VESPER – a tool for managing Vulnerabilities and Exploits in Software with Portscan-Endorsed Results

26. DFN-Konferenz „Sicherheit in vernetzten Systemen“
06.02.2019, Hamburg
Tanja Hanauer, Wolfgang Hommel, Christoph Wüstner
Agenda

- Motivation
- State of the Art
- VESPER
- Proof of Concept: VESPER in a HEI Data Center
- Conclusion
The Boy, who cried Wolf

The Boy Who Cried Wolf

There once was a shepherd boy who was bored as he sat on the hillside watching the village sheep. To amuse himself he took a deep breath and sang out, "Wolf! Wolf! The Wolf is chasing the sheep!"

The villagers came running up the hill to help the boy drive the wolf away. But when they arrived at the top of the hill, they found no wolf. The boy laughed at the sight of their angry faces.

"Don't cry 'wolf', shepherd boy," said the villagers, "when there's no wolf!"
They went grumbling back down the hill.

Later, the boy sang out again, "Wolf! Wolf! The wolf is chasing the sheep!" To his naughty delight, he watched the villagers run up the hill to help him drive the wolf away.

When the villagers saw no wolf they sternly said, "Save your frightened song for when there is really something wrong! Don't cry 'wolf' when there is NO wolf!"

But the boy just grinned and watched them go grumbling down the hill once more.

Later, he saw a REAL wolf prowling about his flock. Alarmed, he leaped to his feet and sang out as loudly as he could, "Wolf! Wolf!"

But the villagers thought he was trying to fool them again, and so they didn't come.
At sunset, everyone wondered why the shepherd boy hadn't returned to the village with their sheep. They went up the hill to find the boy. They found him weeping.

"There really was a wolf here! The flock has scattered! I cried out, "Wolf!" Why didn't you come?"

An old man tried to comfort the boy as they walked back to the village.

"We'll help you look for the lost sheep in the morning," he said, putting his arm around the youth, "Nobody believes a liar...even when he is telling the truth!"

VESPER is our approach to report only confirmed true positives by correlating identified potential vulnerabilities (CVEs) based on selected internal and external data sources.
Motivation

Vulnerabilities

Source: https://commons.wikimedia.org/wiki/Category:Security_vulnerability_logos
Motivation

- False positive results -> *Boy-who-cried-wolf*-scenarios
- Too many vulnerabilities, hard to weigh and overlook
- Lack of automated updates
Vulnerability:
A vulnerability is a weakness or the lack of a countermeasure.

Vulnerability Management:
The systematic management of vulnerabilities with the purpose of preventing attackers from exploiting known, and eventually even yet unknown, vulnerabilities. It is typically implemented as a process.

Only publicly known vulnerabilities are of interest.

Core activities are identifying, classifying, and either remediating or at least mitigating vulnerabilities. It can only be performed efficiently when at least parts of the process are automated through the use of tools.
Common Vulnerabilities and Exposures (CVE)

Example:
CVE-2017-5715
CVE-2017-5753
CVE-2017-5754

Industry standard Uniform naming convention for security vulnerabilities

Common Vulnerability Scoring System (CVSS)

Example:
CVE-2017-5754

Comparable assessment of security vulnerabilities

Tools should work:
- Out-of-the-box
- Fully automated
- Identify all affected machines (FNR=0)

Manual work performed by vulnerability tool vendors

State of the Art
Existing Approaches

Price?
A high false positive rate
Acquire information about vulnerabilities

Map vulnerabilities to assets

Support targeted action

"A picture is worth a thousand words"

Well established visualization guidelines:

- Principles of Gestalt Theory
- Tufte’s Design Principles
- Shneiderman’s Information Seeking Mantra
- Data-driven approach
- Data visualization guidelines to choose a kind of visualization and fine-tune it
“Simply dismissing a vulnerability because that port isn’t open to the internet is not enough.”

- Focus on TP -> overlook (some) vulnerable systems.
- No overall state of the vulnerability or exploitability.
- Only disclosed vulnerabilities.
- PoC: Small number of OS’, products, and scope.
Workflow

VESPER

Organizational Data
Port Scans
Update Scans
NVD CVE Data
Mapping Table
Security Advisories

Find Relevant Servers
Reachable Services

Find Potential CVE Matches
Installed Packages Associated with Vulnerable Service

Verify Results X-Checker
Vulnerable Packages on Server
PoC: Data Sources

- **Environment**
- **Data Sources**
  - Organizational Data
    - CMDB
    - Inventory DB
    - LDAP
  - Scan Data
    - Port Scans
    - Update Scans
  - Vulnerability Data
    - Common Vulnerabilities and Exposures
    - Nessus
    - OpenVAS
PoC: Data Quality Dimensions

- **Completeness**: Proportion of stored data against the potential of 100% complete.

- **Uniqueness**: No thing will be recorded more than once based upon how that thing is identified.

- **Timeliness**: The degree to which data represent reality from the required point in time.

- **Validity**: The data conforms to the syntax (format, type range) of its definition.

- **Accuracy**: The degree to which data correctly describes the „real world“ object or event being described.

- **Consistency**: The absence of difference, when comparing two or more representations of a thing against a definition.
PoC Implementation: X-Checker
### PoC Implementation: Mapping Table

<table>
<thead>
<tr>
<th>OS</th>
<th>Package</th>
<th>Vendor</th>
<th>Product</th>
<th>NmapService</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debian, SLES</td>
<td>apache2</td>
<td>apache</td>
<td>http_server</td>
<td>http</td>
</tr>
<tr>
<td>Debian, SLES</td>
<td>nginx-common</td>
<td>nginx</td>
<td>nginx</td>
<td>http</td>
</tr>
<tr>
<td>Debian, SLES</td>
<td>samba</td>
<td>samba</td>
<td>samba</td>
<td>http</td>
</tr>
<tr>
<td>SLES</td>
<td>vsftpd</td>
<td>beasts</td>
<td>vsftpd</td>
<td>ftp</td>
</tr>
</tbody>
</table>

Nessus version 7.2.0 taking two hours and reporting 197 servers.

Vulnerability Scan Data (OpenVAS): are the result of a full and deep scan in safe mode using OpenVAS version 9 taking 16 hours and reporting exactly the same servers as Nessus.

Data quality in the Proof of Concept is crucial as only valid, complete, timely, and consistent entries in the various data sources can produce correct results. The data quality is ensured when the matching in VESPER utilizes the port scan results to find reachable services. When a reachable service is detected, the operating system and the hostname from the organizational data is correlated with it and the list of the installed software from the update scans to find the installed package.

A manually created and curated ‘Mapping Table’ shown exemplarily in Table 1 maps CVE product information to the name used in the Linux distribution and the service identified by the port scans:

This is necessary because the CPE’s vendor and product strings do not necessarily correspond to the installed package name, for example CPE’s vendor:product is `apache:http_server` for the installed package `apache2`.

The product assumed by the port scan cannot be used without further verification as it is often incorrect and the product provided by the CVE is often not correct either, or not mappable to the software package, or its name onto the server, as observed by [SU17].

Nessus is a proprietary, widely used vulnerability scanner developed by Tenable Network Security [https://www.tenable.com/products/nessus/nessus-professional].

OpenVAS is an open source vulnerability scanner and manager, including network vulnerability tests, developed by Greenbone [http://www.openvas.org/index.html].
PoC: Results and Usability
Evaluation: VESPER, Nessus and OpenVAS

181 externally reachable IPs
OpenVAS > 7,100 results
=> Example Package Apache

Nessus: 89 CVEs (42 servers); 9 upgrade
OpenVAS: 887 CVEs, QoD, no category

Manual analysis
Nessus: 1 server with 2 CVEs
OpenVAS: 14 servers with 14 CVEs
VESPER: 14 servers with 63 CVEs
7 servers + 55 CVEs only found by VESPER
Conclusion

Conclusion I

- Manual setup of a data mapping table reflecting how software packages are named by the operating systems used locally.

- Focus on true positives requires adequate information about one’s own IT infrastructure.

- PoC showed the positive aspects of approach.

- The reported vulnerabilities were all verified and led to actionable tasks tracked easily.
TP results initiate a vulnerability management process.

No false alarms work well to motivate stakeholders with limited resources to look at vulnerability reports (again).

Highlight highly exposed services to be mitigated.
VESPER is consciously not a tool to use out-of-the-box. It requires initial and ongoing maintenance efforts, but then delivers high-quality results. Preparations can be done for example by security team members.

Quickstart Demo (python3 + pip required):

- git clone https://gitlab.lrz.de/vesper/vesper.git
- vesper/setup.sh
- vesper/vesper.sh