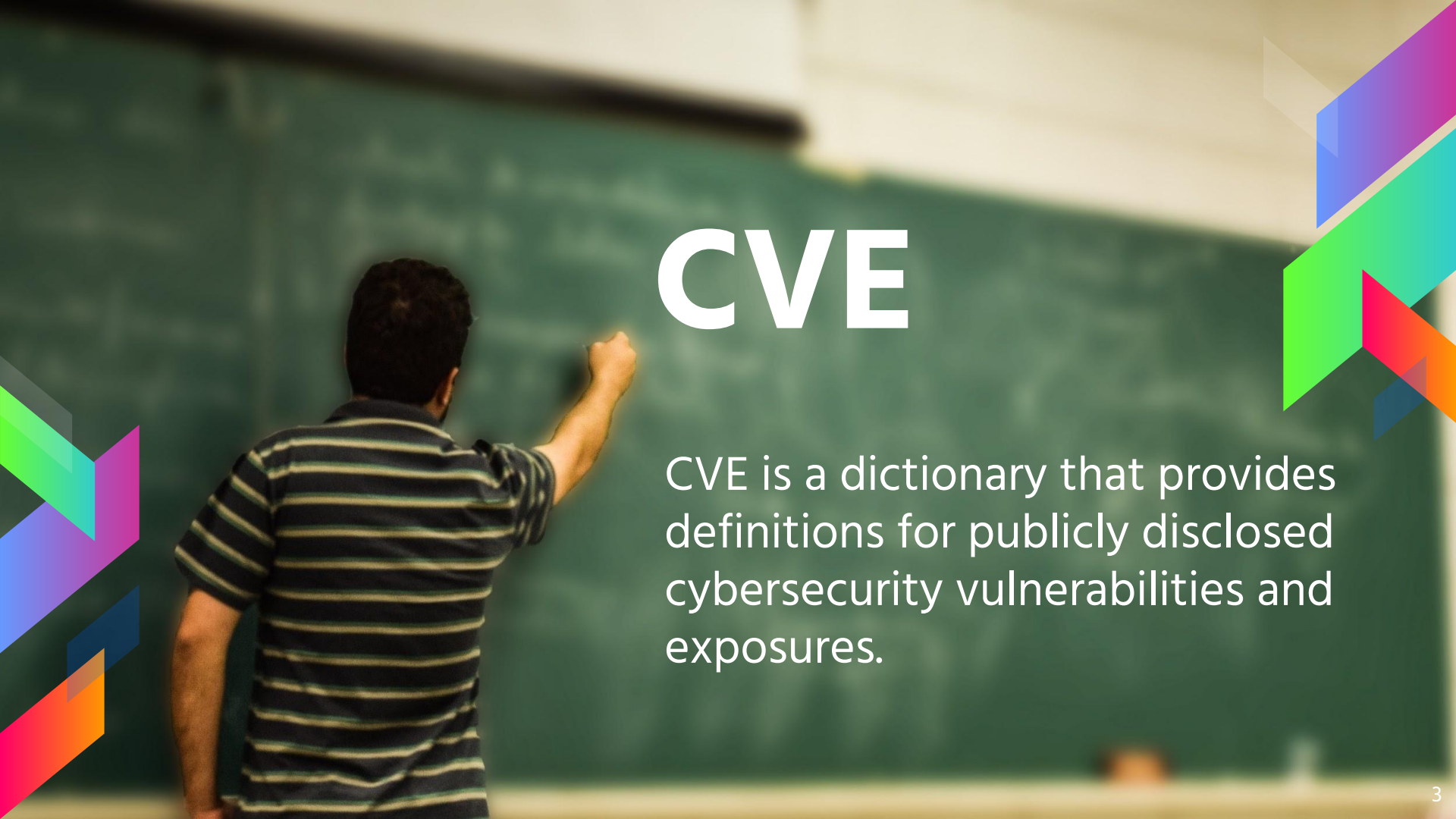
DISTRIBUTION OF VULNERABILITIES IN CVE

Meet our team

Mauro Borrazzo

Giulia Sellitto

Luigi D’Arco



CVE

CVE is a dictionary that provides definitions for publicly disclosed cybersecurity vulnerabilities and exposures.

1999

Launched when most cybersecurity tools used their own databases with their own names for security vulnerabilities.

At that time there was significant variation among products and no easy way to determine when the different databases were referring to the same problem.

NOW

CVE is now the industry standard for vulnerability and exposure identifiers.

The MITRE Corporation currently maintains CVE, a nonprofit that operates research and development centers sponsored by the federal government

The goal of CVE

CVE aims to standardize the names for all publicly known vulnerabilities and security exposures. The goal of CVE is to make it easier to share data across separate vulnerable databases and security tools.

How CVE Works

The process of creating a CVE Entry begins with the discovery of a potential security vulnerability..

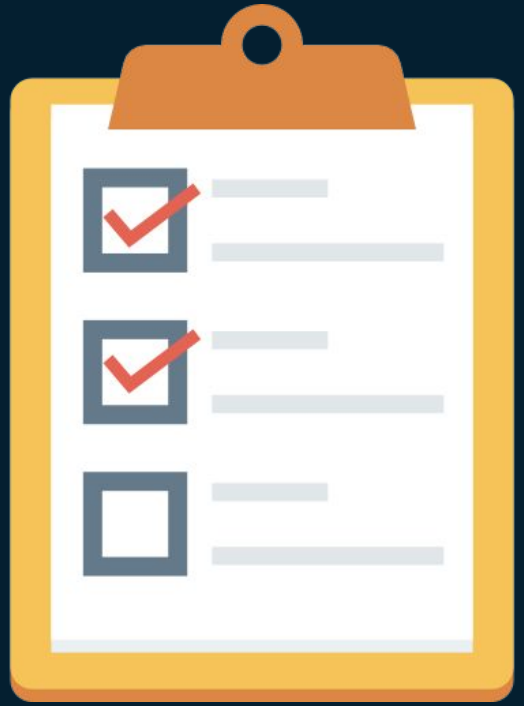
the CNA writes the Description and adds References, and then the completed CVE Entry is added to the CVE List and posted on the CVE website by the CVE Team.

The information is then assigned a CVE ID by a CVE Numbering Authority (CNA)

CVE isn't just another vulnerability database

It is designed to allow vulnerability databases and other capabilities to be linked together, and to facilitate the comparison of security tools and services

A CVE listing only contains the standard identifier number with status indicator, a brief description and references to related vulnerability reports and advisories



It does not include risk, impact, fix or detailed technical information

Can hackers use the CVE to break into networks?

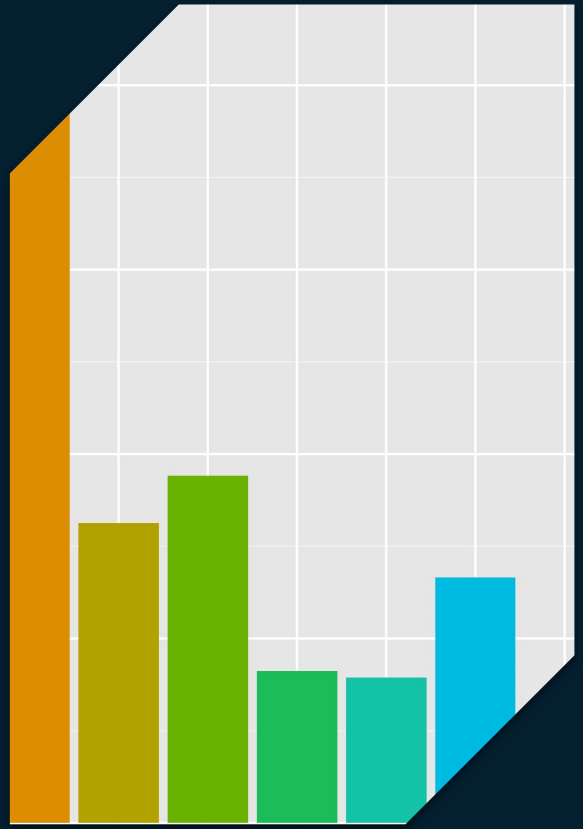
YES

benefits > risks

CVE lists only publicly known vulnerabilities and exposures, which means skilled hackers likely know about them anyway.

It takes much more work for an organization to protect its networks and fix all possible holes than it takes for a hacker to find a single vulnerability, exploit it, and compromise the network.

DATA SETS

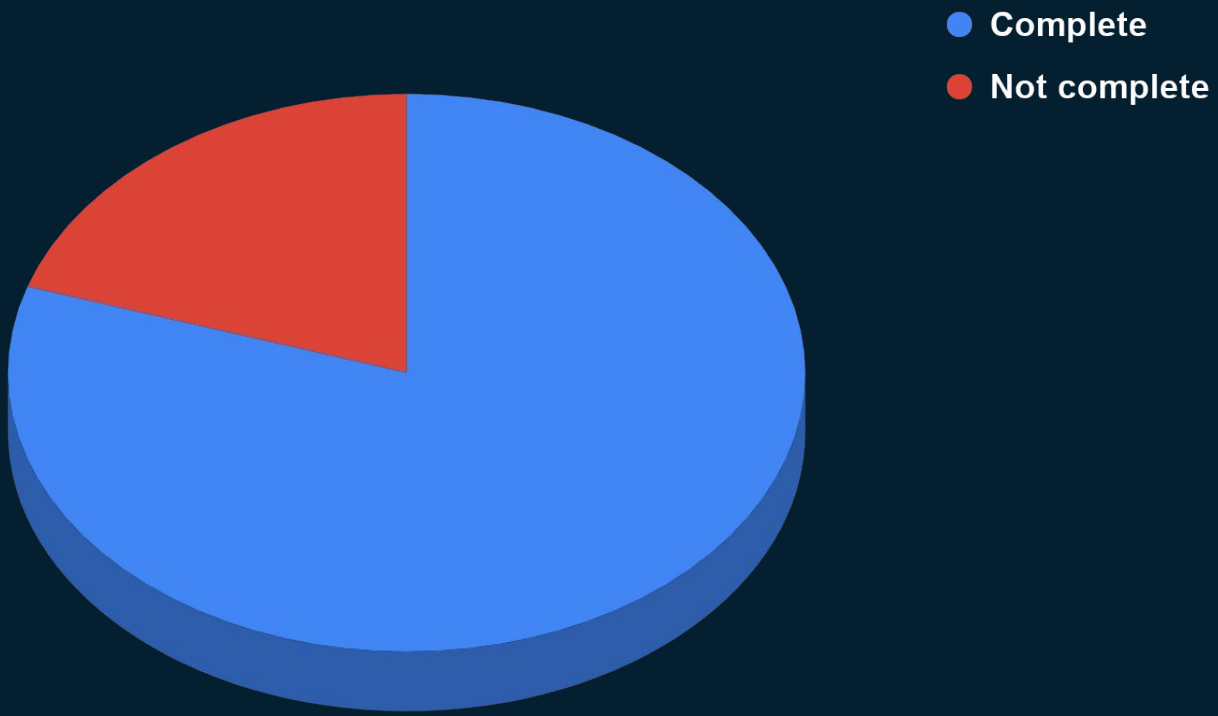


OVERALL

All CVEs that were first publicly reported in 2001 or later.

Includes all types of software, as long as the associated vulnerability has been reported by the owner.

CVE only includes distributable software, it does not include issues reported for custom softwares.



Estimation relative to all major mailing list and vulnerability databases

OS VENDOR

Operating system advisories

To capture kernels' vulnerabilities as well as applications that are supported by the OS vendor

Referring sources

Open source OS vendors

DEBIAN, FREEBSD, MANDRAKE/MANDRIVA, NETBSD, OPENBSD, REDHAT, SUSE

Closed source OS vendors

AIXAPAR, APPLE, CISCO, HP, MS, MSKB, SCO, SGI, SUN, SUNALERT

CVE does not have the internal data fields to support more finegrained analysis for major non-OS vendors

OPEN/CLOSED SOURCE

- Same methods and categories as described before
- Codebase overlapping with open source products
- Both open and closed sets had at least 1700 vulnerabilities

Disputed vulnerabilities in each data set

OVERALL TRENDS

Buffer Overflow

Overrunning a buffer's boundaries
and overwriting adjacent memory locations

- # 1 vulnerability reported
  for several years
- until 2005

WHAT HAPPENED?

The rise of web application vulnerabilities

- Cross-Site Scripting (XSS)
- SQL Injection
- Remote File Inclusion

WHY?

The rise of web app vulnerabilities: why?

It’s simple to perform basic data manipulations.

<script>alert('hello')</script>	XSS
’	SQL Inj

This makes it easier for beginning researchers to quickly test large amounts of software.

The rise of web app vulnerabilities: why?

There is a plethora of freely available web apps.

Much of these are written by inexperienced programmers in easy-to-learn languages.

This leads to easy-to-find vulnerabilities, which are a target for beginning researchers.

The rise of web app vulnerabilities: why?

Also web defacers started publishing their own research, due to the ease of finding vulnerabilities.

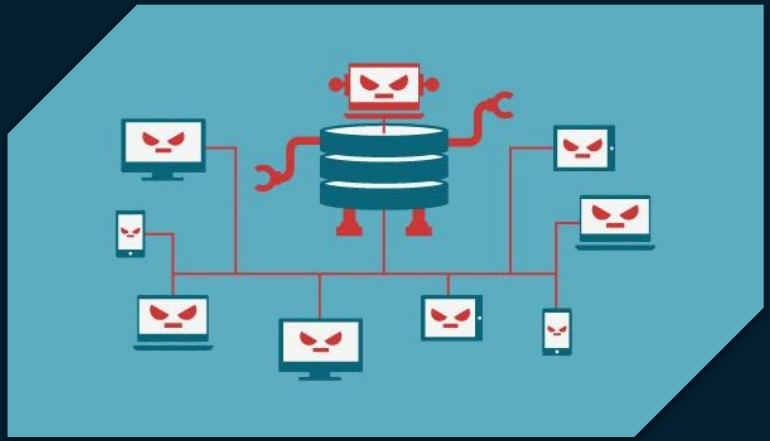
In particular, PHP file inclusion is regularly used to compromise web servers by installing powerful backdoor code.

Overall Trends

Other interesting results

Overall Trends: other interesting results

PHP Remote File Inclusion skyrocketed in 2006, probably due to the rise of botnet creations.



Overall Trends: other interesting results

For 2006, the top 5 vulnerability types are responsible for 57% of all CVEs.

- XSS
- SQL Injection
- PHP Remote File Inclusion
- Buffer Overflow
- Directory Trasversal

Overall Trends: other interesting results

Cross-Site Request Forgery remains a “sleeping giant”.

It is regularly found by researchers who focus on it, but the majority of researchers simply does not investigate this issue.

Overall Trends: other interesting results

Information leaks appear regularly.

Malformed input vulnerability is often reported, but the root cause is not usually investigated.

Both these vulnerabilities are general classes which encapsulate many types.

Table 1: Overall

	Total	2001	2002	2003	2004	2005	2006
Total	18809	1432	2138	1190	2546	4559	6944
XSS	2595 13.8%	31 02.2%	187 08.7%	89 07.5%	278 10.9%	728 16.0%	1282 18.5%
buf	2361 12.6%	279 19.5%	436 20.4%	268 22.05%	392 15.4%	445 09.8%	541 07.8%
sql-inject	1754 09.3%	6 00.4%	38 01.8%	36 03.0%	142 05.6%	588 12.9%	944 13.6%
php-include	1065 05.7%	1 00.1%	7 00.3%	12 01.0%	36 01.4%	96 02.1%	913 13.1%
dot	888 04.7%	127 08.9%	110 05.1%	34 02.9%	106 04.2%	196 04.3%	315 04.5%

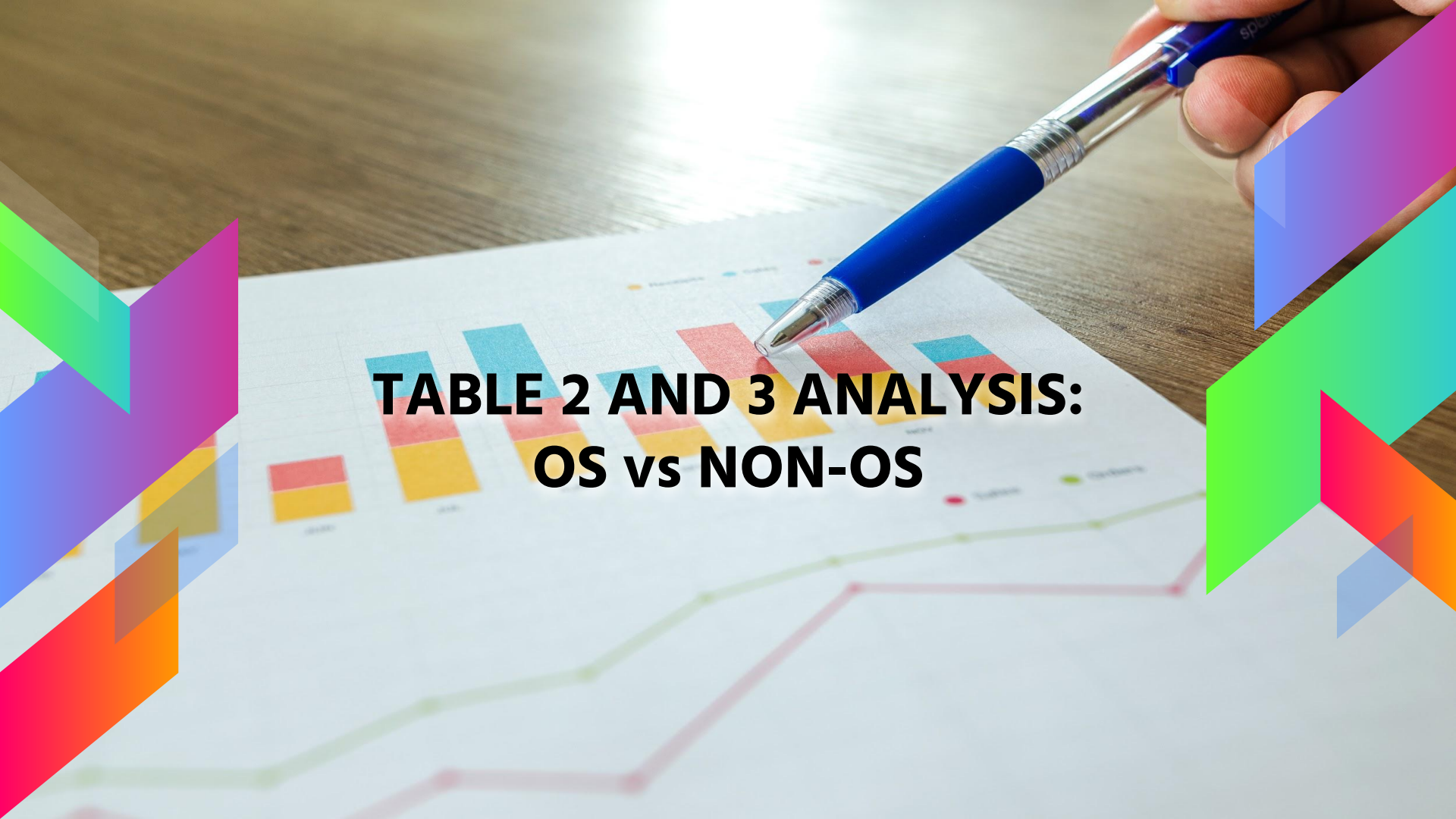


TABLE 2 AND 3 ANALYSIS: OS vs NON-OS

Web application vulnerability

Based solely on advisories from OS vendors

Table 2: OS Vendors

	Total	2001	2002	2003	2004	2005	2006
Total	4893	443	664	530	745	1216	1295
buf	958	93	178	131	152	195	209
link	186	33	22	22	38	51	20
dos-malform	182	25	41	14	33	22	47
XSS	168	7	29	16	11	51	54
int-overflow	140	0	8	12	34	25	61

buf	958	93	178	131	152	195	209

Buffer overflows are the #1.

This is probably due to underrepresentation of web applications in OS advisories, relative to other CVEs

Table 3: OS Vendors vs. Others

	Total	2001	2002	2003	2004	2005	2006
Total	4893 13916	443 989	664 1474	530 660	745 1801	1216 3343	1295 5649
XSS	168 2427	7 24	29 158	16 73	11 267	51 677	54 1228
buf	958 1403	93 186	178 258	131 137	152 240	195 250	209 332
sql-inject	48 1706	1 5	4 34	6 30	5 137	11 577	21 923
php-include	6 1059	0 1	0 7	0 12	0 36	4 92	2 911
dot	68 820	7 120	10 100	6 28	12 94	15 181	18 297

XSS	168 2427	7 24	29 158	16 73	11 267	51 677	54 1228

cross-site scripting (XSS) is the #1.

An informal analysis shows that the affected software includes web servers, web browsers, email clients...

SQL Injection

SQL injection vulnerabilities has had exponential growth with web application, because it seems that the OS-supported applications do not use database or aren’t web accessible.

PHP remote file inclusion

Similarly to the SQL injection, this has also had exponential growth with the advent of the web application

Table 4: Open and Closed Sources

	Total	2001	2002	2003	2004	2005	2006
Total	raw numbers omitted for open sources raw numbers omitted for closed sources						
buf	19.7% 19.6%	20.1% 20.2%	24.6% 27.6%	25.0% 25.7%	25.3% 15.0%	14.6% 18.5%	14.6% 18.5%
link	6.3% 1.4%	14.2% 1.0%	4.9% 1.8%	4.9% 3.0%	8.7% 1.9%	6.4% 0.8%	2.5% 1.2%
dos-malform	3.2% 4.8%	2.7% 9.1%	4.5% 8.1%	2.6% 2.5%	3.5% 7.1%	1.7% 2.1%	5.1% 3.1%
XSS	4.6% 2.3%	2.7% 0.5%	6.0% 3.6%	3.0% 2.5%	1.7% 0.8%	5.5% 2.1%	6.2% 3.1%
int-overflow	2.7% 1.9%	... ...	2.2% ...	3.4% 1.0%	4.1% 3.4%	2.2% 0.8%	3.4% 4.1%

Future Work

Future Work

- More precise classification
- Information about researchers
- Further market leaders

THANKS!

Any questions?