="cpe://microsoft:windows:2003">
       <title>Microsoft Windows Server 2003</title>
       <check system="http://oval.mitre.org/XMLSchema/oval-definitions-5">
          oval:org.mitre.oval:def:128
       </check>
   </cpe-item>

   <cpe-item name="cpe:/intel:ia-64:itanium">
       <title>Intel Itanium (IA-64)</title>
   </cpe-item>

</cpe-list>
```

The CPE Description format can also be used to create a community dictionary of common CPE Names.

©2007 The MITRE Corporation. All rights reserved.

6. References

[1] Fallside, D.C., *XML Schema Part 0: Primer*, W3C Recommendation, 2 May 2001.

W3C documents are available from http://www.w3.org/.

[2] Berners-Lee, T., Fielding, R., and Masinter, L., *Uniform Resource Identifier (URI): Generic Syntax*, RFC 3986, January 2005.

Internet RFCs are available from http://www.rfc-editor.org.

[3] "OVAL – The Open Vulnerability and Assessment Language", The MITRE Corporation, September 2006.

This is the OVAL web site, http://oval.mitre.org/.

[4] Mockapetris, P., "Domain Names – Implementation and Specification", RFC 1035, November 1987.

Internet RFCs are available from http://www.rfc-editor.org.

[5] Grobauer, B., "CVE, CME,…CMSI? Standardising System Information", Siemens AG, April 2005.

This paper introduces the Common Model of System Information, a structured format for describing IT platforms.

©2007 The MITRE Corporation. All rights reserved.

Appendix A: CPE Description Format Schema

The schema below specifies the XML format for CPE name descriptions.

```
<?xml version="1.0"  encoding="UTF-8"?>

<xsd:schema
 targetNamespace="http://cpe.mitre.org/XMLSchema/cpe/1.0"
 elementFormDefault="qualified" attributeFormDefault="unqualified"
 xmlns:cpe="http://cpe.mitre.org/XMLSchema/cpe/1.0"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:annotation>
      <xsd:documentation>
        This is an XML Schema for defining names in the
        Common Platform Enumeration, a naming system for
        IT platforms.  Each CPE name has the following URI format:

        cpe:/ [ hardware-part ] [ / [ os-part ] [ / application-part ] ]

        Each part consists of zero or more name elements separated
        by semicolons.  Generally, in a description listing like
        this, at most one name element will comprise each part
        Each name element consists of one or more name components
        separated by colons.  The first name component of each
        name element is a supplier or vendor name, expressed as
        the organization-specific label from the supplier's DNS
        name (e.g. from "microsoft.com" use "microsoft")

        This specification was written by Neal Ziring, with assistance
        from Andrew Buttner, and ideas from Todd Wittbold and other
        attendees at the 2nd Annual NIST Security Automation Workshop
        For more information, consult the CPE specification document
        <version date="2007-01-30">1.0</version>
      </xsd:documentation>
    </xsd:annotation>

    <!-- ********************************************************** -->
    <!-- Top-level Object Elements                                  * -->
    <!-- ********************************************************** -->

    <xsd:element name="cpe-list">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">
          This element acts as a top-level container for CPE
          name items.
        </xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="cpe:cpe-item"
                       minOccurs="1" maxOccurs="unbounded"/>
          <xsd:any namespace="##other" processContents="skip"
                   minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:complexType>
```

©2007 The MITRE Corporation. All rights reserved.

```
    <!-- no duplicate names in a single CPE description file -->
    <xsd:key name="itemURIKey">
      <xsd:selector xpath="./cpe:cpe-item"/>
      <xsd:field xpath="@name"/>
    </xsd:key>
  </xsd:element>

  <xsd:element name="cpe-item">
    <xsd:annotation>
      <xsd:documentation xml:lang="en">
        This element denotes a single name in the Common
        Platform Enumeration.  It has the following parts:
              - name, a URI, which must be a unique key, and
                  should follow the URI structure outlined in
                  the CPE specification.
              - title, arbitrary friendly name
              - notes, optional descriptive material
              - references, optional external info references
              - check, optional reference to an OVAL test that
                  can confirm or reject an IT system as an
                  instance of the named platform.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="title" type="xsd:string"
                     minOccurs="1" maxOccurs="1"/>
        <xsd:element name="notes" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="note" type="xsd:string"
                           minOccurs="1" maxOccurs="unbounded"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="references" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="reference" type="cpe:referenceType"
                           minOccurs="1" maxOccurs="unbounded"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="check" type="cpe:checkType"
                     minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="name" type="xsd:anyURI"
                     use="required"/>
      <xsd:attribute name="deprecated" type="xsd:boolean"
                     use="optional" default="false"/>
    </xsd:complexType>
     <xsd:unique name="checkSystemKey">
        <xsd:selector xpath="./cpe:check"/>
        <xsd:field xpath="@system"/>
     </xsd:unique>
  </xsd:element>
```

©2007 The MITRE Corporation. All rights reserved.

```
    <!-- ********************************************************** -->
    <!-- Supporting Element Types                                   * -->
    <!-- ********************************************************** -->

    <xsd:complexType name="referenceType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
              Type for an reference in the description of a CPE
              item.  This would normally be used to point to extra
              descriptive material, or the supplier's web site, or
              the platform documentation.  It consists of a piece
              of text (intended to be human-readable) and a URI
              (intended to be a URL, and point to a real resource).
            </xsd:documentation>
        </xsd:annotation>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
              <xsd:attribute name="href" type="xsd:anyURI"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>

    <xsd:complexType name="checkType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
            Data type for the check element, a checking system
            specification URI, string content, and an optional
            external file reference.  The checking system
            specification should be the URI for a particular
            version of OVAL or a related system testing language,
            and the content will be an identifier of a test written
            in that language.  The external file reference could
            be used to point to the file in which the content test
            identifier is defined.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
              <xsd:attribute name="system" type="xsd:anyURI"
                             use="required"/>
              <xsd:attribute name="href" type="xsd:anyURI"
                             use="optional"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>

</xsd:schema>
```

This schema has been tested with Altova XMLSpy (`cpe:///altova:xmlspy:2005`), and with the Apache Xerces schema validating parser (`cpe:///apache:xerces-j:2.6`).

©2007 The MITRE Corporation. All rights reserved.

Appendix B: CPE URI Grammar

The BNF grammar below defines the syntax for CPE URIs.

```
cpe-name       ::= "cpe:/"  hw-part  "/"  os-part  "/"  app-part
                   "cpe:/"  hw-part  "/"  os-part
                   "cpe:/"  hw-part

hw-part        ::= vendor  ":"  component-list  ";"  hw-part
                   vendor  ":"  component-list
                   empty

os-part        ::= vendor  ":"  component-list  ";"  os-part
                   vendor  ":"  component-list
                   empty

app-part       ::= vendor  ":"  component-list  ";"  app-part
                   vendor  ":"  component-list
                   empty

vendor         ::= dns-label
                   empty

component-list ::= component  ":"  component-list
                   component
                   empty

component      ::= "~" string
                   term

term           ::= string "!" term
                   string

string         ::= ( ALPHA | DIGIT | PUNC ) +

PUNC           ::= ( "." | "_" | "-" | "," | "(" | ")" | "@" | "#" )
```

Where ALPHA and DIGIT are defined in [2].

©2007 The MITRE Corporation. All rights reserved.

Appendix C: Single Word Abbreviations

The following table lists designated abbreviations for single word names. For an updated table of abbreviations, please visit the CPE web site at http://cpe.mitre.org/.

Full Designation	CPE Abbreviation
advanced	adv
professional	pro
server	srv
standard	std

©2007 The MITRE Corporation. All rights reserved.