Quantitative Cyber-Security

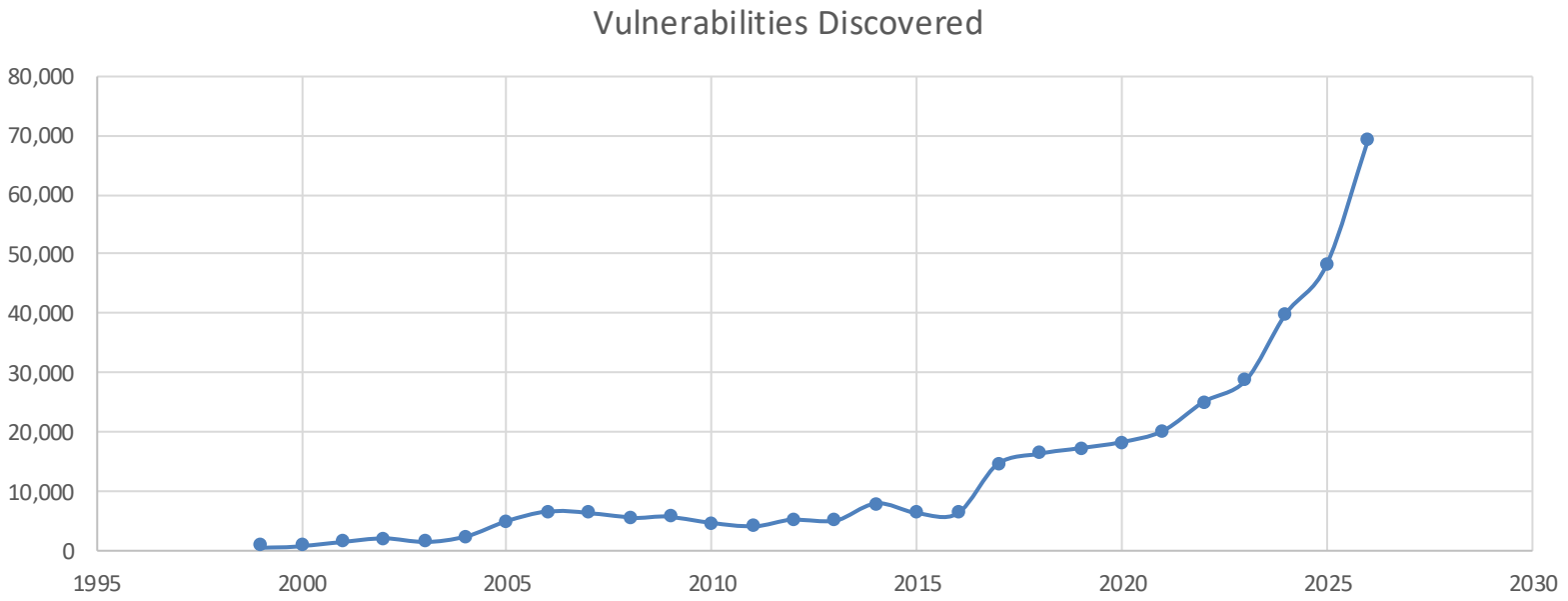
Colorado State University
Yashwant K Malaiya

CS559
Vulnerabilities



CSU Cybersecurity Center
Computer Science Dept

Vulnerability Discovery Trend



Vulnerability Data-bases

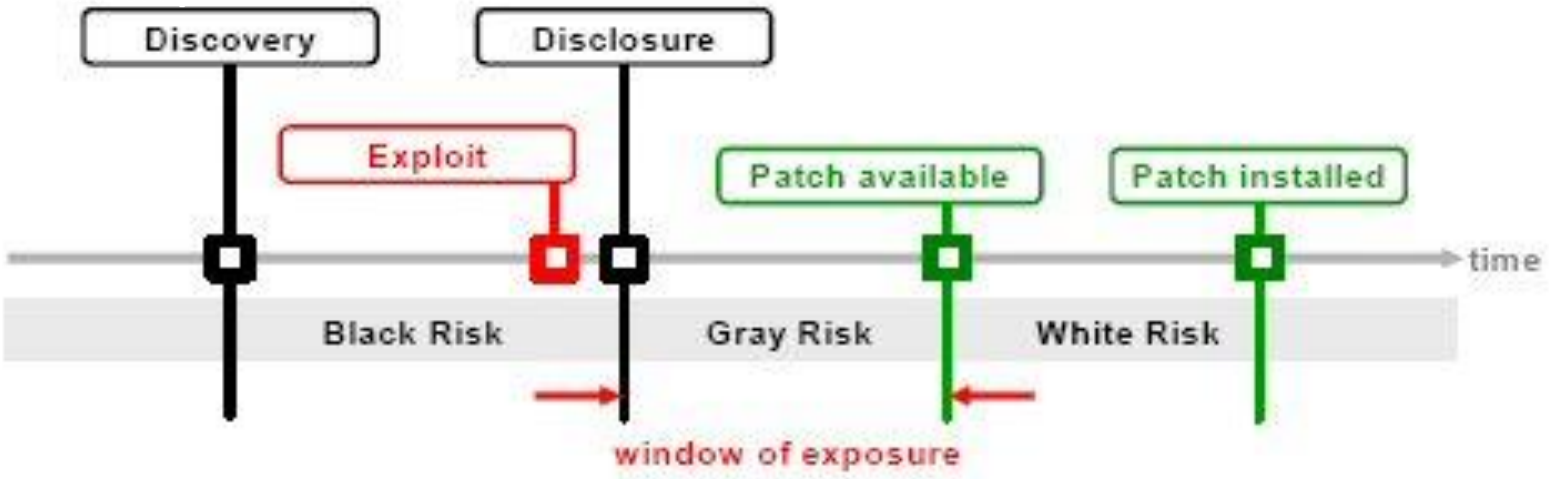
NVD database
CVEDetails
VulnDB
ExploitDB

2026: only partial data. Note the **jump in 2026**.
My theory: AI has made finding vulnerabilities easier

Why the jump in 2017?
More extensive collection by NVD?

Colorado State University

Vulnerability Lifecycle



- Add: creation of a vulnerability
- Exploit can be created before or after disclosure
- Many vulnerabilities are often inherited from a previous version.

Vulnerability density and defect density

- **Vulnerability densities**: 95/98: 0.003-0.004 NT/2000/XP: 0.01-0.02
- V_{KD}/D_{KD}: 0.75-2.31% about 1-5%

System	MSLOC	Known Defects (1000s)	D_{KD} (/Kloc)	Known Vulner - abilies	V_{KD} (/Kloc)	Ratio V_{KD} /D_{KD}
Win 95	15	5	0.33	45	0.0031	0.90%
NT 4.0	16	10	0.625	231	0.0144	2.31%
Win 98	18	10	0.556	109	0.0060	1.09%
Win2000	35	63	1.8	637	0.0182	1.01%
Win XP	40	600	15	1352	0.0338	0.23%*
Win 10	65	1000	15.38	7458	0.1147	0.75%

Updated in 2026 from Alhazmi, Malaiya, Ray, " Measuring, Analyzing and Predicting Security Vulnerabilities in Software Systems," Computers and Security, May 2007, P 219-228. Note many vulnerabilities are inherited from a previous version, and are discovered much later.

2011-2026: **Most Prolific Vulnerability Discoverers**

Based on historical CVE tracking metrics and platform leaderboards dbugs.

Researcher	Organization / Affiliation	Vulnerabilities Discovered (CVEs)	Country / Location
Abdi Pranata	Independent Researcher	456	Indonesia
John Zuccato	IBM X-Force Ethical Hacking Team	436	United States
Luigi Auriemma	ReSecurity / Independent	430	Malta / Italy
Muhammad Yudha	Independent White-Hat	421	Indonesia
Chris Shepherd	IBM X-Force Ethical Hacking Team	420	United States
Rodney Ryan	IBM X-Force Ethical Hacking Team	416	United States
Rafael Pedrero	Independent	386	Spain
Tran Nguyen Bao Khanh	VNPT Cyber Immunity (VCI)	366	Vietnam
Stealthcopter	Independent	352	United Kingdom
Yuki Chen	Qihoo 360 Vulcan Team	348	China

Class Notes

Quizzes: You get two attempts for each quiz.

- Quizzes use different Canvas formats. The formula questions use randomized numbers, the results are computed automatically.
- Questions may be slightly different in the second attempt.
- Quiz with lowest scores will be dropped.

The Midterm will use the the Respondus lockdown browser,

  - for both on-campus and distance students.
  - You need to use a laptop (Windows or Mac) capable of running the Respondus lockdown browser.

Major Bug Bounty Platforms

- **HackerOne**: The largest general crowdsourced platform, nearly 2,000 active programs. Serving tech giants, financial institutions, and government agencies like the US Department of Defense.
- **Bugcrowd**: Second largest providing Managed Bug Bounties, Vulnerability Disclosure Programs (VDPs), and Pen-Testing-as-a-Service.
- **Immunefi**: The crowdsourced platform specifically for Web3, smart contracts, and decentralized finance (DeFi).
- **In-House Corporate Programs**: Google VRP, Microsoft, Apple, Meta, etc—often bypass third-party platforms to run massive independent crowdsourced initiatives directly

Vulnerability Classification CVE Details

Year	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	File Inclusion	CSRF	XXE	SSRF	Open Redirect	Input Validation
2016	418	1096	85	476	90	4	85	39	15	28	0
2017	945	1340	485	1427	252	13	326	101	55	69	1
2018	2055	1720	503	2039	569	111	479	187	118	85	1204
2019	1194	1975	544	2387	485	125	558	136	103	121	878
2020	1212	1818	462	2196	434	107	414	119	130	99	786
2021	1641	2454	740	2713	545	86	520	126	187	133	645
2022	1749	2794	1760	3362	680	85	766	123	228	137	646
2023	1588	1964	2111	5090	736	105	1389	124	237	163	507
2024	1719	2353	2647	7431	911	244	1428	110	372	112	86
2025	2249	2894	3946	8728	1053	672	1950	115	558	166	0
2026	3715	4423	3210	5829	1800	624	1037	120	1424	219	0
Total	18485	24831	16493	41678	7555	2176	8952	1300	3427	1332	4753

https://www.cvedetails.com/vulnerabilities-by-types.php
2016-2026
Note: A vulnerability can have multiple types.

Classification by impact

Year	Code Execution	Bypass	Privilege Escalation	Denial of Service	Information Leak
2016	1239	1	149	2050	102
2017	1870	340	538	3372	470
2018	1728	640	824	2207	1390
2019	1556	651	895	1697	1313
2020	1695	789	1360	1677	1083
2021	2087	764	1077	2297	906
2022	2067	808	1391	2437	1101
2023	2580	809	1276	2560	1378
2024	3962	628	1052	2470	892
2025	3039	637	1019	2526	742
2026	4834	1850	1447	5324	1168
Total	26657	7917	11028	28617	10545

Vulnerabilities that lead to a denial of service condition published in 2023

)

CVE numbering system

CVE, (Common Vulnerabilities and Exposures), is a list of publicly disclosed computer security flaws.

- A CVE, usually means the CVE ID number assigned to a security flaw ("vulnerability").
- Security advisories issued almost always mention at least one CVE ID.
- CVE is overseen by the MITRE corporation
  - funded from the Cybersecurity and Infrastructure Security Agency, part of the U.S. Department of Homeland Security
- CVE identifiers are assigned by a CVE Numbering Authority (CNA).
  - There are about 100 CNAs, including major IT vendors as well as security companies and research organizations. MITRE can also issue CVEs directly.
  - CVE reports can come from a vendor, a researcher, or anyone who has discovered a flaw
  - There are mechanisms for responsible disclosure.
  - a CVE ID is assigned before a security advisory is made public. Vendors generally keep security flaws secret until a fix has been developed and tested.
  - Once made public, a CVE entry includes
    - the CVE ID (in the format "CVE-2019-1234567"),
    - a brief description of the security vulnerability or exposure, and
    - references, which can include links to vulnerability reports and advisories.

Reward Programs 2026

Payer / Platform Type	Low Severity Tier	High / Critical Severity Tier	Maximum Ceiling Reward	Key Comments & Specializations
Sherlock (Web3)	$500 – $1,000	$20,000 – $150,000	$16,000,000	Uses a stake-to-submit model. Recorded the largest single payout in tech history for the Usual protocol.
Immunefi (Web3)	$1,000 – $2,500	$50,000 – $1,000,000	$10,000,000	Dedicated exclusively to smart contracts and DeFi. Sky (formerly MakerDAO) maintains a current $10M max bounty.
Apple (In-House Tech)	$5,000 – $10,000	$100,000 – $500,000	$2,000,000+	Focuses heavily on mobile OS and firmware. Max reward scales past $5,000,000 with specific beta/Lockdown Mode bonuses.
Google (In-House Tech)	$500 – $1,500	$30,000 – $250,000	$1,500,000	Covers Android, Chrome, and Core Infrastructure. The $1.5M ceiling is reserved for full-chain compromise of the Pixel Titan M2 chip.
Microsoft (In-House Tech)	$500 – $2,000	$20,000 – $100,000	$250,000	Distributes more total cash than almost any individual program (over $20 million paid out across 15 bounty tracks).
HackerOne (Enterprise Platform)	$50 – $300	$2,000 – $20,000	$2,000,000	The overall market leader. While corporate program averages sit around $15,000 for critical bugs, elite programs (like) scale to $2M.
Bugcrowd (Enterprise Platform)	$50 – $250	$1,500 – $15,000	$100,000+	Focuses on managed testing, VDPs, and pen-testing. Offers highly structured reward metrics based on precise technical risk mapping.
Synack (Premium Platform)	$100 – $500	$5,000 – $50,000	$200,000	A gated, invite-only "Red Team" model. Features a higher baseline payout per bug due to rigorous researcher vetting.

11

Average Bug Bounty

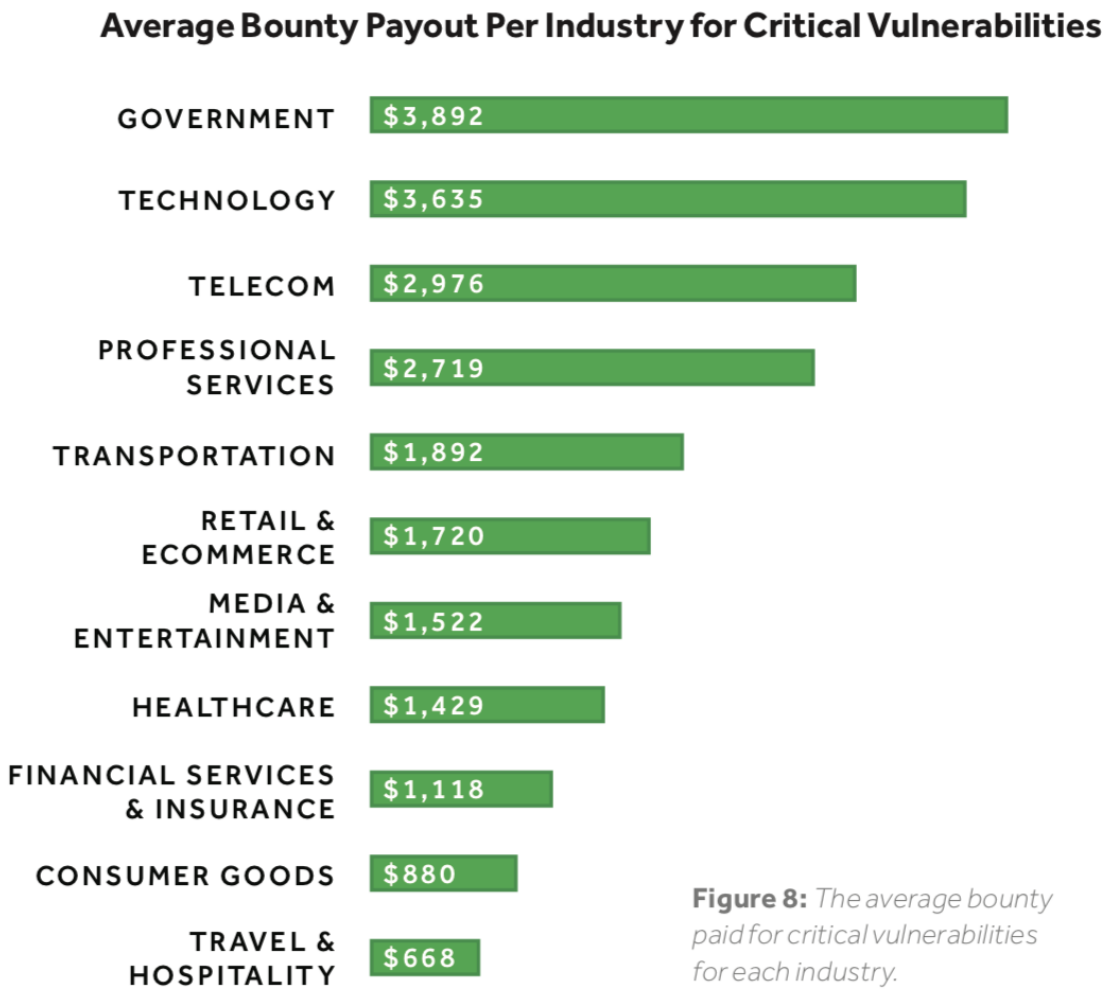


Figure 8: *The average bounty paid for critical vulnerabilities for each industry.*

https://threatpost.com/threatlist-bug-bounty-payouts-increase-six-percent-for-critical-vulnerabilities/133930/

Colorado State University

Vulnerabilities for sale

- Stefan Frei, NSS Labs looked at reports about some of those private vendors
  - Endgame Systems, Exodus Intelligence, Netragard, ReVuln and VUPEN
  - concluded that jointly these firms alone have the capacity to sell more than 100 zero-day exploits per year.

Provider	Offering	Remark / Source
Endgame Systems	25 exploits/year USD $2.5 million	*Business Week* http://www.businessweek.com/magazine/cyber-weapons-the-new-arms-race-07212011.html
Exodus Intelligence	60 exploits/year	Service Offering https://www.exodusintel.com/rsrc/ExodusIntelligence_EXP.pdf
ReVuln	> 9 exploits/year	Minimum estimate by counting exploits demonstrated here: http://vimeo.com/53806381 (2013-09-27)
VUPEN	> 7 exploits/year > 15 to 20 binary analysis and private 1-day exploits/month	Minimum estimate by counting list of published exploits here: http://www.vupen.com/blog/ (2013-09-27) Service Offering: http://www.vupen.com/english/services/ba-gov.php

Frei's minimum estimate of exploits offered by boutique exploit providers each year.

https://krebsonsecurity.com/2013/12/how-many-zero-days-hit-you-today/

Vulnerability Data Bases

- Vulnerabilities are security related defects.
- For ordinary defects found, the developers do not have an obligation to report them.
  - They just have to address them in the next patch/version.
- All recognized vulnerabilities are analyzed and reported in the data-bases.
- The new entries in the data-bases are used by antivirus developers/maintainers to add to their signatures.
- Used by researchers like us.

Major vulnerability databases

- NVD National Vulnerability Database https://nvd.nist.gov/
  - US 2005, includes vulnerability operation, its CVSS rating, and links to any available patches/fixes
- CVE Mitre DB https://cve.mitre.org/data/downloads/index.html
- CWE Common weakness enumeration https://cwe.mitre.org/
- VulnDB https://vulndb.cyberriskanalytics.com
  - Open source vulnerability database 2004. Commercia.
- WhiteSource Vuln DB https://www.whitesourcesoftware.com/vulnerability-database/
  - Includes not yet recognized vulnerabilities
- CVEDetails https://www.cvedetails.com better interface?
- Exploit DB https://www.exploit-db.com available exploits

Example: CVE-2018-8174

A remote code execution vulnerability exists in the way that the VBScript engine handles objects in memory

- NVD Details
- Microsoft acknowledgements
- CVE Details cvss2.0, products/versions affected, links
- Exploit-db has an exploit

HW2

- Will be assigned soon. Will be due Oct 15.
- You will extract vulnerability discovery data for two software products from a vulnerability data base (NVD).
- For each product, create a plot of the cumulative number of vulnerabilities starting from a date before the first vulnerability in it was detected.
- The next assignment 2b will use the data acquired from this assignment to analyze the vulnerability discovery process.

Quantitative Security

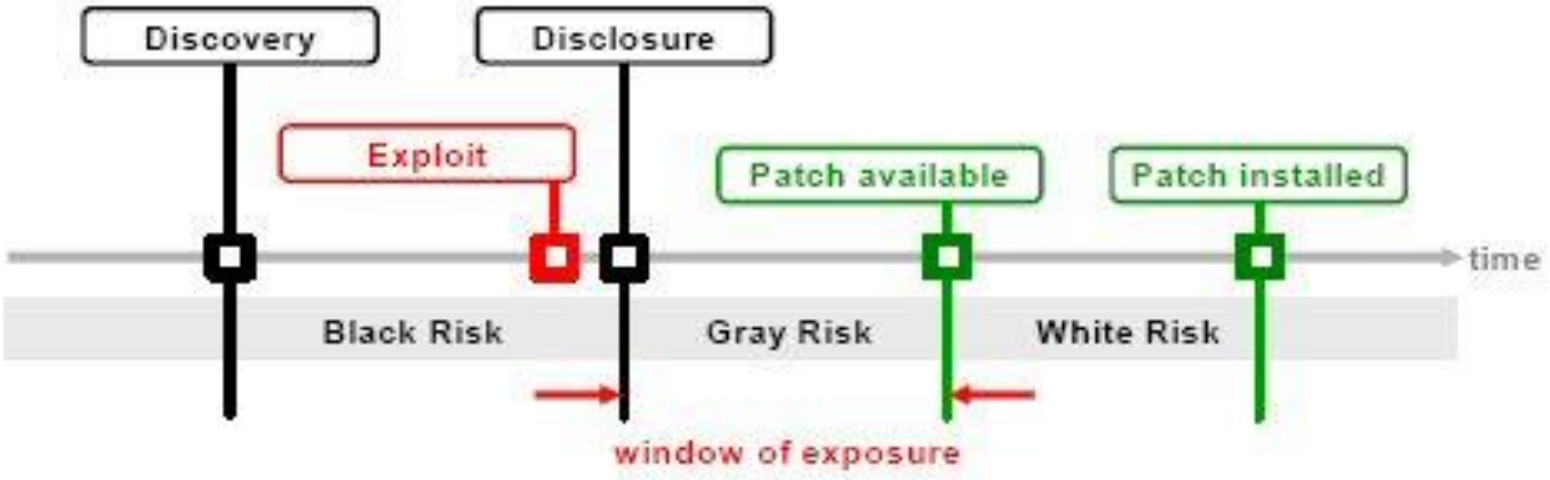
Colorado State University
Yashwant K Malaiya

CS 559
Vulnerability Life Cycle



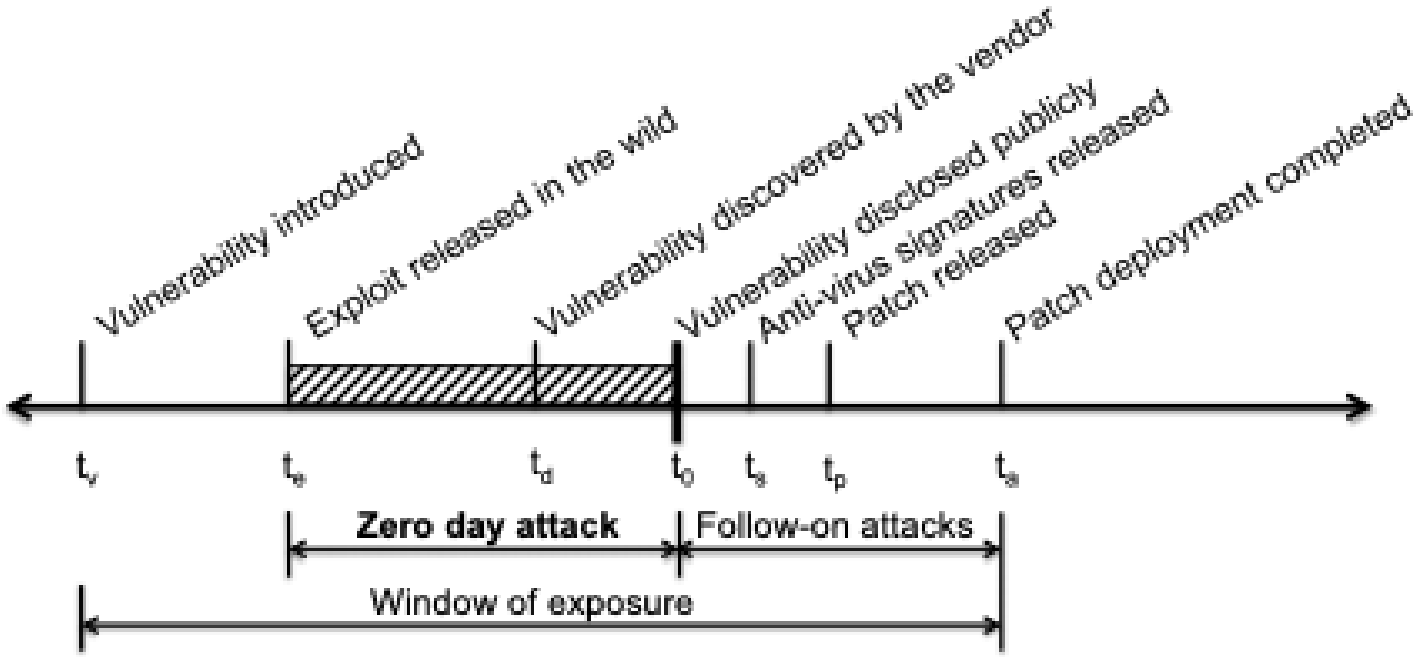
CSU Cybersecurity Center
Computer Science Dept

Vulnerability Lifecycle



19

Timeline



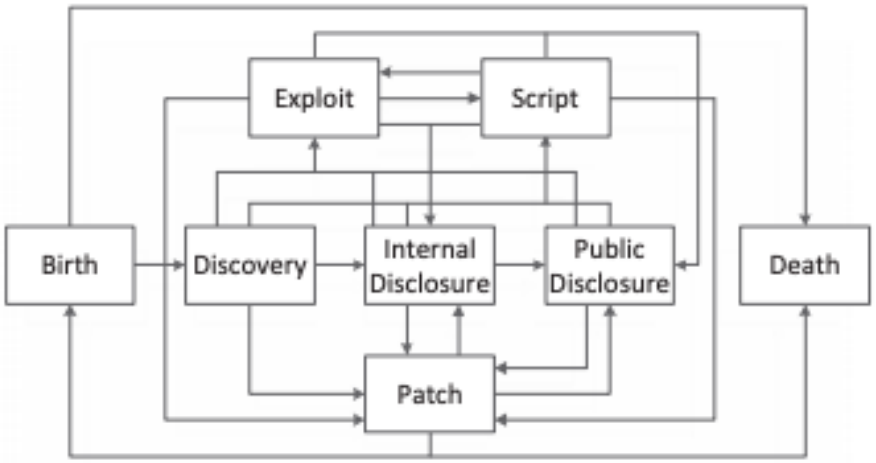
Attack timeline. These events do not always occur in this order, but ta > tp ≥ td > tv and t0 ≥ td.
The relation between td and te cannot be determined in most cases. For a zero-day attack t0 > te.

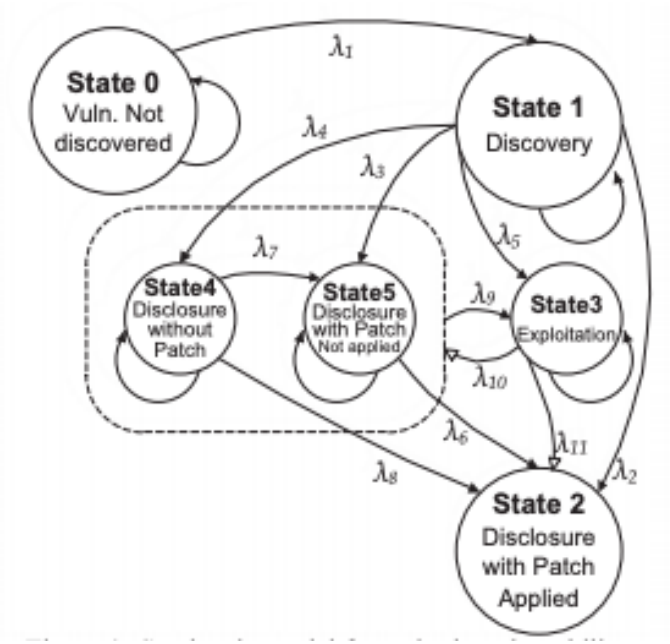
Before We Knew It An Empirical Study of Zero-Day Attacks In The Real World

Vulnerability Lifecycle

- **Vulnerability introduced**. A bug is introduced in software (time = t_v).
- **Exploit released in the wild.** Actors in the underground economy discover the vulnerability, create a working exploit and use it to conduct stealth attacks against selected targets (time = t_e)
- **Vulnerability discovered by the vendor.** The vendor learns about the vulnerability, assesses its severity, assigns a priority for fixing it and starts working **on a** patch (time = t_d).
- **Vulnerability disclosed publicly. The** vulnerability is disclosed, either by the vendor or on public forums and mailing lists. A CVE identifier (e.g., CVE-2010-2568) is assigned to the vulnerability (time = t_0).
- **Anti-virus signatures released.** Once the vulnerability is disclosed, anti-virus vendors release new signatures (time = t_s),
- **Patch released.** On the disclosure date, or shortly afterward the software vendor releases a patch for the vulnerability. After this point, the hosts that have applied the patch are no longer susceptible to the exploit (time = t_p)
- **Patch deployment completed.** All vulnerable hosts worldwide are patched and the vulnerability ceases to have an impact (time = t_a).

Stochastic Modeling



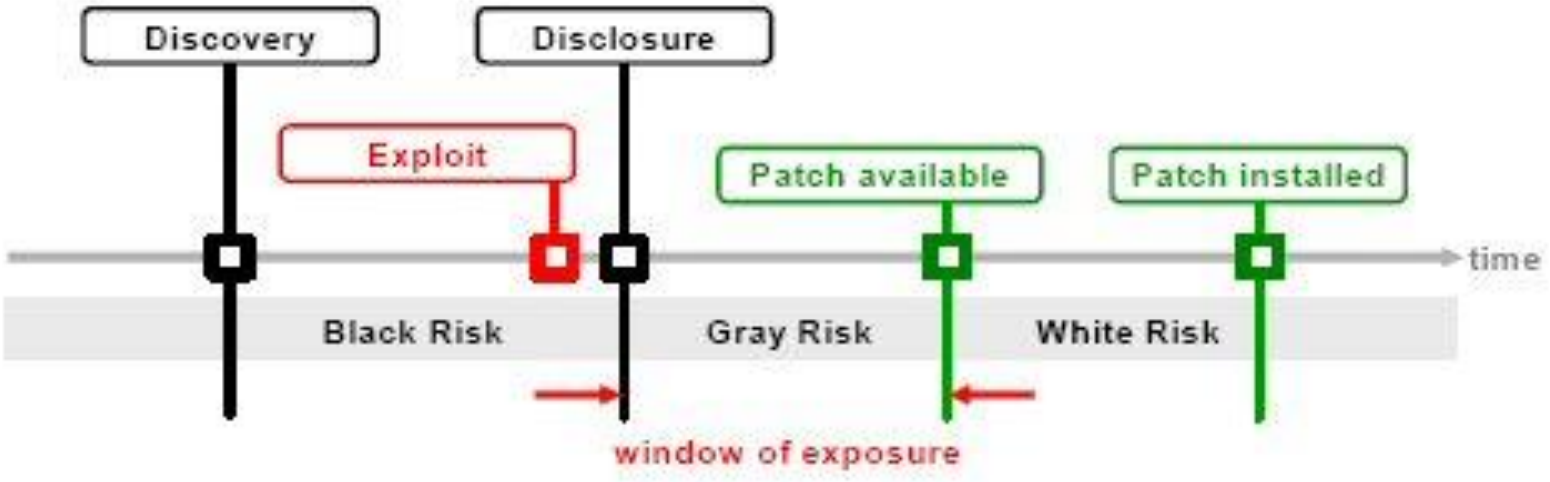


For a single vulnerability, the cumulative risk in a specific system at time t can be expressed as

- probability of the vulnerability being in State 3 at time t
- multiplied by
- the consequence of the vulnerability exploitation.

Joh and Malaiya, "A Framework for Software Security Risk Evaluation using the Vulnerability Lifecycle and CVSS Metrics" 2010

Zero-day attacks



- A **zero-day attack** is a cyber attack exploiting a vulnerability that has not been disclosed publicly.
- There is almost no defense against a zero-day attack:
  - while the vulnerability remains unknown, the software affected cannot be patched and
  - anti-virus products cannot detect the attack through signature-based scanning
- Notable zero-day attacks include (Bilge, Dumitras)
  - the 2010 Hydraq trojan, also known as the "Aurora" attack
  - the 2010 Stuxnet worm, which combined four zero-day vulnerabilities to target
  - industrial control systems and
  - the 2011 attack against RSA.

23

An Empirical Study of Zero-Day Attacks In The Real World

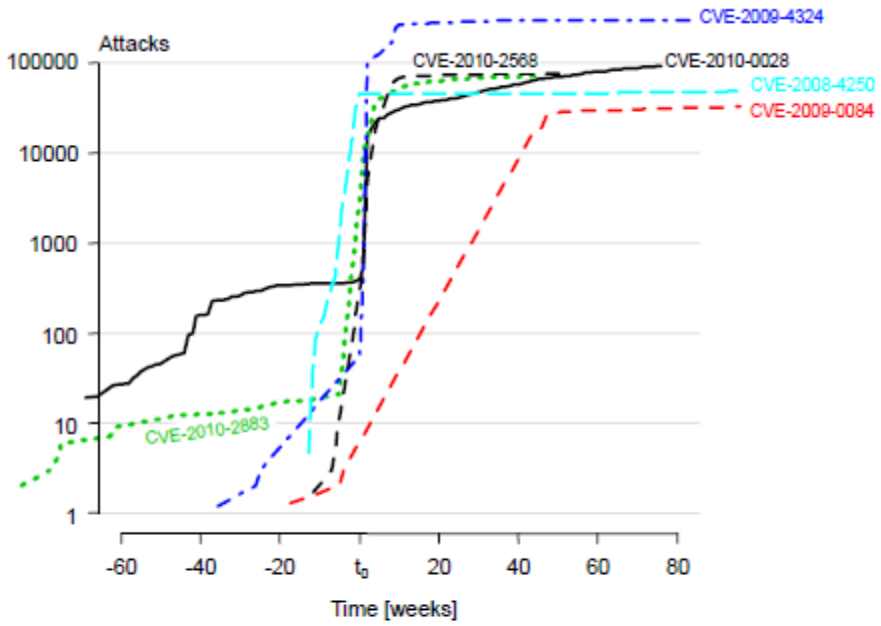
- L. Bilge and T. Dumitraş, CCS '12
- Field-gathered data for 11 million real hosts around the world.
- Searching this data set for malicious files that exploit known vulnerabilities indicates which files appeared on the Internet before the vulnerabilities were disclosed.
- They identify 18 vulnerabilities exploited before disclosure, of which 11 were not previously known to have been employed in zero-day attacks.
- They also find that a typical zero-day attack lasts 312 days on average
- After vulnerabilities are disclosed publicly, the volume of attacks exploiting them increases by up to 5 orders of magnitude.

- L. Bilge and T. Dumitraş. Before we knew it: an empirical study of zero-day attacks in the real world.
- Proc. 2012 ACM conference on Computer and communications security (CCS '12)., 833-844.

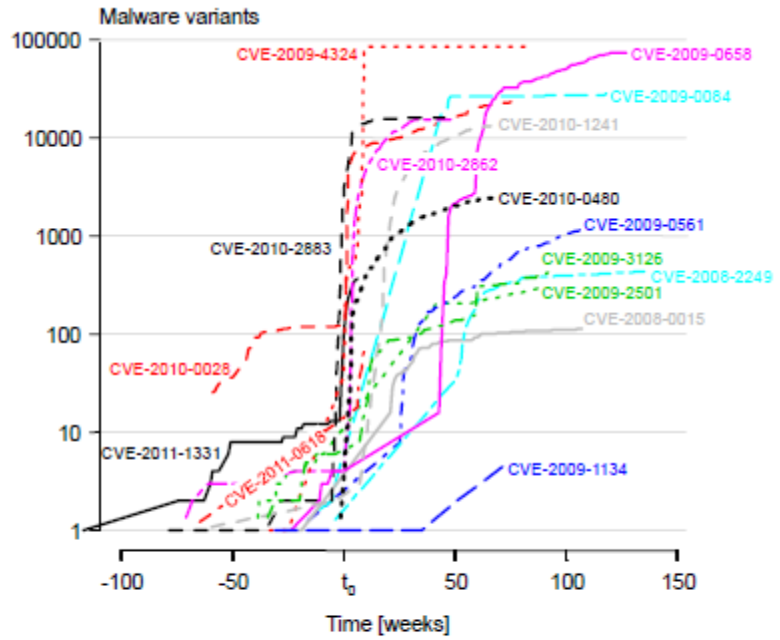
Summary of findings

Findings	Implications
Zero-day attacks are more frequent than previously thought: 11 out of 18 vulnerabilities identified were not known zero-day vulnerabilities.	Zero-day attacks are serious threats that may have a significant impact on the organizations affected.
Zero-day attacks last between 19 days and 30 months, with a median of 8 months and an average of approximately 10 months.	Zero-day attacks are not detected in a timely manner using current policies and technologies.
Most zero-day attacks affect few hosts, with the exception of a few high-profile attacks (e.g., Stuxnet).	Most zero-day vulnerabilities are employed in targeted attacks.
58% of the anti-virus signatures are still active at the time of writing.	Data covering 4 years is not sufficient for observing all the phases in the vulnerability lifecycle.
After zero-day vulnerabilities are disclosed, the number of malware variants exploiting them increases 183–85,000 times and the number of attacks increases 2–100,000 times.	The public disclosure of vulnerabilities is followed by an increase of up to five orders of magnitude in the volume of attacks.
Exploits for 42% of all vulnerabilities employed in host-based threats are detected in field data within 30 days after the disclosure date.	Cyber criminals watch closely the disclosure of new vulnerabilities, in order to start exploiting them.

Impact of discosure



(a) Attacks exploiting zero-day vulnerabilities before and after the disclosure (time = t_0).



(b) Malware variants exploiting zero-day vulnerabilities before and after disclosure (time = t_0).

Figure 6: Impact of vulnerability disclosures on the volume of attacks. We utilize logarithmic scales to illustrate an increase of several orders of magnitude after disclosure.

Time to exploit

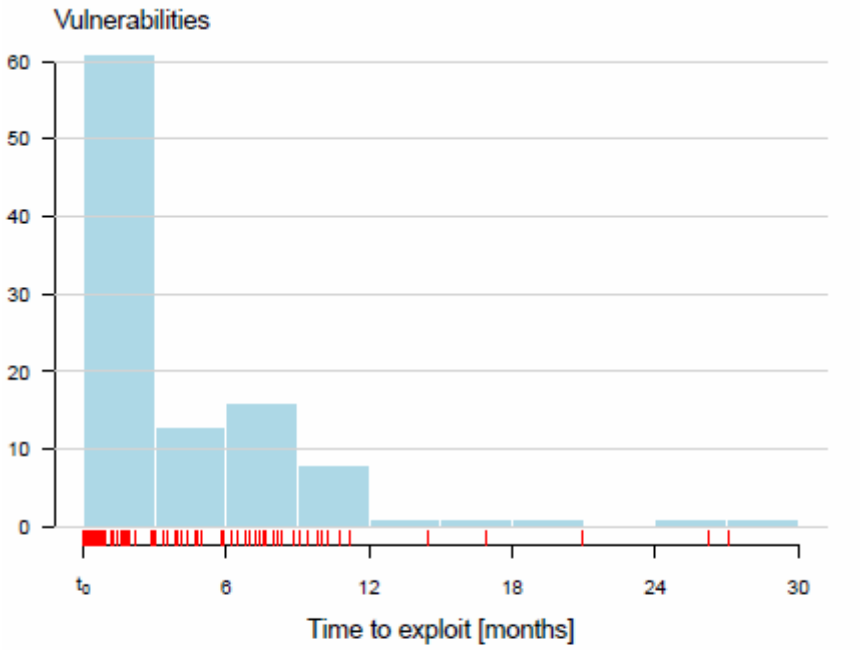
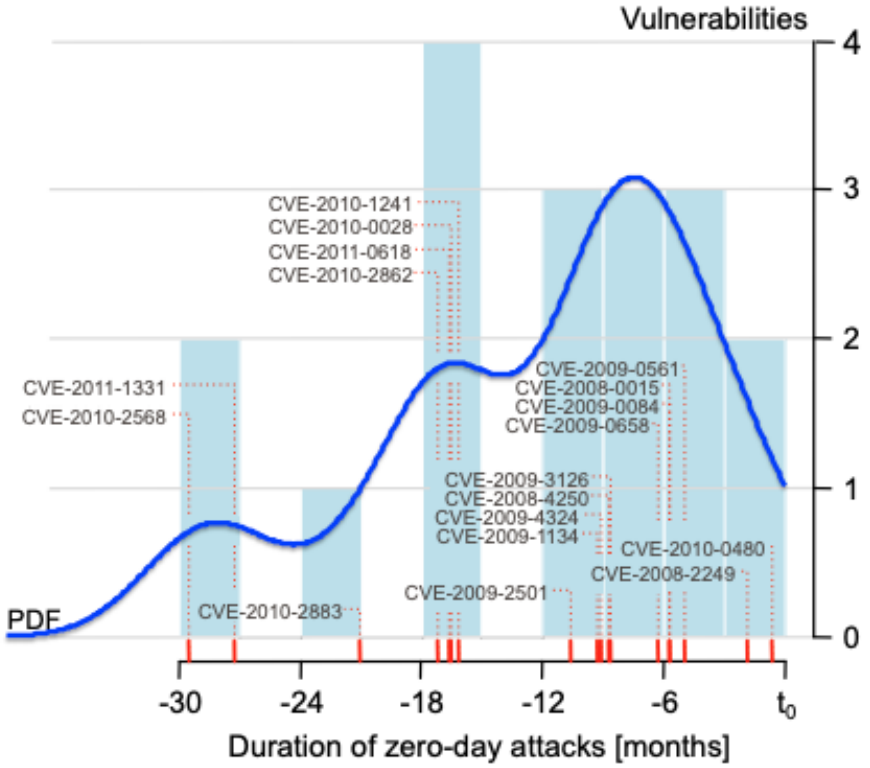


Figure 7: Time before vulnerabilities disclosed between 2008–2011 started being exploited in the field. The histograms group the exploitation lag in 3-month increments, after disclosure, and the red rug indicates the lag for each exploited vulnerability. The zero-day attacks are excluded from this figure.

Duration of zero-day attacks

- The zero-day attacks they identify lasted between
- 19 days (CVE-2010-0480) and
- 30 months (CVE-2010-2568), and
- the average duration of a zero-day attack is 312 days.

Before We Knew It An Empirical Study of Zero-Day Attacks In The Real World



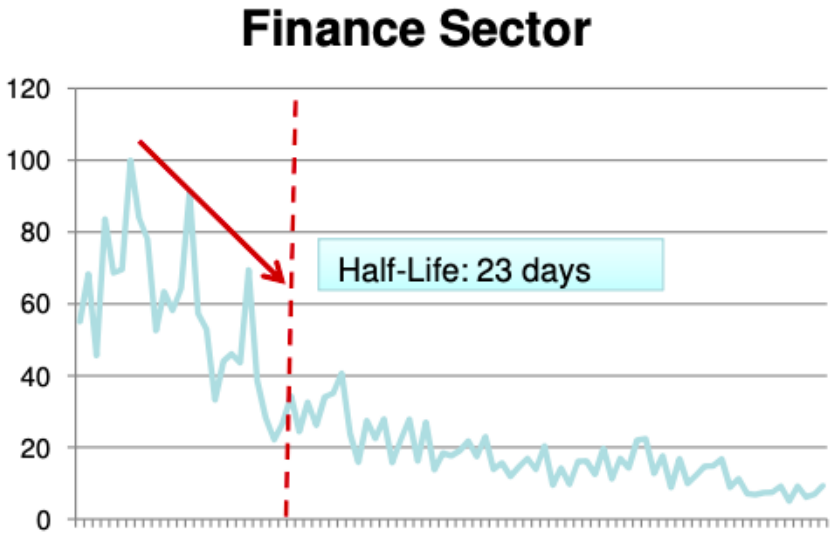
Colorado State University

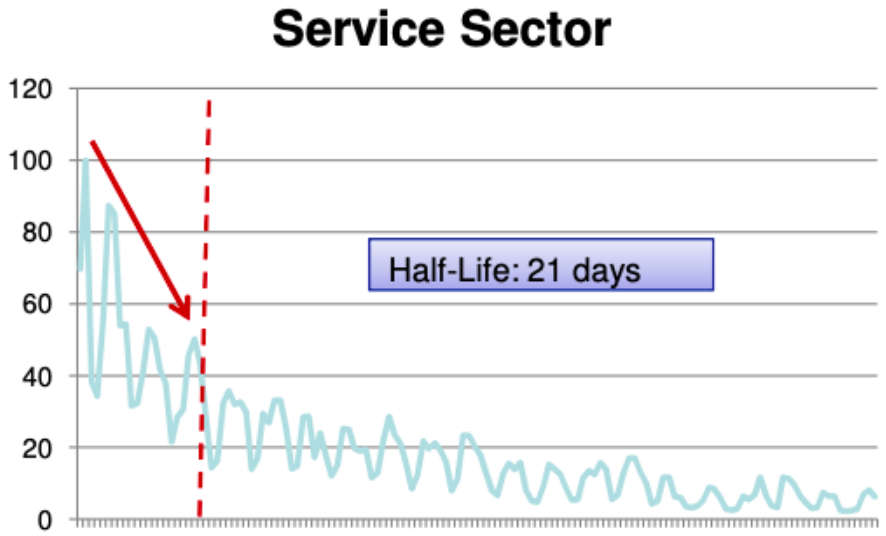
Qualys “Laws of Vulnerabilities”

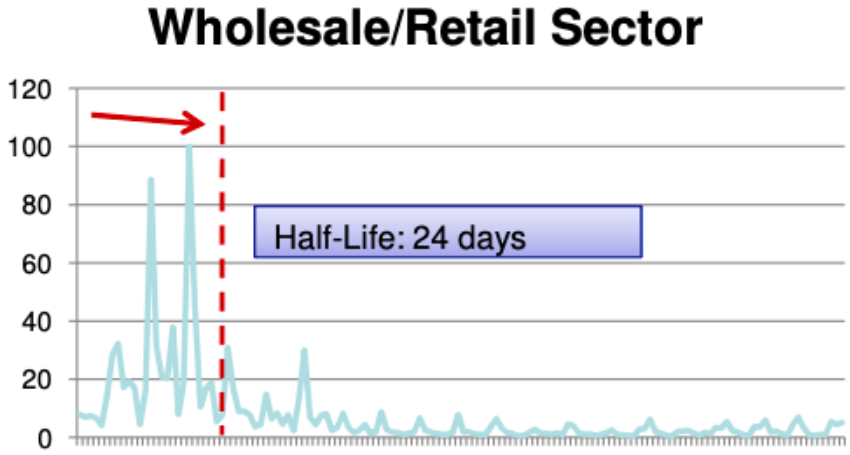
Gerhard Eschelbeck of Qualys

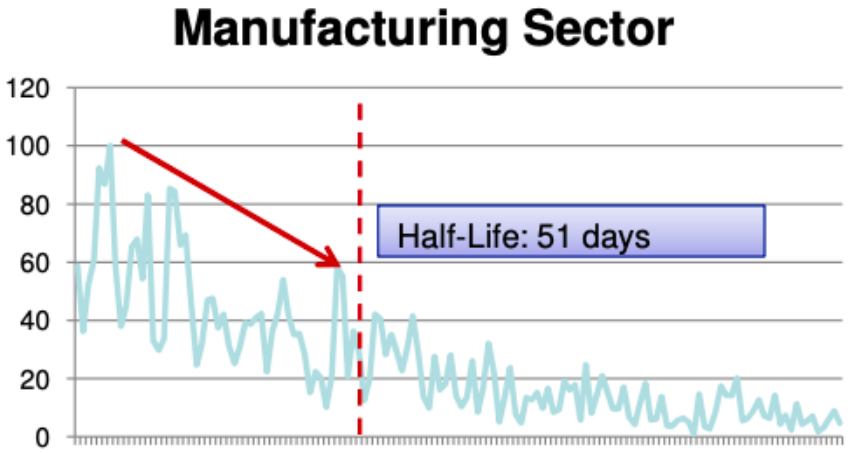
- 14 million Vulnerability scans performed with the QualysGuard vulnerability management service
- Centralized knowledge base with signatures for more than 4000 unique vulnerabilities.
- V2 2008
- 200 external (Internet)scanners and 5000+ internal (Intranet) scanners
- Data is anonymous and non traceable
- Interesting though somewhat outdated

Laws 2.0 – 1. Half-Life of attacks by Industry







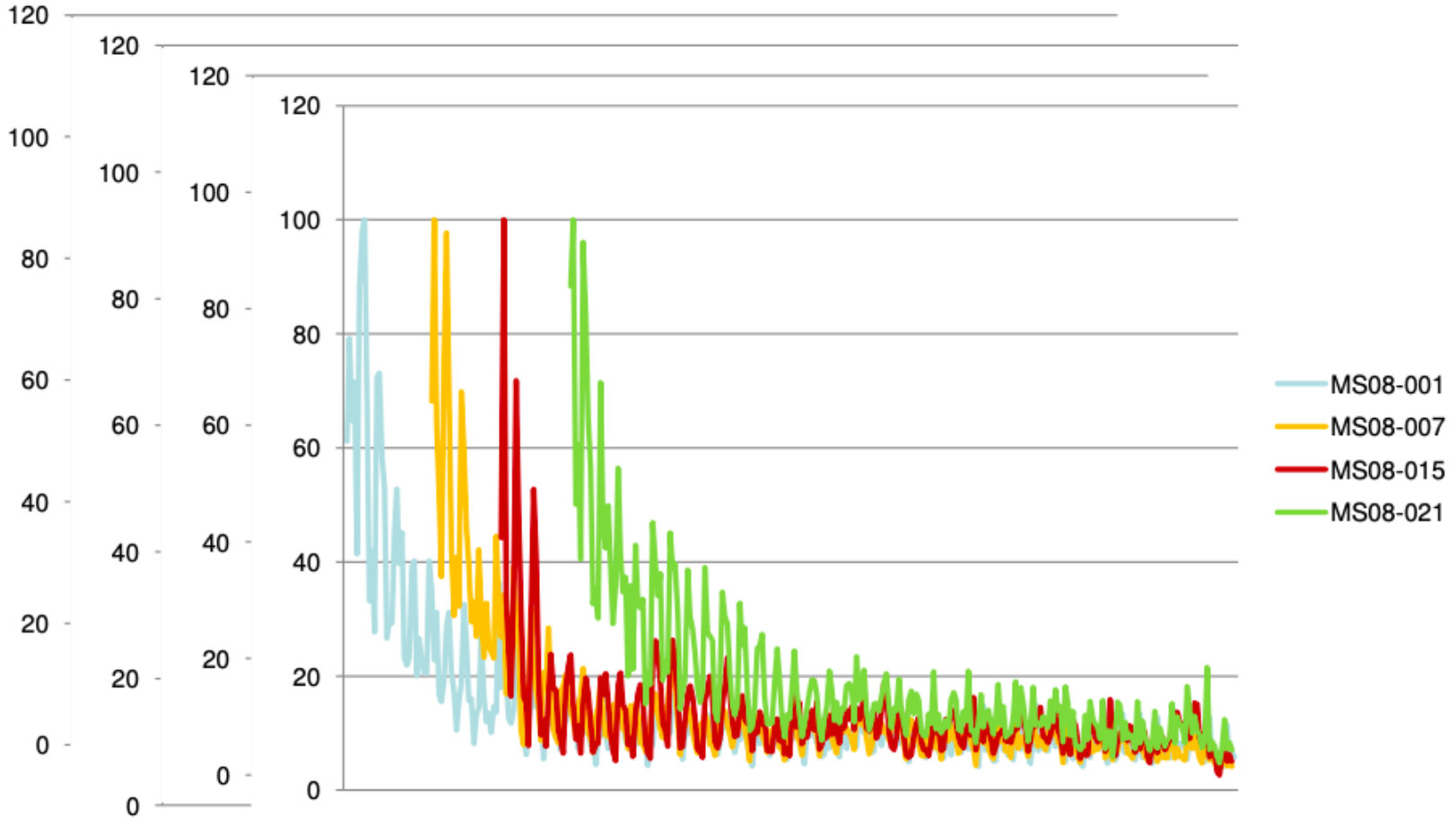


Colorado State University

Laws 2.0 – Persistence

most, if not all

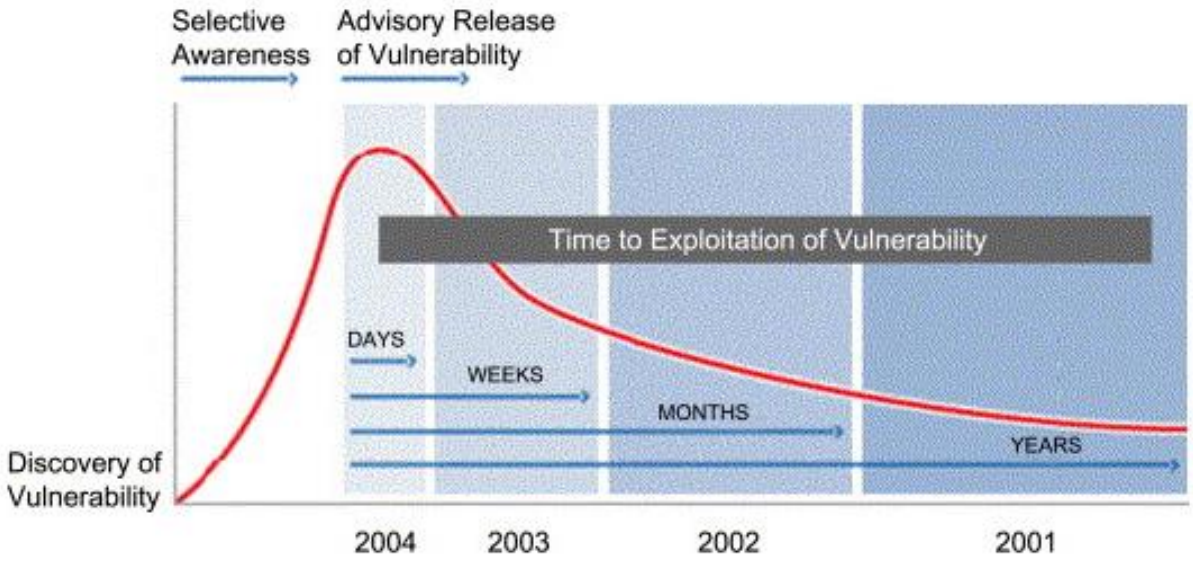
"The lifespan of some vulnerabilities is unlimited"



Colorado State University

Laws 2.0 – Exploitation

- Window for the availability of an exploit is constantly shrinking
- Attackers are professional and driven
- 0-day exploits–56 in Qualys knowledgebase
- Exploit availability is now measured in single=digit days



Gerhard Eschelbeck, The Laws of Vulnerabilities: Which security vulnerabilities really matter?, Information Security Technical Report, Volume 10, Issue 4, 2005, Pages 213-219.

Updated metrics

Over time, Qualys and security teams updated these core metrics

Original Law (2003)	Modern Updated Status & Findings
Law of Half-Life: The time it takes to reduce the number of open vulnerabilities by 50% was originally 30 days.	Drastically Shrunk: The updated Law of Half-Life shows the cycle shrinking significantly. The half-life dropped to 19 days for external systems and 48 days for internal systems.
Law of Prevalence: Critical vulnerabilities were historically concentrated primarily in server applications.	Over 60% of critical vulnerabilities shifted to client applications (browsers, media players, security tools). More recent updates track an aggressive evolution targeting identity infrastructure, control planes, and AI gateways.
Law of Exploitation: Originally noted that 80% of critical exploits appeared within 60 days of a public announcement.	Updates show that weaponized vulnerabilities are now actively exploited in 19.5 days on average, meaning attackers frequently strike before organizations can even begin patching.

Quantitative Security

Colorado State University
Yashwant K Malaiya

CS 559

Vulnerability Discovery Models



CSU Cybersecurity Center
Computer Science Dept

Modeling Vulnerability Discovery

- Quantitative Vulnerability Assessment Alhazmi 2004-2008
- Seasonality in Vulnerability Discovery Joh 2008,2009
- Discovery in Multi-Version Software Kim 2006,2007

Vulnerabilities



Colorado State University

Goal: Modeling Vulnerability Discovery

- Developing a quantitative model to estimate vulnerability discovery.
  - Using *calendar time*.
  - Using *equivalent effort*.
- Validate these measurements and models.
  - Testing the models using available data
- Identify security Assessment metrics
  - *Vulnerability density*
  - *Vulnerability to Total defect ratio*

AML VDM

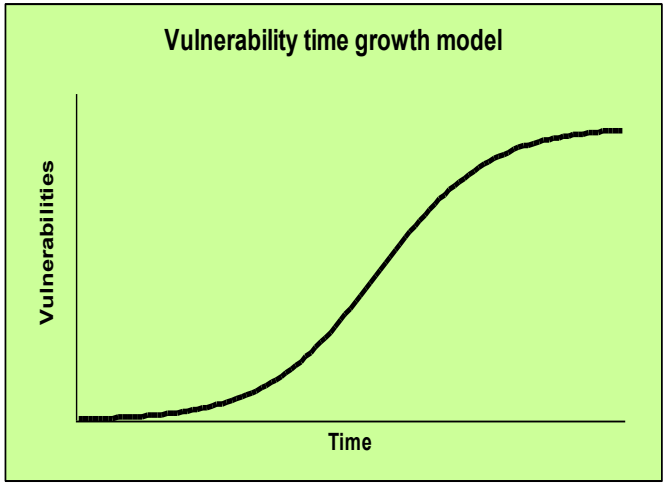
Alhazmi Malaiya Logistic (AML) vulnerability discovery model

$$\frac{d\Omega}{dt} = A\Omega(B - \Omega), \qquad (3)$$

where Ω is the cumulative number of vulnerabilities, t is the calendar time, and initially t=0. A and B are empirical constants determined from the recorded data.

By solving the differential equation we obtain

$$\Omega(t) = \frac{B}{BCe^{-ABt} + 1}, \qquad (4)$$

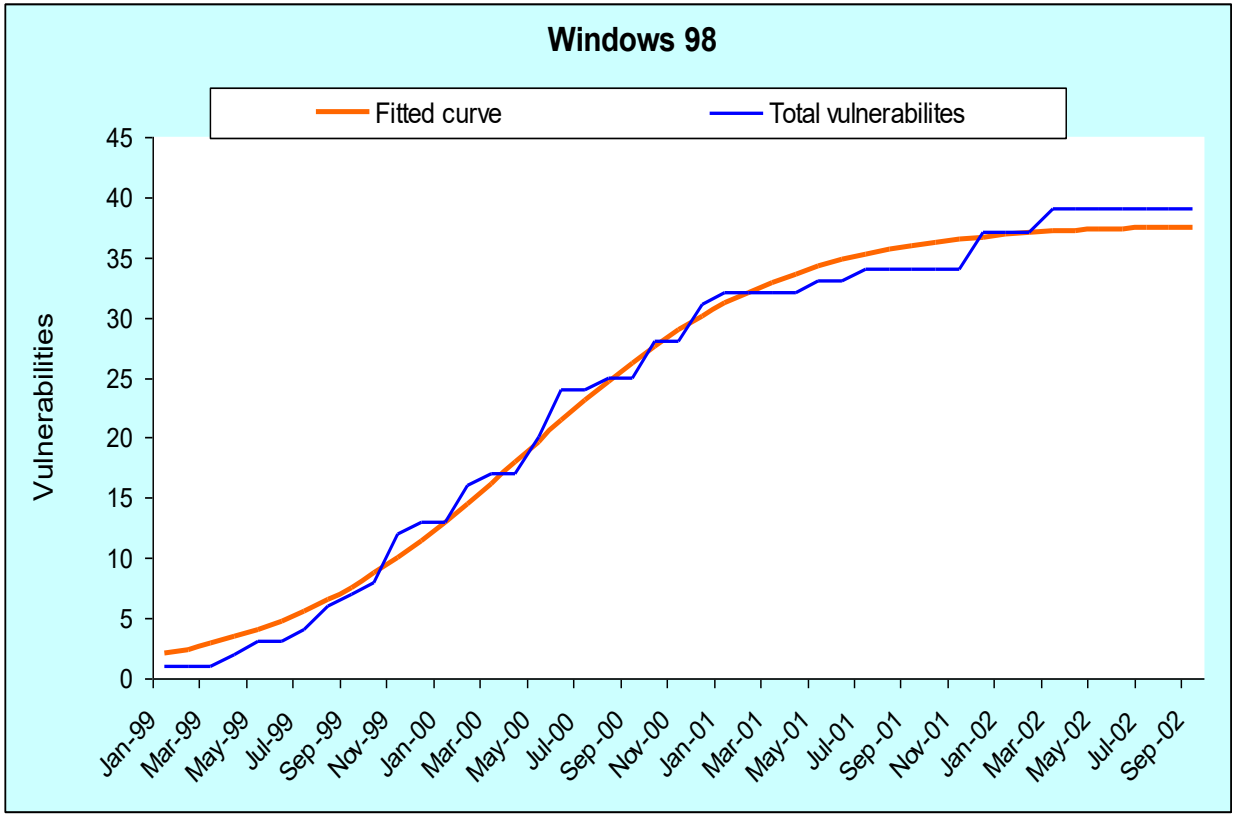


3 phase model S-shaped model.

- Phase 1: Installed base –low.
- Phase 2: Installed base–higher and growing/stable.
- Phase 3: Installed base–dropping.

O. H. Alhazmi and Y. K. Malaiya, "Quantitative Vulnerability Assessment of Systems Software Proc. Ann. IEEE Reliability and Maintainability Symp., 2005, pp. 615-620

Time–based model: Windows 98



	Windows 98
A	0.004873
B	37.7328
C	0.5543
X^2	7.365
$X^2_{critial}$	60.481
P-value	$1- 7.6x10^{-11}$

Colorado State University

Usage –vulnerability Discovery model

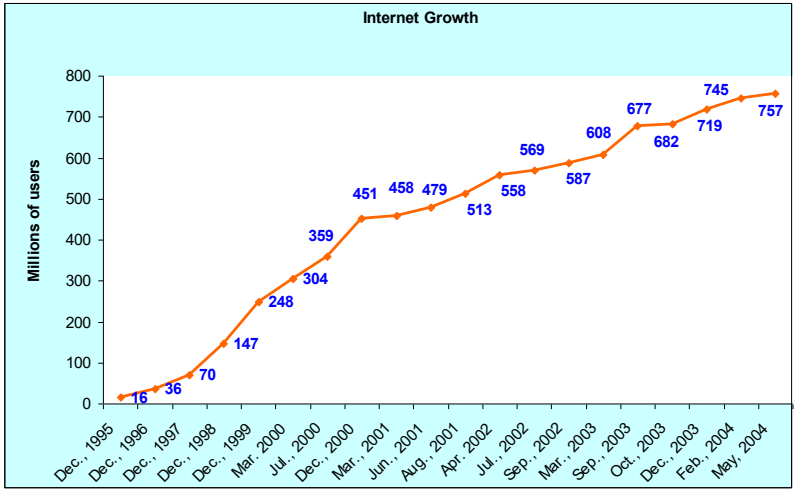
The data:

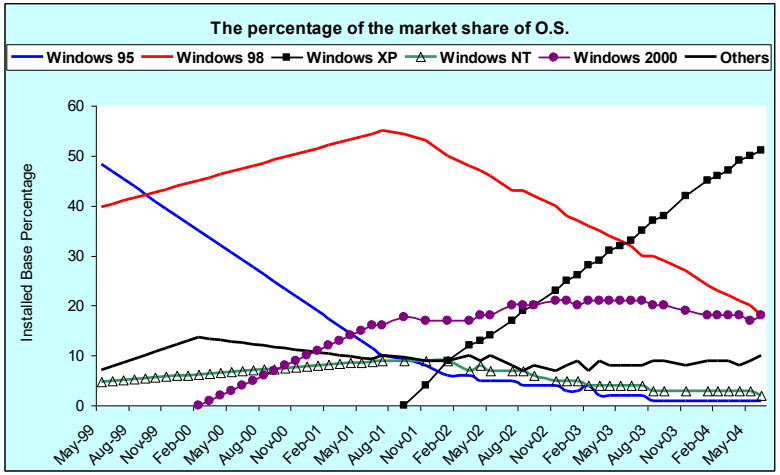
- The global internet population.
- The market share of the system during a period of time.

Equivalent effort

- The real environment performs an intensive testing.
- Malicious activities is relevant to overall activities.
- Defined as

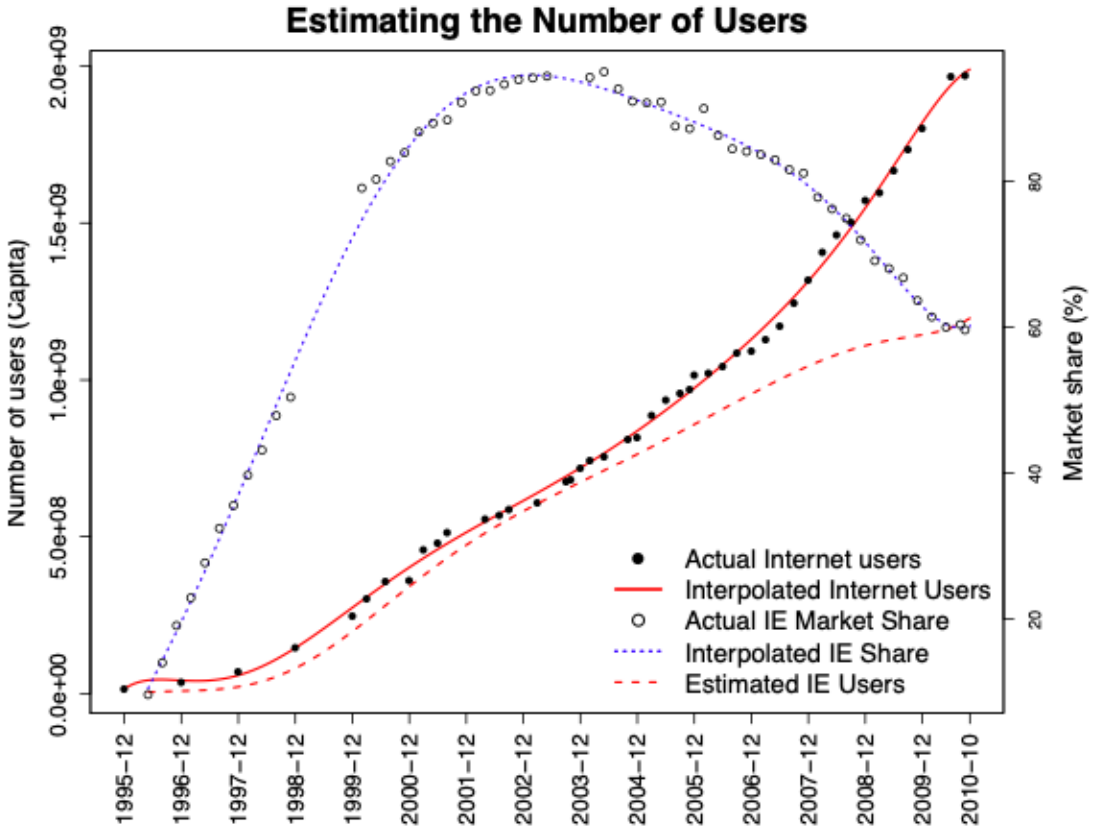
$$E = \sum_{i=0}^{n} (U_i \times P_i)$$





Colorado State University

Estimating number of users



Estimating the number of IE users (internet users x market share)

QUANTITATIVE ANALYSES OF SOFTWARE VULNERABILITIES, HyunChul Joh, PhD Dissertation 2011

Colorado State University

Software Reliability Modeling

- Applicable to general software bugs
- Key Static software metrics
  - Software size (without comments, KLOC)
  - Defect density (total defects/size)
    - Typical range Range 16 -0.1 /KLOC
    - Software evolution/reuse, requirement volatility
    - Team capabilities, extent of testing
  - Defect finding efficiency
- Software reliability growth models (SRGMs)

Exponential SRGM

Exponential Reliability Growth Model

- Assumption: rate of finding and removing bugs is proportional to the number of bugs present at time t.

$$-\frac{dN(t)}{dt} = \beta_1 N(t)$$

Which yields

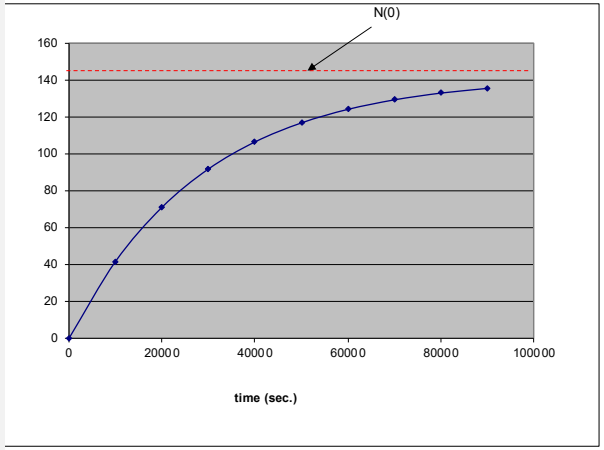
$$N(t) = N(0)e^{-\beta_1 t}$$

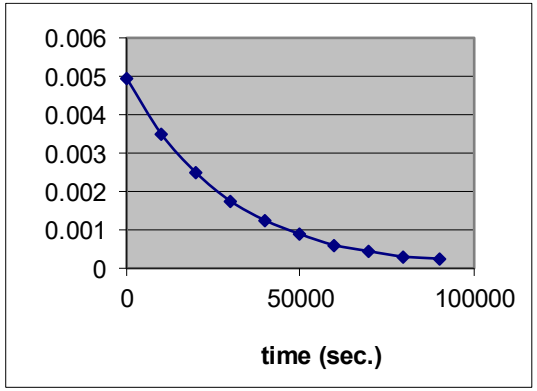
- Cumulative number of defects found is

$$N(0)(1 - e^{-\beta_1 t})$$

- Defect finding rate is

$$N(0)e^{-\beta_1 t}$$



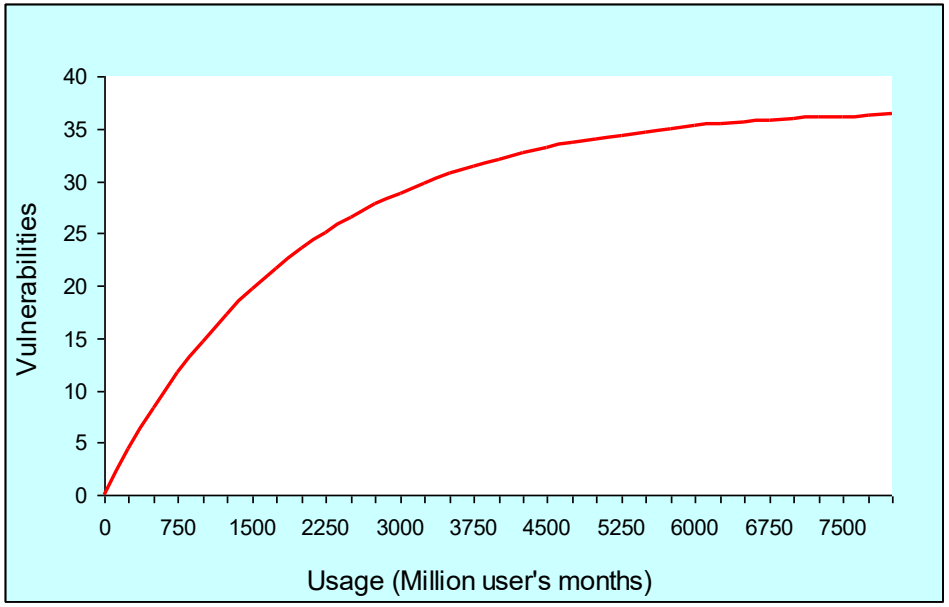


- N(0) may be estimated using defect density and size
- β_1 depends to defect finding efficiency

Colorado State University

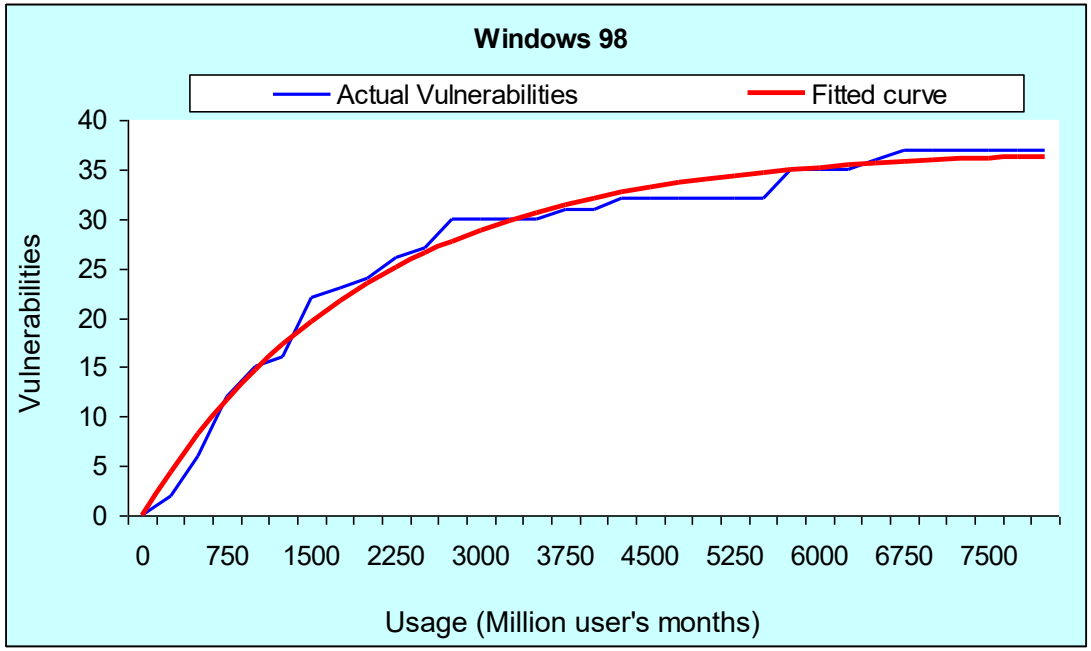
Usage –vulnerability Discovery model

- The model: growth with effort.
- Growth model based on the exponential SRGM [Musa].
- Time is eliminated.
- $y = N(0)(1 - e^{-\beta_1 E})$



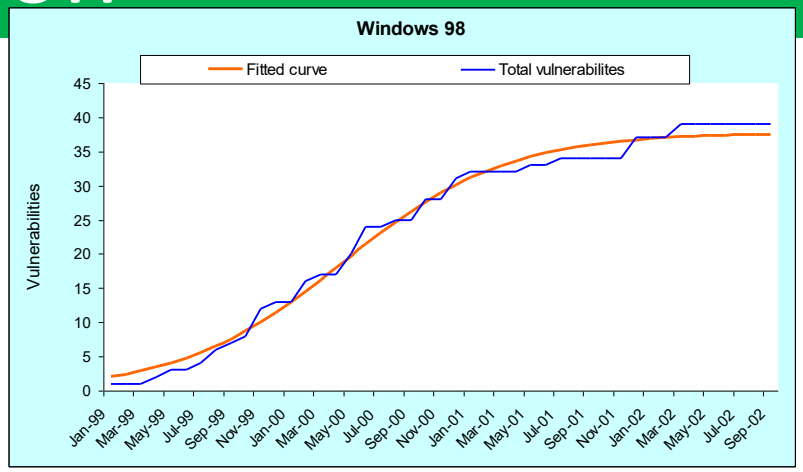
Colorado State University

Effort-based model: Windows 98



	Windows 98
B	37
λ_{vu}	0.000505
χ^2	3.510
$\chi^2_{critial}$	44.9853
P-value	1- 3.3×10^{-11}

Discussion



- Excellent fit for Windows 98 and NT 4.0.
- Model fits data for all OSs examined.
- Deviation from the model caused by overlap:
  - Windows 98 and Windows XP
  - Windows NT 4.0 and Windows 2000
- Vulnerabilities in shared code may be detected in the newer OS.
- Need: approach for handling such overlap

Quantitative Security

Colorado State University
Yashwant K Malaiya

CS 559
Multi-version Systems



CSU Cybersecurity Center
Computer Science Dept

Motivation for Multi-version VDMs

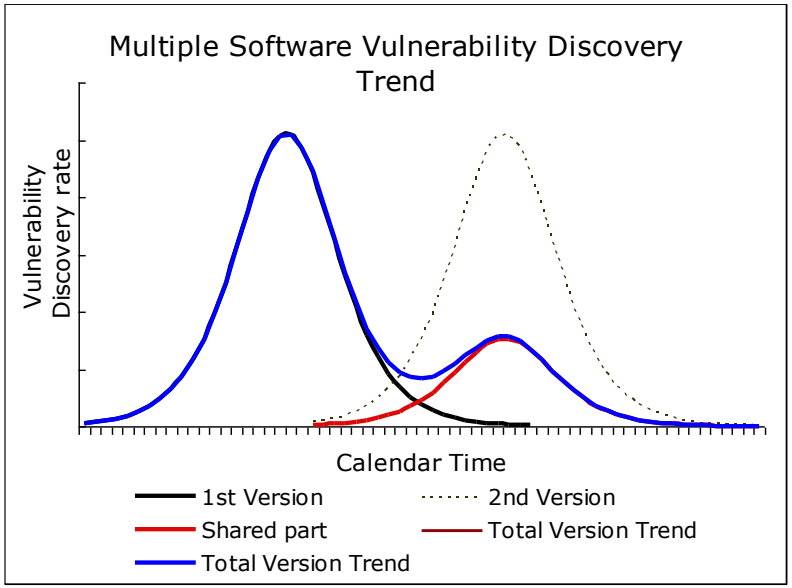
- Superposition effect on vulnerability discovery process due to shared code in successive versions.
- Examination of software evolution: impact on vulnerability introduction and discovery
- Other factors impacting vulnerability discovery process not considered before

48

Software Reuse

- New software projects use both new and reused blocks.
  - New blocks have a higher defect density because they have undergone less testing.
  - Reused blocks are more reliable.
  - Some defects may be introduced at the new/reused block interface.
  - Overall defect density is weighted average of the two.
  - Encounter rate during execution depends on weighted usage

Multi-version Vulnerability Discovery



$$\Omega(t) = \frac{B}{BCe^{-ABt}+1} + \alpha \frac{B'}{B'C'e^{-A'B'(t-\varepsilon)}+1}$$

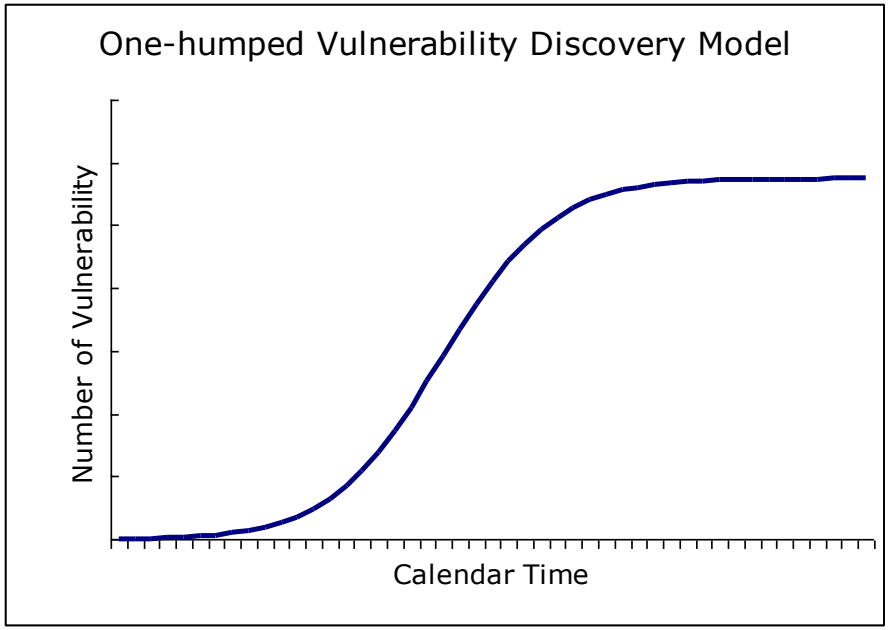
Parameter α indicates shared components such as shared code and shared functionality, and the parameter ε is the time lag between the release dates of the two versions.

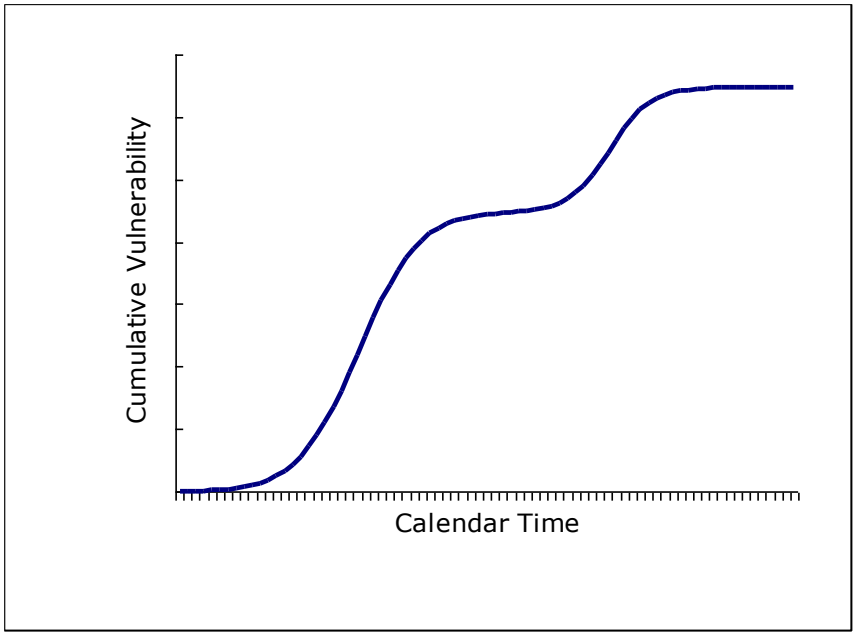
	Previous Version	Next Version	Shared Code Ratio α
Apache	1.3.24 (3-21-2002)	2.0.35 (4-6-2002)	79.84%
Mysql	4.1.1 (12-1-2003)	5.0.0 (12-22-2003)	83.52%

http://www.cs.colostate.edu/~malaiya/p/kim07.pdf

50

One vs Two Humps

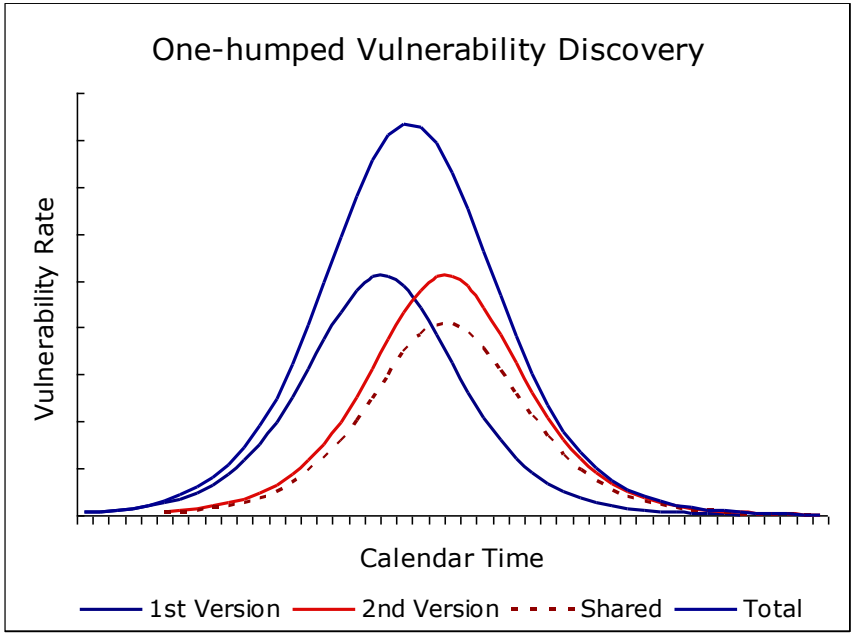


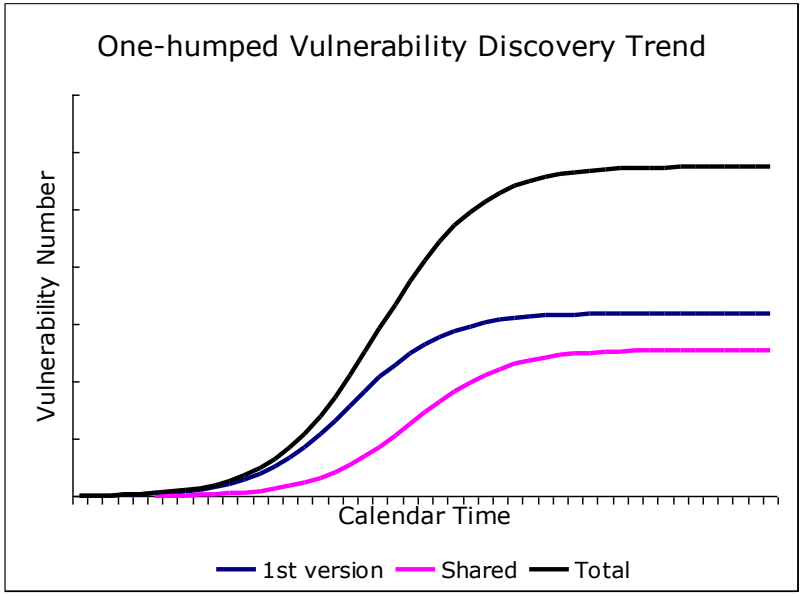


Superposition affect

51

Multi-version Vulnerability Discovery





- May result in a single hump with prolonged linear period

52

Evolving Programs

Gradually evolving software

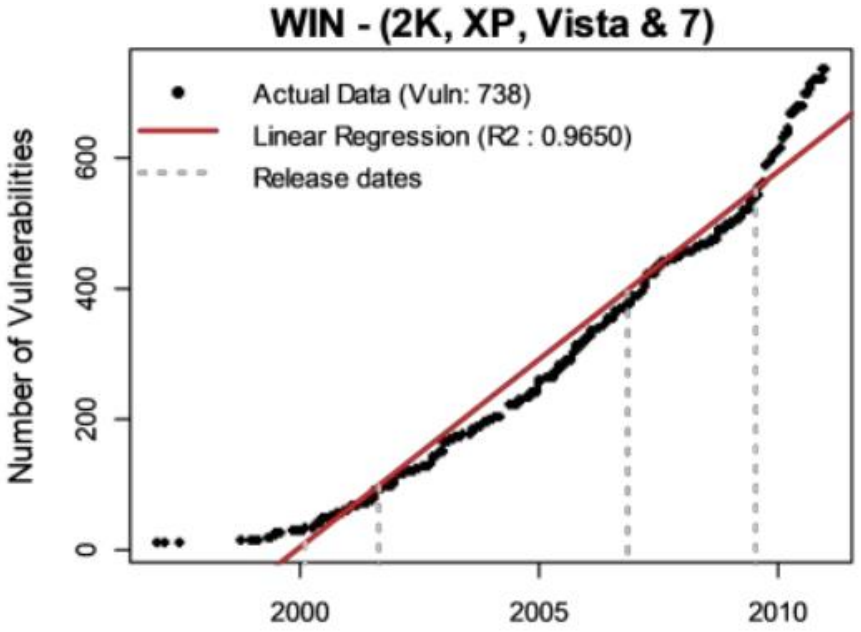
Software evolves in each version.

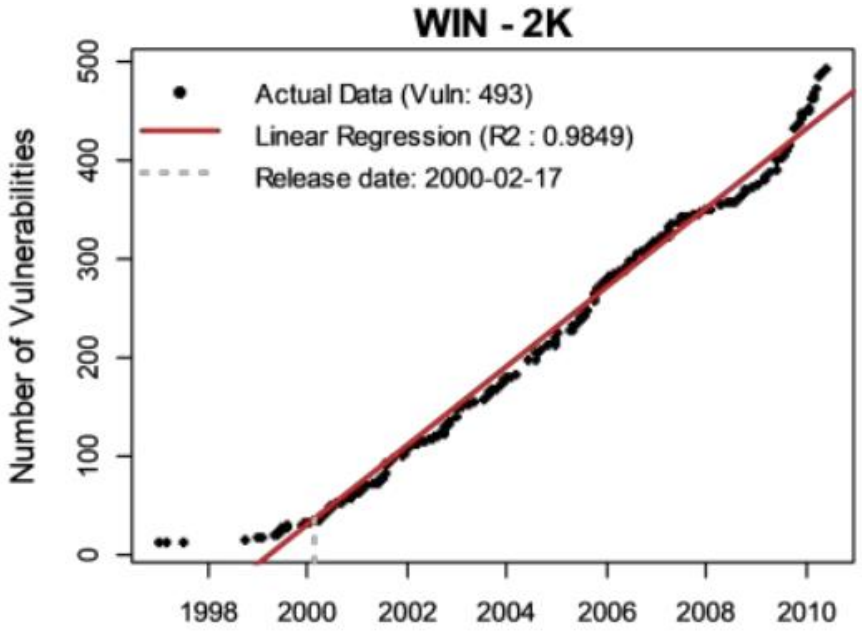
- Existing code fixed
  - some vulnerabilities found and patched
- Code added for increasing functionality
  - New vulnerabilities injected
  - Total number of vulnerabilities may remain about the same
- Overall code size keeps increasing
  - Vulnerability discovery rate may remain stable

53

Linear model

- Because of nearly continuous evolution, the linear phase may get stretched.



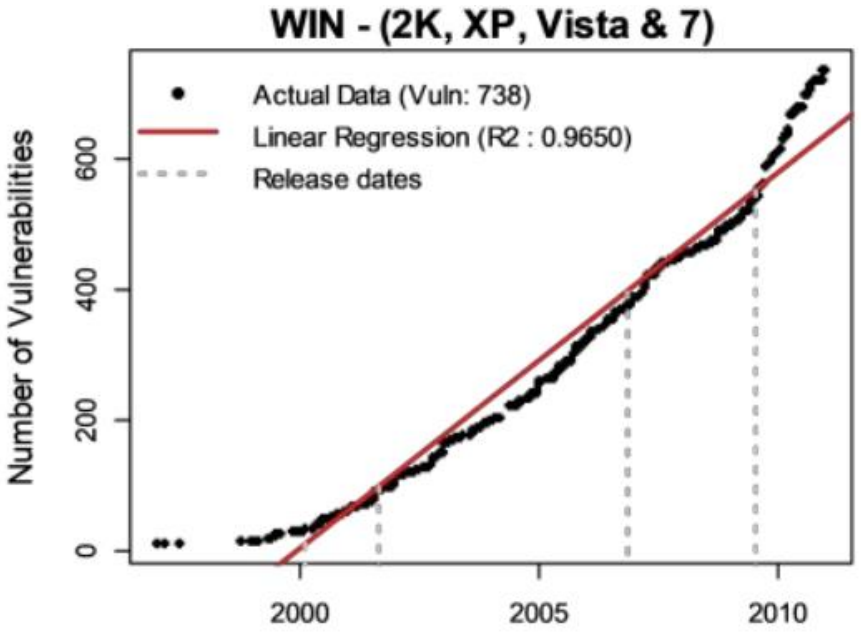


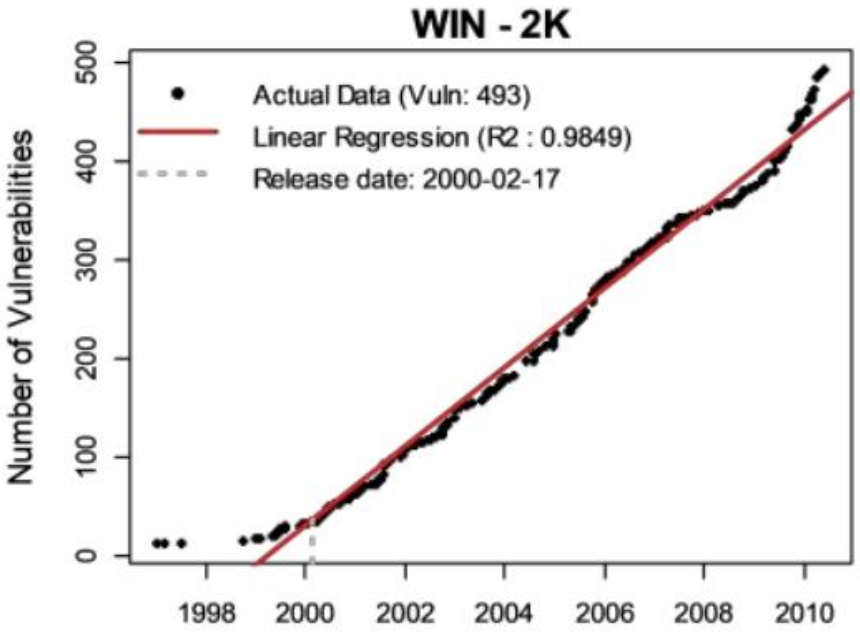
- If the evolution rate is steady, the size of the pool of undiscovered vulnerabilities stays the same (vulnerabilities removal rate = injection rate)
- If the market share is steady, the number of vulnerability finders remains steady.

Joh's thesis

54

Linear model





- Four Windows releases: 500 vulnerabilities during July 1998-July 2009
- Size: 35-50 M LOC
- Slope = about 45 vulnerabilities/year
- Further investigation is needed.

Data from Joh's thesis

55