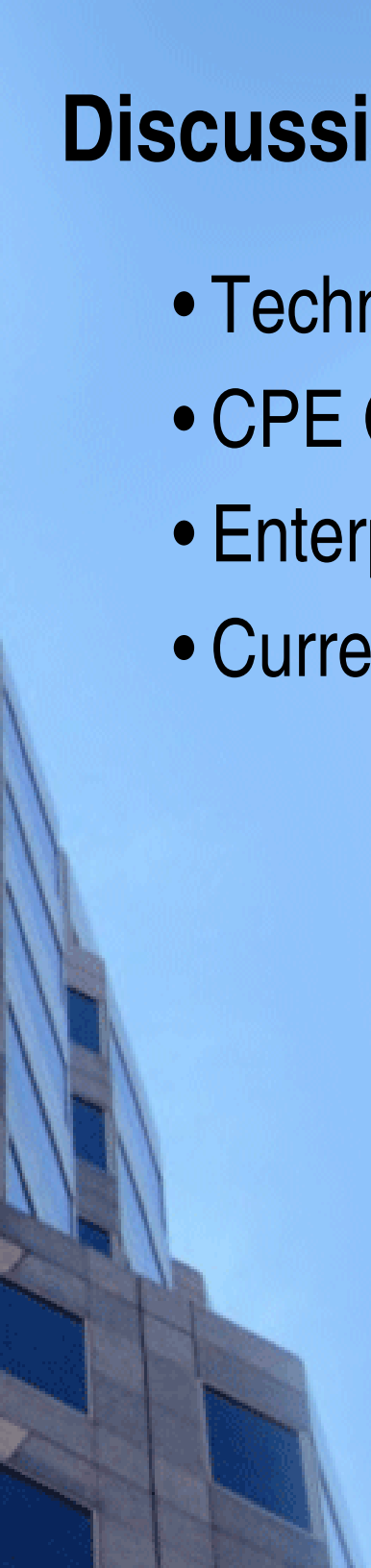
The Common Platform Enumeration (CPE)

September, 2008

David Waltermire

Discussion Points

- Technical Use Cases
- CPE Overview
- Enterprise Use Cases
- Current Issues



Technical Use Cases

- Identification
- Matching and Querying
- Product inventory

CPE provides a standardized naming scheme for products allowing identification

- All applications share a common product vocabulary allowing interoperability
- Allows identification of products at a standardized level of granularity
- Data can be associated with products by referencing a CPE Name

CPE provides powerful querying capabilities

- Allows searching of products based on abstract CPE Name based search criteria
- The CPE Language provides matching capabilities using logical groupings of products

CPE provides automation capabilities for asset inventory

- Use of inventory definitions provides a technical mechanism for determining the presence of products on an asset
- Mappings to/from CPE names allows integration into legacy architectures that do not speak CPE

CPE Overview

- CPE Name Format
- CPE Name matching and the CPE Language
- CPE Dictionary

A CPE name is a special type of URI

The URI scheme

- Identifies that the URI is a CPE name
- The “cpe” scheme has not been registered with IANA

The scheme specific part

- Uses special syntax specific to CPE
- A URI may contain only ASCII characters
- Hierarchical by nature
- Each component is separated by a colon

cpe:/ {part} : {vendor} : {product} : {version} : {update} : {edition} : {language}

The part component classifies the CPE name

Possible values are:

h – Hardware
o – Operating System
a – Application

cpe:/ **{part}** : {vendor} : {product} : {version} : {update} : {edition} : {language}

The vendor component is the supplier of the product

- Each vendor organization has a unique name
- Generally represents the highest organization-specific label of the organization's DNS name
- Products developed by individuals outside of an organization can use the creator's name

Organization's Full Name	DNS Domain	Vendor Component
The National Institute for Standards and Technology	nist.gov	nist
Acme Corporation	acme.com	acme
The Acme Organization	acme.org	acme.org
John Doe		john_doe

cpe:/ {part} : **{vendor}** : {product} : {version} : {update} : {edition} : {language}

The product component is the name of the product

- Generally represents the most common and recognizable name for the product
- Multi-word names should be spelled out in full, replacing spaces with underscores “_”

For example:

- application_server
- linux_kernel
- windows_xp

cpe:/ {part} : {vendor} : **{product}** : {version} : {update} : {edition} : {language}

The version component is the version of the product

- Should be the same format as what is seen within the product and on the system

For example:

- 5.1
- 2.1.4.254

cpe:/ {part} : {vendor} : {product} : **{version}** : {update} : {edition} : {language}

The update component represents a sub-release of a specific product version

- Used to represent beta, release candidates and service packs
- The “ga”, for *general availability,* placeholder may be used to represent an initial release without an update specified

For example:

- ga
- beta2
- rc1
- sp3

cpe:/ {part} : {vendor} : {product} : {version} : **{update}** : {edition} : {language}

The edition component represents a specific flavor of a product

- Often used to represent the target OS/software, architecture, and/or feature set of a product

For example:

- x86
- x64
- linux_i386
- professional

cpe:/ {part} : {vendor} : {product} : {version} : {update} : **{edition}** : {language}

The language component indicates a language specific release of a product

- Any valid language tag defined by the IETF RFC 4646
- Generally only language and region codes are necessary

For example:

- en_US – US English
- en_GB – UK English
- es – Spanish
- ja – Japanese
- zh - Chinese

cpe:/ {part} : {vendor} : {product} : {version} : {update} : {edition} : **{language}**

Matching is used to determine if two CPE names refer to the same set of products

- Applies a recursive algorithm that evaluates the CPE names hierarchical structure
- Blank components match any value

For example:

cpe:/o::linux_kernel:2.6.27::i586

Would match:

cpe:/o:kernel.org:linux_kernel:2.6.27:rc6:i586
cpe:/o:fedora:linux_kernel:2.6.27:rc1:i586
cpe:/o:redhat:linux_kernel:2.6.27:ga:i586

The CPE Language allows arbitrary logical groupings of CPE names to be evaluated using the matching algorithm

- Defines a collection of products
- Uses CPE name matching for evaluation

For example:

```
<cpe:platform id="abc123">
  <cpe:title>Microsoft Windows XP SP3 x64 Edition, US English release
    AND Microsoft Internet Explorer 7.0 Beta 3</cpe:title>
  <cpe:logical-test operator="AND" negate="FALSE">
    <cpe:fact-ref name="cpe:/o:microsoft:windows_xp::sp3" />
    <cpe:fact-ref name="cpe:/a:microsoft:ie:7.0
  </cpe:logical-test>
</cpe:platform>
```

Would match the set of products:

cpe:/o:microsoft:windows_xp::sp3:x64:en_US
cpe:/a:microsoft:ie:7.0:beta3

The CPE Dictionary is an enumeration of CPE Names

- Currently contains 15,000+ CPE names
- Represents 3000+ products from 200+ vendors

The CPE Dictionary is a large XML catalog

```
<cpe-item name="cpe:/a:microsoft:.net_framework:2.0">
  <title xml:lang="en-US">Microsoft .NET Framework 2.0</title>
  <check
    system="http://oval.mitre.org/XMLSchema/oval-definitions-5"
    href="http://oval.mitre.org/repository/data/DownloadDefinition...">
      oval:org.mitre.oval:def:310</check>
  <meta:item-metadata modification-date="2008-04-15T19:55:43.797-04:00"
    status="FINAL" nvd-id="61877" />
</cpe-item>
```

CPE Name

Internationalized Title

Check Reference

Repository Metadata

The CPE Dictionary also contains component metadata

```
<meta:component-tree>
  <meta:vendor value="adobe">
    <meta:title xml:lang="en-US">Adobe Systems
  Incorporated</meta:title>
    <meta:product value="acrobat_reader" part="a">
      <meta:title xml:lang="en-US">Acrobat
  Reader</meta:title>
      <meta:version value="7.0" />
      <meta:version value="7.0.1" />
      <meta:version value="7.0.2" />
      <meta:version value="7.0.3" />
      <meta:version value="7.0.4" />
      <meta:version value="7.0.5" />
      <meta:version value="7.0.6" />
      <meta:version value="7.0.7" />
      <meta:version value="7.0.8" />
      <meta:version value="7.0.9" />
      <meta:version value="8.0" />
      <meta:version value="8.1" />
    </meta:product>
  </meta:vendor>
</meta:component-tree>
```

Vendor

Product

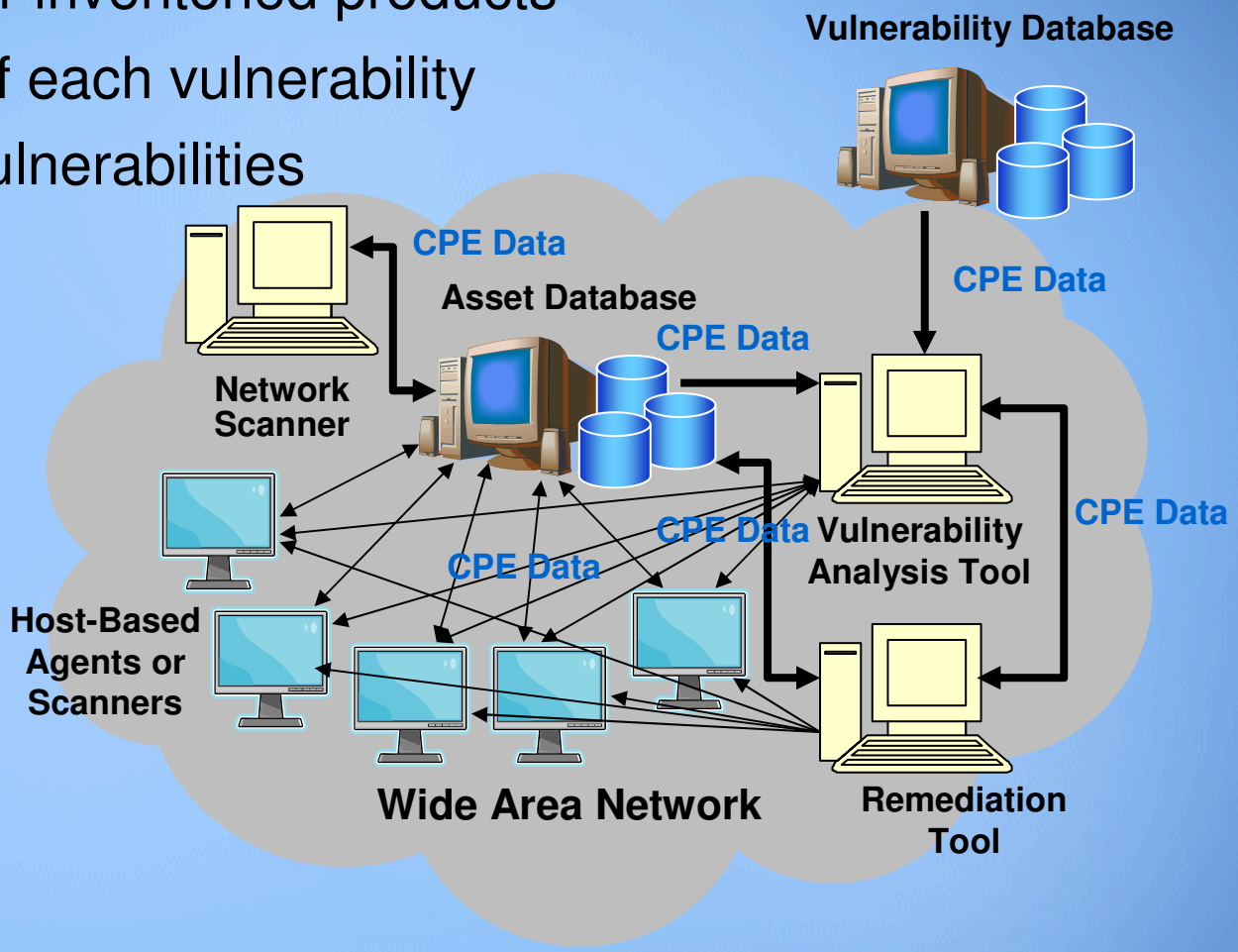
Versions

Enterprise Use Cases

- Vulnerability Management
- Configuration Management
- Asset Reporting

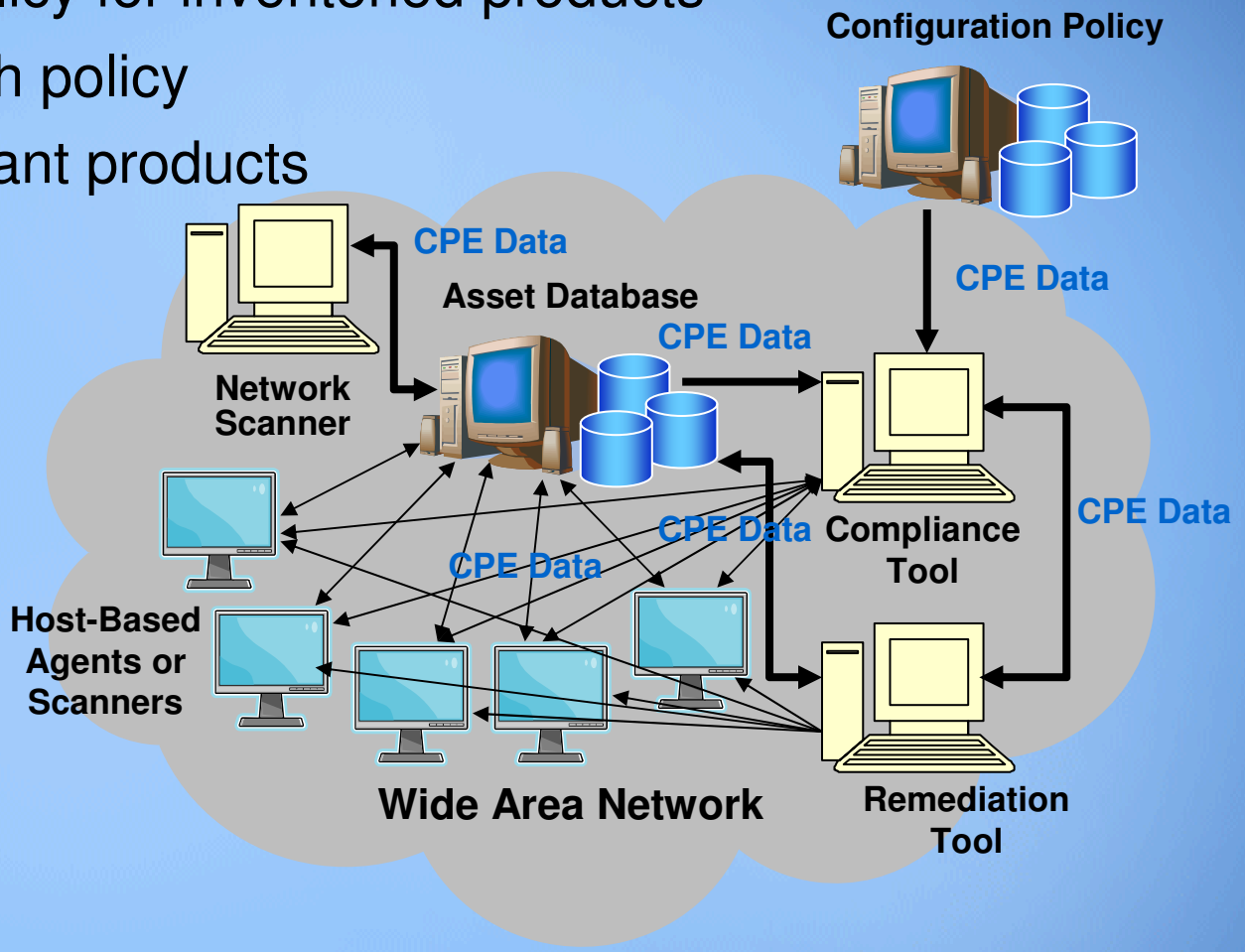
Vulnerability Management

1) Inventory assets to collect deployed products
2) Query vulnerabilities for inventoried products
3) Assess the presence of each vulnerability
4) Remediate identified vulnerabilities
5) Re-assess



Configuration Management

1) Inventory assets to collect deployed products
2) Query configuration policy for inventoried products
3) Assess compliance with policy
4) Remediate non-compliant products
5) Re-assess



Asset Reporting

- CPE Names identify products that compose an asset

- Metadata can be associated with CPE names to identify:
  - Function of a product (i.e. web server, DNS server, etc.)
  - Existence of product vulnerabilities
  - Product configuration compliance
  - Product license usage

Current Issues

- Fully qualified CPE Names
- Complexity of the specification
- Version matching
- Tagging
- Non-computing CPE Names

Problem: Fully Qualified CPE Names are needed for product identification

The CPE Name:

```
cpe:/a:sun:staroffice:8.0
```

Matches ALL updates, editions, and languages

Solution: Differentiate between fully qualified and abstract CPE names

- All components used
- Use of “nil” for unused components
- Add discrete=“truelfalse” metadata tag to differentiate fully qualified vs. abstract CPE names

Now the CPE Name:

```
cpe:/a:sun:staroffice:8.0:nil:nil:nil
```

Matches NO updates, editions, and languages

Problem: The CPE specification contains many parts that change independently of each other

- CPE Name
- CPE Matching
- CPE Language
- CPE Dictionary

- Each capability within CPE is at a different maturity level
- Clarifications regularly needed on CPE naming conventions
- The CPE Name specification should not imply that the only valid CPE names are those specified in the dictionary

Solution: Decompose the CPE capabilities into multiple specifications

Modularize the CPE specification into multiple specifications:

- CPE Name
- CPE Matching
- CPE Language
- CPE Dictionary

- Allows each specification to evolve at different intervals

Problem: Versions in CPE Names exist at multiple levels of granularity

For CPE Names:

cpe:/o:redhat:linux_kernel:2.6:beta1

cpe:/o:redhat:linux_kernel:2.6.1:ga

cpe:/o:redhat:linux_kernel:2.6.12:rc1

The abstract CPE name:

cpe:/o:redhat:linux_kernel:2.6

Matches:

cpe:/o:redhat:linux_kernel:2.6:beta1

Solution: Allow wildcard matching of versions in CPE Names

Allow the matching operations:

- Begins with – foo*
- Ends with – *foo
- Contains – *foo*

For the CPE Names:
cpe:/o:redhat:linux_kernel:2.6:beta1
cpe:/o:redhat:linux_kernel:2.6.1:ga
cpe:/o:redhat:linux_kernel:2.6.12:rc1

The abstract, wildcard CPE name:
cpe:/o:redhat:linux_kernel:2.6*

Matches:
cpe:/o:redhat:linux_kernel:2.6:beta1
cpe:/o:redhat:linux_kernel:2.6.1:ga
cpe:/o:redhat:linux_kernel:2.6.12:rc1

Problem: The need exists to query products using categorizations and other metadata

- Function – Services the product provides (i.e HTTP, FTP, DNS, etc)
- Role – Product use cases (i.e. Domain Controller, Caching DNS server)
- Release Date
- End of lifecycle
- Supersession
- Runs on another product
- Part of another product (e.g. Word is part of Office)
- Distributed with another product

Solution: Allow additional metadata to be assigned with CPE Names using tagging

- Support tagging in the CPE Dictionary
  - Declarative model of tagging
    - Datatype (e.g. string, date, integer, decimal)
    - Enumerate allowed values
  - Associate tags with CPE components and CPE Names allowing inheritance of tags to more specific CPE names
- Enhance the CPE Language to query tags
  - Allow querying tags to determine a set of CPEs in addition to standard CPE Name matching
- Existing CPE Name components are essentially tags
  - Normalize version – (e.g. major, minor and patch level tags)

CPE can be used to report on non-computing assets

- Security policies exist for non-computing products that are produced by vendors
- Non-computing devices can be considered another type of asset for assessment and reporting

For example:

- Safes
- Door locks

The CPE Name components can also be used with non-computing products

CPE Name: cpe:/h:sentrysafe:d880

Title: SentrySafe Security Safe D880

CPE Name: cpe:/h:simplex:sim1011

Title: Simplex Pushbutton Lock, Knob without Bypass, Dull Chrome

Important CPE Information

Website: http://cpe.mitre.org

CPE Dictionary Website: http://nvd.nist.gov/cpe.cfm

Discussion List: cpe-discussion-list@lists.mitre.org

Presenter:

David Waltermire

david.waltermire@nist.gov