Trustworthiness of Cyber Infrastructure for e-Science

Research Project

Berry Hoekstra & Niels Monen

{bhoekstra,nmonen}@os3.nl

June 30, 2010

Outline

Problem domain

- Some applications need assurance.
  - Privacy sensitive data.
  - Protection of resources.
  - Protection of integrity.

Research question

Research question:

> *Is it possible to reason about how secure a system is?*
> *Is it feasible to determine the robustness of a system to withstand attacks?*

Research context

- Grid systems.
- Redhat- and Debian-based OS.
- Use vulnerability database(s).
  - OSVDB
  - NVD

Approach

- Determine the state of a system.
- Generate a host description.
  - List of OS, package and binary versions.
- Compare versions against a vulnerability database.
- List and analyse results.
  - Export and show information for others to evaluate this system's security.

National Vulnerability Database

- We chose NVD.
  - Use of standards.
  - Vulnerability scores included (CVSS).
  - Aggregated databases.
  - Machine-readable.
  - XML data feed.

Proof of Concept (1)

- Generate a version list of local binaries and packages (host description).
- Get a list of known vulnerabilities from the NVD.
- Match both lists.
- Generate results.
- Limitations.
  - Libraries: only lib packages are checked.

Proof of Concept (2)

- Generate a host description file.

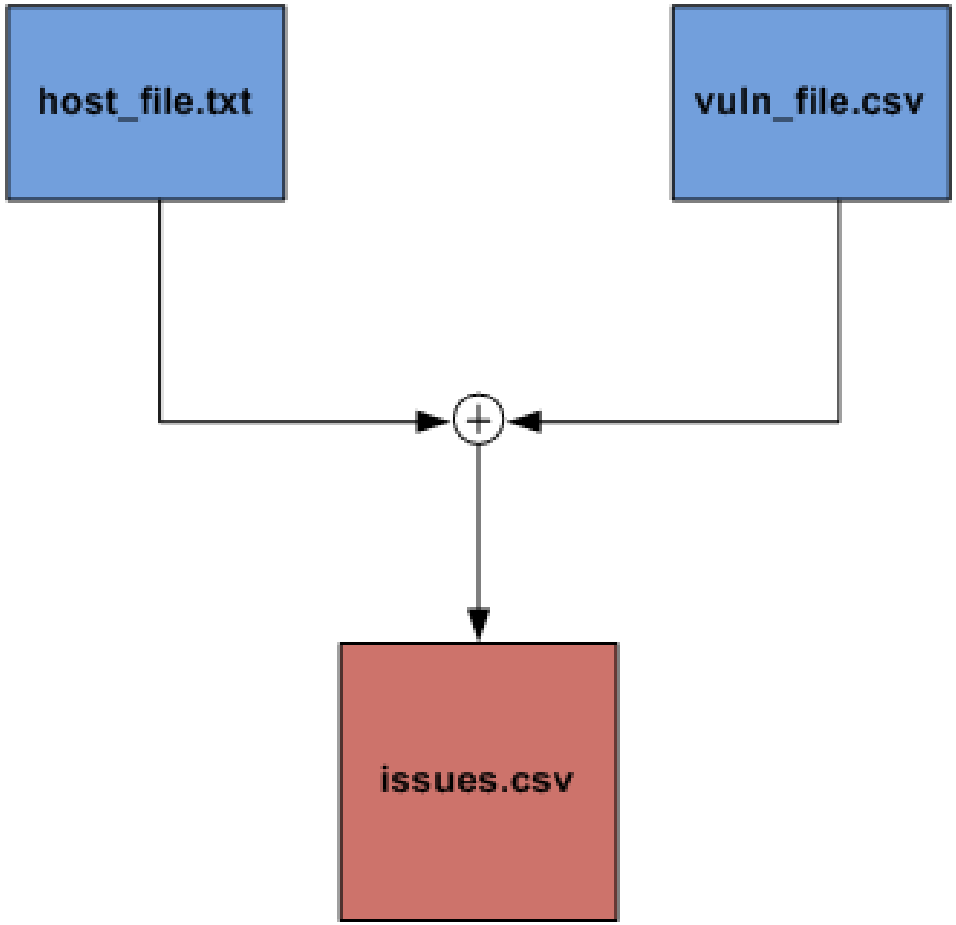
```
uname -r >> hostdescriptionlist.txt
for program in 'ls /usr/bin/'
do
$version = '$program --version'
echo $program $version >> hostdescriptionlist.txt
done
```

- Issues
  - Some binaries don't output version info.
  - Version format not consistent.
  - Binary name != package name.

Proof of Concept (3)

- Get vulnerability information.
  - Get latest NVD XML snapshot.
  - Canonicalize snapshot to CSV format.
  - Only extract necessary information.
    - Vulnerable binaries with version number.
    - CVE ID.
    - Publishing date.
    - CVSS score.
    - Access vector.
    - Privilege escalation.
    - Vulnerability summary (human-readable).
- End up with vulnerability file.

Proof of Concept (4)



Proof of Concept (5)

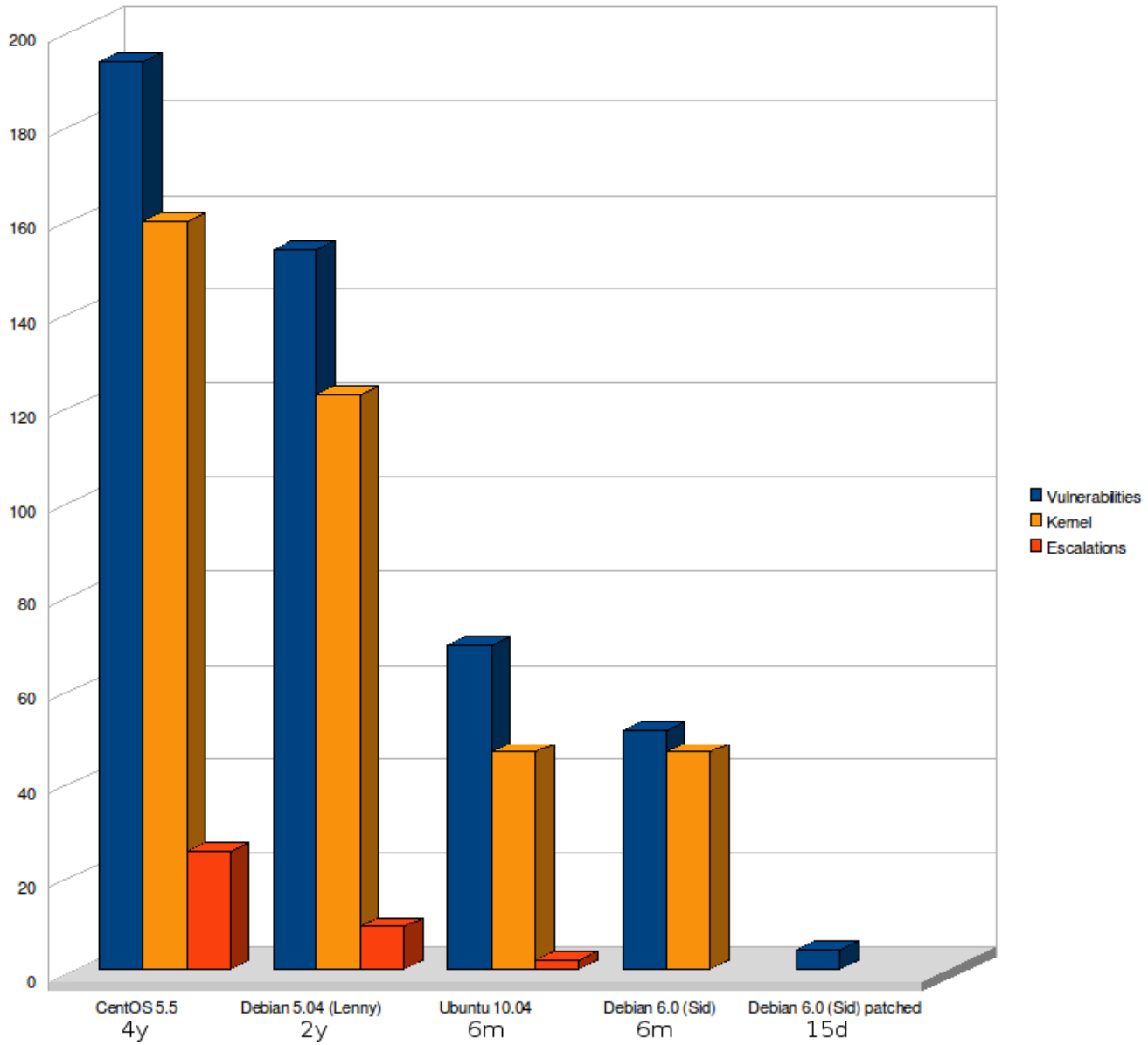
- Generated output.

```
nano,2.2.2,CVE-2010-1160,2010-04-016T20:30:01.397-04:00,1.9,
LOCAL,NONE,GNU nano before 2.2.4 does not verify whether a
file has been changed before it is overwritten in a file-
save operation, which allows local user-assisted attackers
to overwrite arbitrary files via a symlink attack on an
attacker-owned file that is being edited by the victim.
```

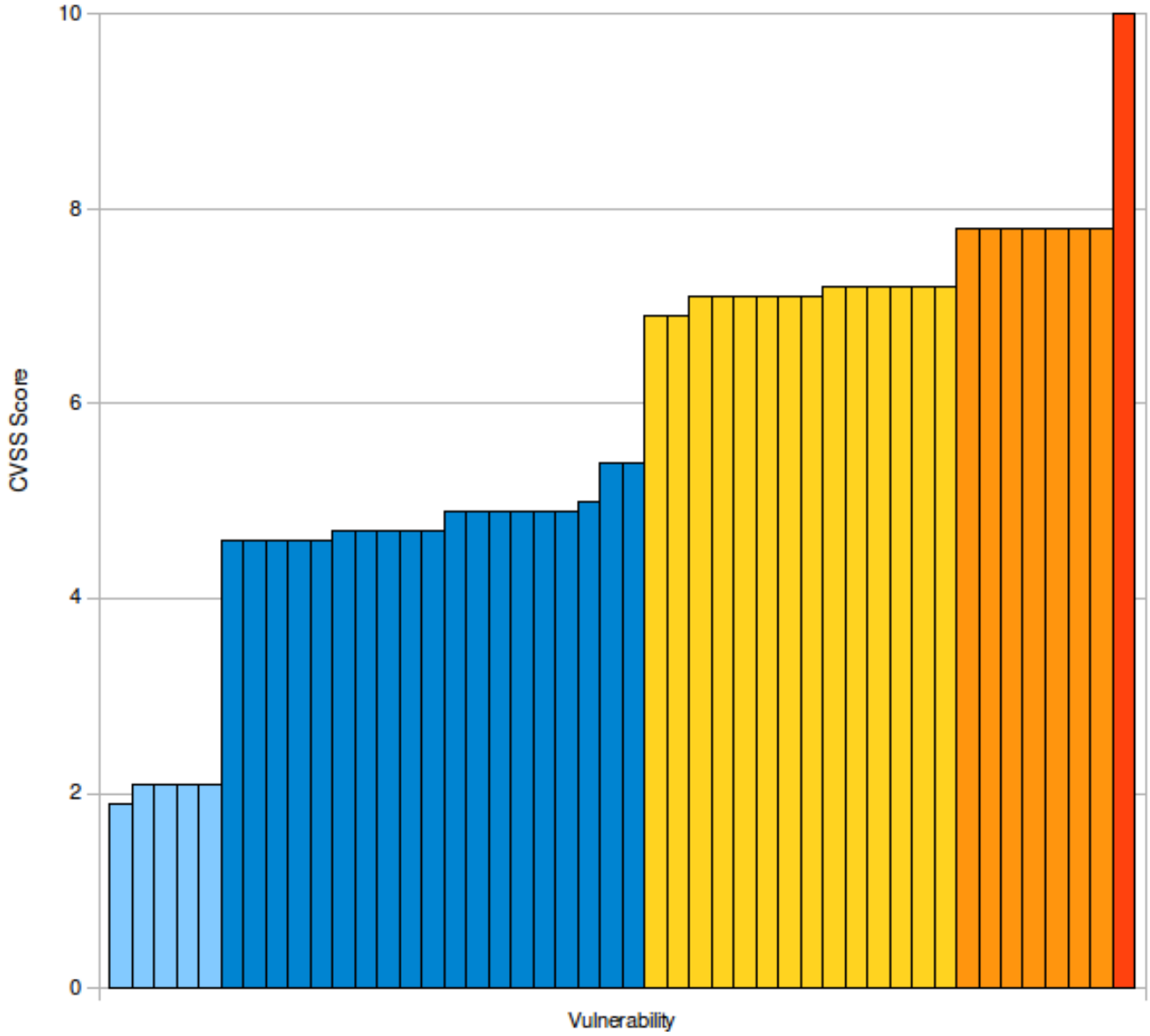
Generated Results

- Analysis of the output.
- Fully updated Debian 5.04 (Lenny) (2 years old)
  - 122 vulnerabilities for kernel 2.6.26.
  - 31 vulnerabilities for packages.
  - 9 privilege escalations.
- Fully patched Debian 6.0 (Squeeze/Sid) (6 months old)
  - 46 vulnerabilities for kernel 2.6.32.
  - 5 vulnerabilities for packages.
  - 0 privilege escalations.
- Kernel issues have CVSS score from 1.9 to 10.0 (max).

System statistics (1)



System statistics (2)



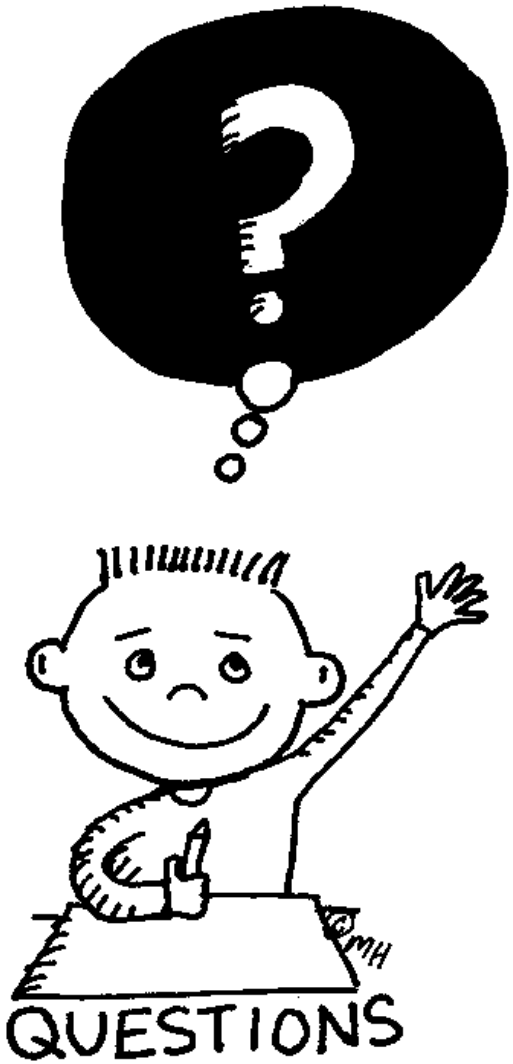
Conclusion

- Proof of concept shows that:
  - Possible to quickly generate a host description file.
  - Even bleeding edge OS can be vulnerable.
  - Found vulnerabilities are mostly patched in next version.
  - Drivers and file system vulnerabilities are most present.

Future work

- Is CVSS scoring useful?
  - Driver bias.
  - Local vs. Network bias.
  - Not usable for batch systems.
- Grid vulnerability scoring system.

Questions?



Source: discoveryeducation.com