Zero Trust Theorem

Boiling Frogs 2019, Wrocław

Andrzej Dyjak

🤝

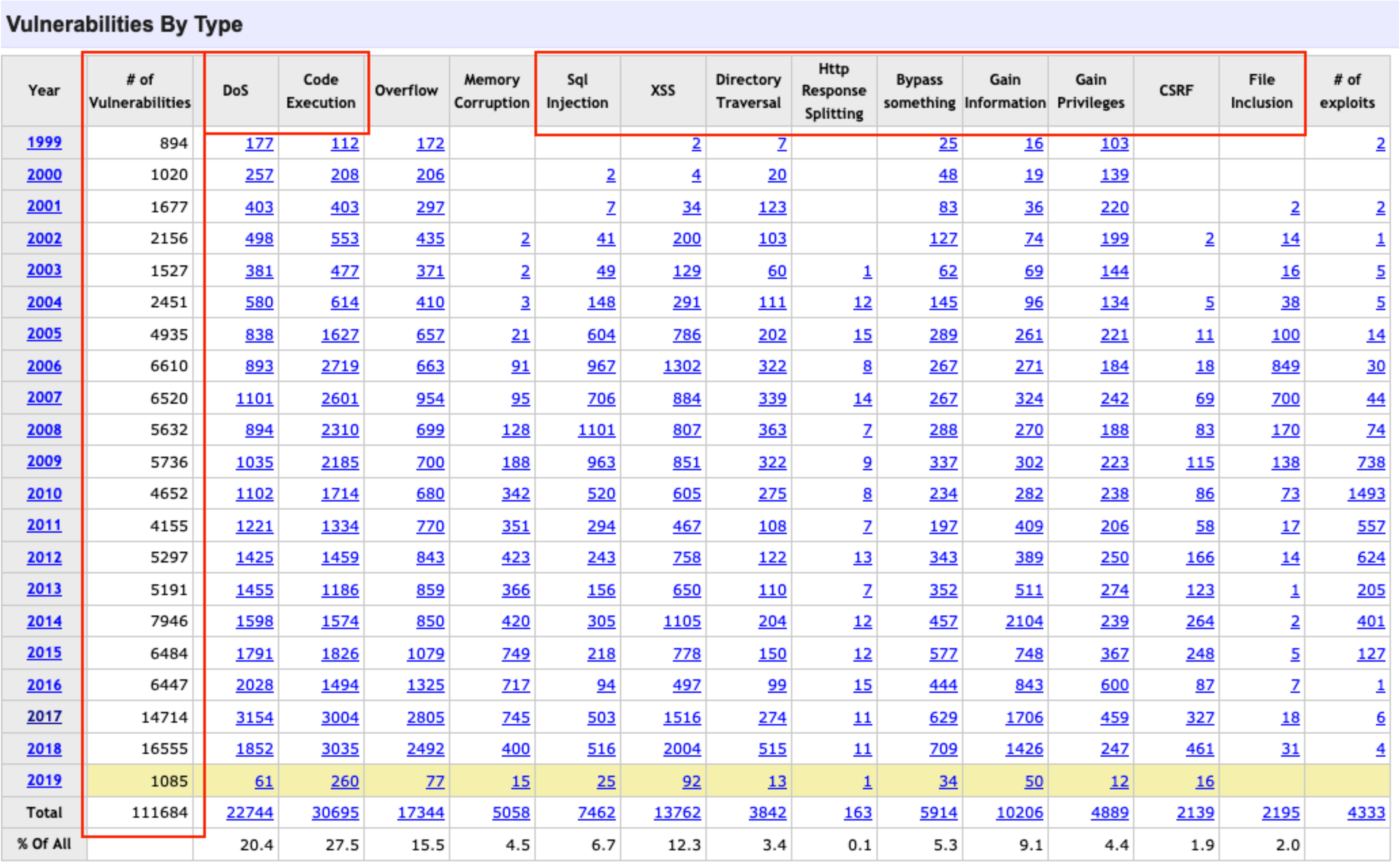
# whoami

Preludium

- O czym będę opowiadał (hint: AppSec)
- W jaki sposób będę o tym opowiadał (hint: praktycznie)

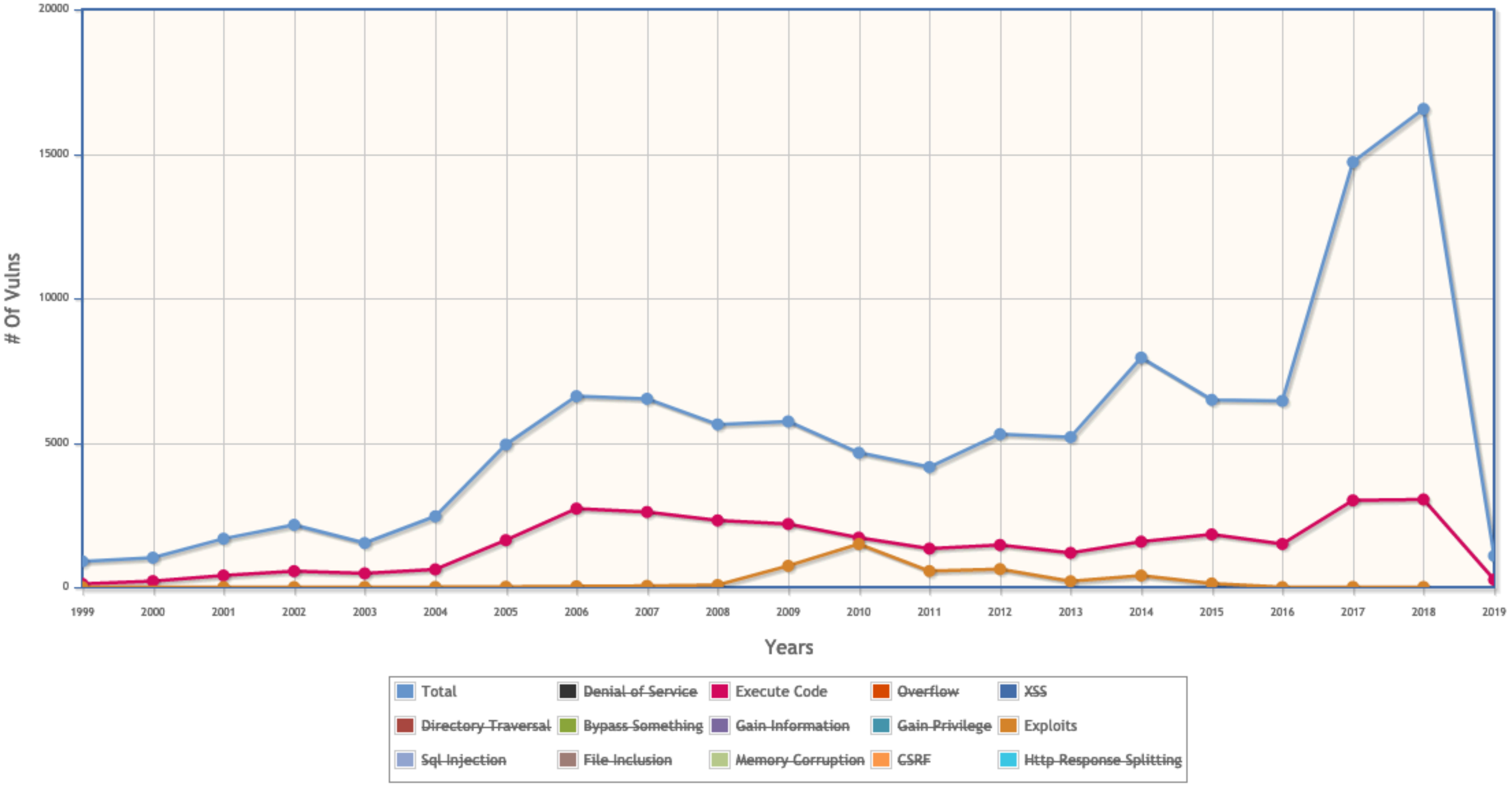
Web aplikacje

😱



Vulnerabilities By Type

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
1999	894	177	112	172			2	7		25	16	103			2
2000	1020	257	208	206		2	4	20		48	19	139			
2001	1677	403	403	297		7	34	123		83	36	220		2	2
2002	2156	498	553	435	2	41	200	103		127	74	199	2	14	1
2003	1527	381	477	371	2	49	129	60	1	62	69	144		16	5
2004	2451	580	614	410	3	148	291	111	12	145	96	134	5	38	5
2005	4935	838	1627	657	21	604	786	202	15	289	261	221	11	100	14
2006	6610	893	2719	663	91	967	1302	322	8	267	271	184	18	849	30
2007	6520	1101	2601	954	95	706	884	339	14	267	324	242	69	700	44
2008	5632	894	2310	699	128	1101	807	363	7	288	270	188	83	170	74
2009	5736	1035	2185	700	188	963	851	322	9	337	302	223	115	138	738
2010	4652	1102	1714	680	342	520	605	275	8	234	282	238	86	73	1493
2011	4155	1221	1334	770	351	294	467	108	7	197	409	206	58	17	557
2012	5297	1425	1459	843	423	243	758	122	13	343	389	250	166	14	624
2013	5191	1455	1186	859	366	156	650	110	7	352	511	274	123	1	205
2014	7946	1598	1574	850	420	305	1105	204	12	457	2104	239	264	2	401
2015	6484	1791	1826	1079	749	218	778	150	12	577	748	367	248	5	127
2016	6447	2028	1494	1325	717	94	497	99	15	444	843	600	87	7	1
2017	14714	3154	3004	2805	745	503	1516	274	11	629	1706	459	327	18	6
2018	16555	1852	3035	2492	400	516	2004	515	11	709	1426	247	461	31	4
2019	1085	61	260	77	15	25	92	13	1	34	50	12	16		
Total	111684	22744	30695	17344	5058	7462	13762	3842	163	5914	10206	4889	2139	2195	4333
% Of All		20.4	27.5	15.5	4.5	6.7	12.3	3.4	0.1	5.3	9.1	4.4	1.9	2.0	


20000
15000
10000
5000
0
# Of Vulns
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
Years
🟦 Total
⬛ ~~Denial of Service~~
🟥 Execute Code
🟧 ~~Overflow~~
🟦 ~~XSS~~
🟥 ~~Directory Traversal~~
🟩 ~~Bypass Something~~
🟪 ~~Gain Information~~
🟦 ~~Gain Privilege~~
🟧 Exploits
🟦 ~~Sql Injection~~
🟫 ~~File Inclusion~~
🟩 ~~Memory Corruption~~
🟧 ~~CSRF~~
🟦 ~~Http Response Splitting~~

Przykłady

- Wybór jest tak duży, że trudno było się zdecydować więc…
- Prywatna historia o XSS i RCE
  - XSS — Cross-Site Scripting (w tym wypadku *stored*)
  - RCE — Remote Code/Command Execution (w tym wypadku *code*)

Przeciwdziałanie

- Używanie powszechnie uznanych frameworków (zasada “*Given a thousand eyes, all bugs are shallow.*” — Linus)
- Podniesienie higieny wytwarzania oprogramowania
  - Testowanie pod kątem uznanych standardów/wytycznych (e.g. OWASP ASVS, OWASP Top 10, etc)
  - Secure by Design — wbudowanie security w proces wytwarzania oprogramowania via Secure SDLC / DevSecOps (OWASP SAMM, Synopsys’ BSIMM)

Zewnętrzne komponenty

Przykłady

- Neex i bug (OS command injection) w sposobie wywoływania narzędzia z pakietu GraphicsMagick — Imgur

- Chris Evans i bugi (memory disclosure) w ImageMagick — podatne wersje zidentyfikowane na serwerach aplikacyjnych od m.in. Dropbox czy Yahoo!

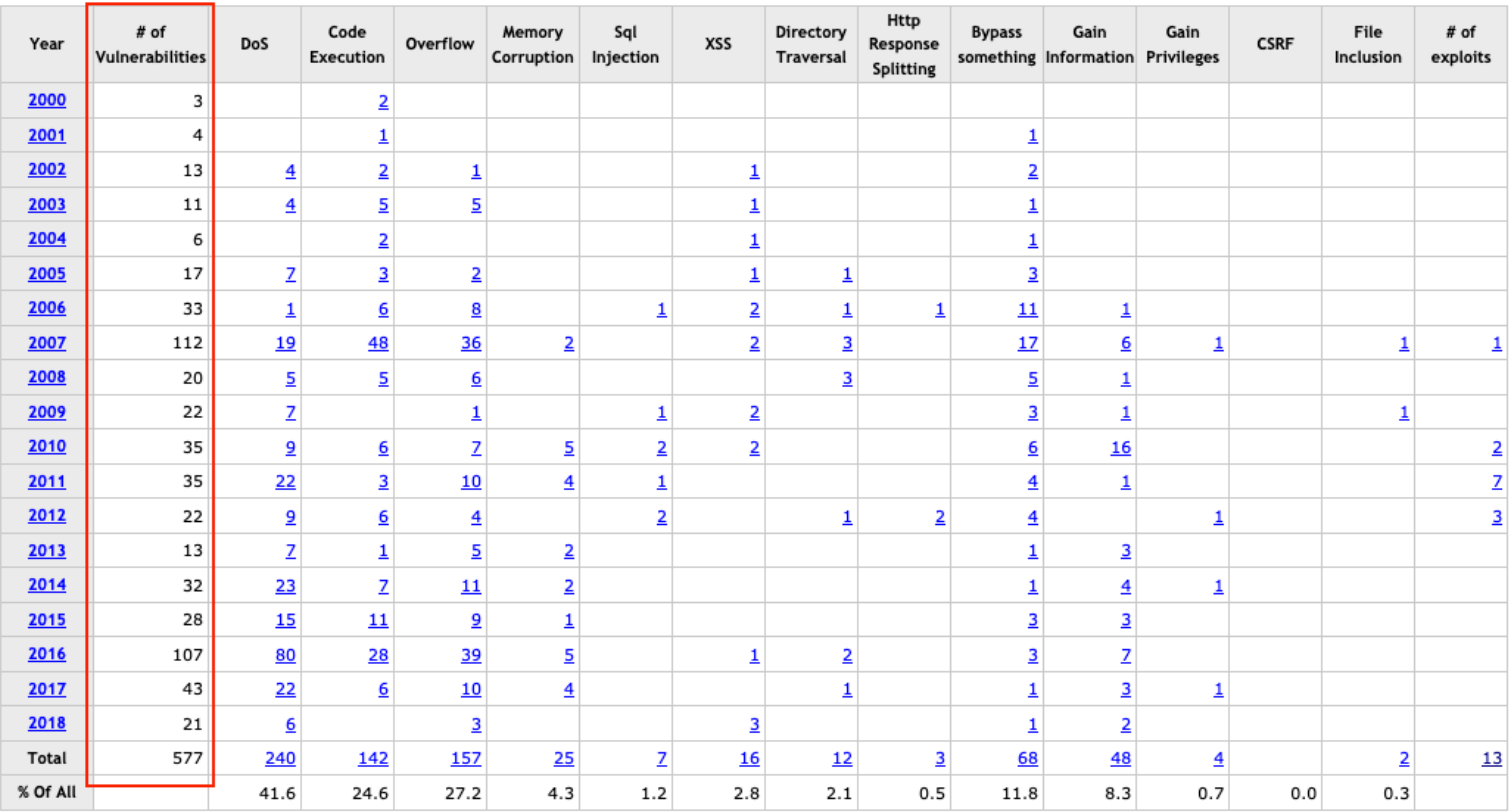
Przeciwdziałanie

- Świadomy wybór zewnętrznych komponentów
  - Mniejsza powierzchnia ataku = mniejsze ryzyko
- Stosowanie zasady *least-privilege required* (e.g. via sandbox)

📚

- https://scarybeastsecurity.blogspot.co.uk/2017/05/proving-missing-aslr-on-dropboxcom-and.html
- https://scarybeastsecurity.blogspot.co.uk/2017/05/0day-proving-boxcom-fixed-aslr-via.html
- https://scarybeastsecurity.blogspot.co.uk/2017/05/bleed-more-powerful-dumping-yahoo.html
- https://scarybeastsecurity.blogspot.co.uk/2017/05/bleed-continues-18-byte-file-14k-bounty.html
- https://hackerone.com/reports/212696
- https://github.com/neex/gifoeb
- https://4lemon.ru/2017-01-17_facebook_imagetragick_remote_code_execution.html
- https://blog.sigsegv.pl/external-third-party-resources-and-your-web-application/
- https://onedrive.live.com/view.aspx?resid=2664E65DD698885E!120&ithint=file%2cpptx&app=PowerPoint&authkey=!AK39RoVxiJ5re8Y
- https://medium.com/@ilja.bv/yet-another-memory-leak-in-imagemagick-or-how-to-exploit-cve-2018-16323-a60f048a1e12
- https://en.wikipedia.org/wiki/Principle_of_least_privilege

Interpretery
&
Wirtualne Maszyny (JVM, CLR, etc)



Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2000	3		2												
2001	4		1							1					
2002	13	4	2	1			1			2					
2003	11	4	5	5			1			1					
2004	6		2				1			1					
2005	17	7	3	2			1	1		3					
2006	33	1	6	8		1	2	1	1	11	1				
2007	112	19	48	36	2		2	3		17	6	1		1	1
2008	20	5	5	6				3		5	1				
2009	22	7		1		1	2			3	1			1	
2010	35	9	6	7	5	2	2			6	16				2
2011	35	22	3	10	4	1				4	1				7
2012	22	9	6	4		2		1	2	4		1			3
2013	13	7	1	5	2					1	3				
2014	32	23	7	11	2					1	4	1			
2015	28	15	11	9	1					3	3				
2016	107	80	28	39	5		1	2		3	7				
2017	43	22	6	10	4			1		1	3	1			
2018	21	6		3			3			1	2				
Total	577	240	142	157	25	7	16	12	3	68	48	4		2	13
% Of All		41.6	24.6	27.2	4.3	1.2	2.8	2.1	0.5	11.8	8.3	0.7	0.0	0.3	

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2010	1		1												
2011	3														
2012	59	3	1							2					
2013	180	1	10	4	4		1			32					2
2014	115	1	1												
2015	80														
2016	37		1	1							1	1			
2017	69	14								1	2				
2018	53	16	2								4				
2019	2														
Total	599	35	16	5	4		1			35	7	1			2
% Of All		5.8	2.7	0.8	0.7	0.0	0.2	0.0	0.0	5.8	1.2	0.2	0.0	0.0	

Przykłady

- Deserializacja parametru `cookie`, oraz memory corruption w natywnej implementacji funkcji `unserialize()` w PHP (Zend) — PornHub

- “The worst bug bounty ever” — bardzo drogi romans Shopify z mruby

- “Exposing Hidden Exploitable Behaviors in Programming Languages Using Differential Fuzzing” — ciekawe i niebezpieczne zachowania interpreterów

- Własny vulnerability research popularnych interpreterów (for fun & no profit)

	PHP (php-7.1.1-asan)	HHVM (hhvm-3.15.6-dev)	Ruby (mri-2.4.0-asan)	Python (cpython-2.7.13-asan)
EXPLOITABLE	**58**	**35**	**74**	**2**
PROBABLY_EXPLOITABLE	**8**	**0**	**0**	**2**
PROBABLY_NOT_EXPLOITABLE	**8**	**0**	**2**	**4**
UNKNOWN	**12**	**5**	**5**	**3**

Przeciwdziałanie

- Stosowanie zasady *least-privilege required* (e.g. via sandbox)
- Banowanie problematycznych funkcjonalności
  - **Softcore:** Na poziomie Code Review / SCM 😀
  - **Hardcore:** Na poziomie interpretera (wycięcie funkcjonalności i rekompilacja 😎)
    - Rekompilacja? Hm… 🤔

📚

- https://www.evonide.com/how-we-broke-php-hacked-pornhub-and-earned-20000-dollar/
- https://www.evonide.com/fuzzing-unserialize/
- https://sean.heelan.io/2017/08/12/fuzzing-phps-unserialize-function/
- https://externals.io/message/100147
- https://bugs.php.net/bug.php?id=75006
- http://mruby.sh/201703261726.html
- https://www.blackhat.com/docs/eu-17/materials/eu-17-Arnaboldi-Exposing-Hidden-Exploitable-Behaviors-In-Programming-Languages-Using-Differential-Fuzzing-wp.pdf
- https://github.com/dyjakan/interpreter-bugs
- https://github.com/rust-fuzz
- https://hackernoon.com/python-sandbox-escape-via-a-memory-corruption-bug-19dde4d5fea5

Kompilatory

Przykłady

- “Reflections on Trusting Trust” — Ken Thompson
- CVE-2018-1037 — .PDB Heap Memory Disclosure w Visual Studio (j00ru (Project Zero) 🤘)

Przeciwdziałanie

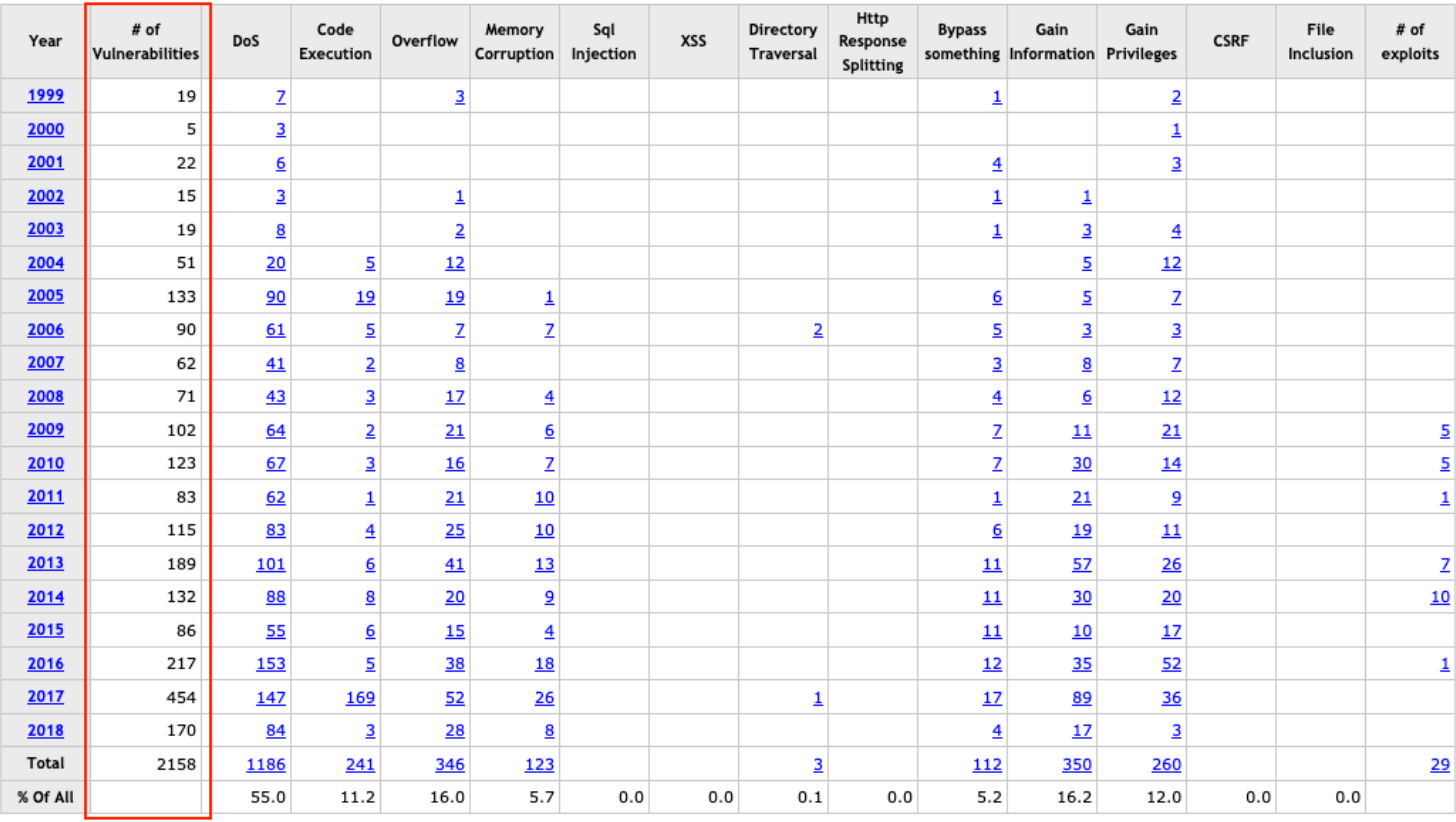
- Brak skalowalnej aktywnej ochrony
- Pasywne monitorowanie systemów pod kątem integralności

📚

- https://www.ece.cmu.edu/~ganger/712.fall02/papers/p761-thompson.pdf
- https://twitter.com/j00ru/status/985894472478265344
    - https://bugs.chromium.org/p/project-zero/issues/detail?id=1500

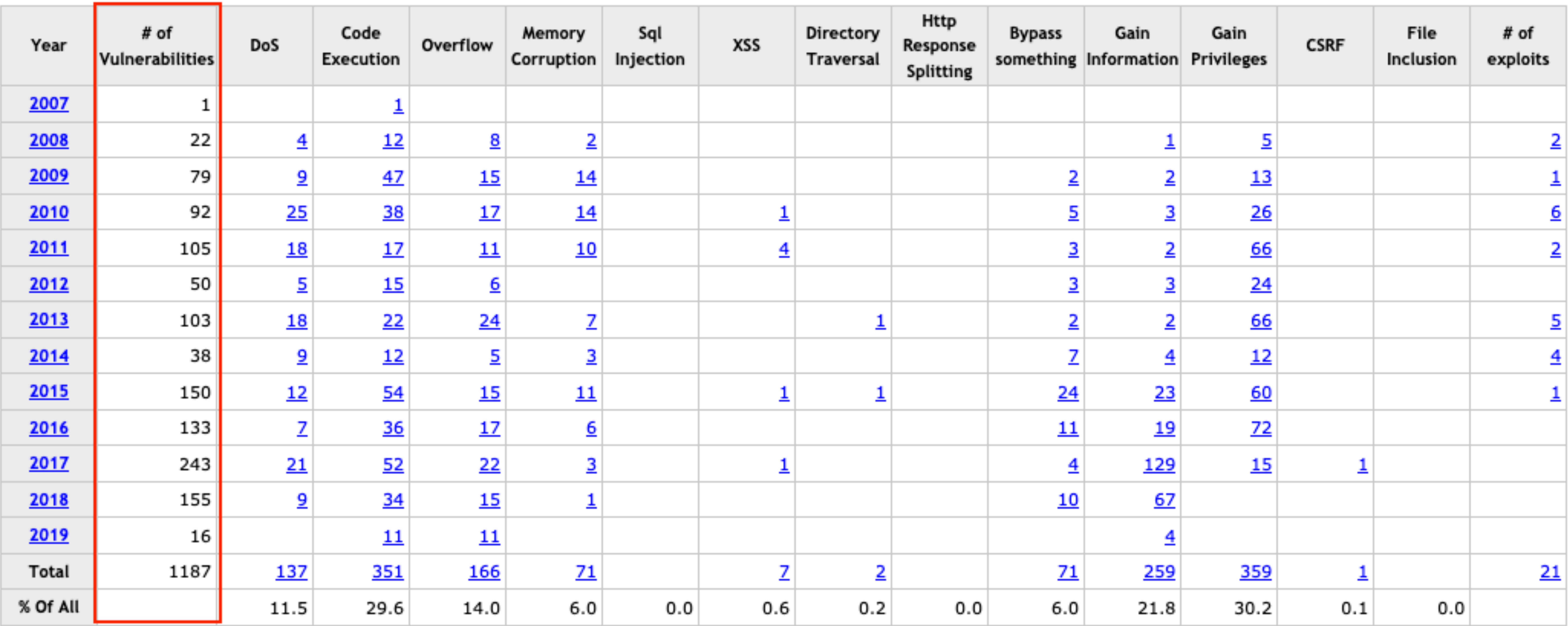
Systemy Operacyjne

Linux



Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
1999	19	7		3						1		2			
2000	5	3										1			
2001	22	6								4		3			
2002	15	3		1						1	1				
2003	19	8		2						1	3	4			
2004	51	20	5	12							5	12			
2005	133	90	19	19	1					6	5	7			
2006	90	61	5	7	7			2		5	3	3			
2007	62	41	2	8						3	8	7			
2008	71	43	3	17	4					4	6	12			
2009	102	64	2	21	6					7	11	21			5
2010	123	67	3	16	7					7	30	14			5
2011	83	62	1	21	10					1	21	9			1
2012	115	83	4	25	10					6	19	11			
2013	189	101	6	41	13					11	57	26			7
2014	132	88	8	20	9					11	30	20			10
2015	86	55	6	15	4					11	10	17			
2016	217	153	5	38	18					12	35	52			1
2017	454	147	169	52	26			1		17	89	36			
2018	170	84	3	28	8					4	17	3			
Total	2158	1186	241	346	123			3		112	350	260			29
% Of All		55.0	11.2	16.0	5.7	0.0	0.0	0.1	0.0	5.2	16.2	12.0	0.0	0.0	

Windows*



Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2007	1		1												
2008	22	4	12	8	2						1	5			2
2009	79	9	47	15	14					2	2	13			1
2010	92	25	38	17	14		1			5	3	26			6
2011	105	18	17	11	10		4			3	2	66			2
2012	50	5	15	6						3	3	24			
2013	103	18	22	24	7			1		2	2	66			5
2014	38	9	12	5	3					7	4	12			4
2015	150	12	54	15	11		1	1		24	23	60			1
2016	133	7	36	17	6					11	19	72			
2017	243	21	52	22	3		1			4	129	15	1		
2018	155	9	34	15	1					10	67				
2019	16		11	11							4				
Total	1187	137	351	166	71		7	2		71	259	359	1		21
% Of All		11.5	29.6	14.0	6.0	0.0	0.6	0.2	0.0	6.0	21.8	30.2	0.1	0.0	

* Windows Server 2008

Przykłady

- CVE-2016-5195 — DirtyCOW
- CVE-2010-0232 — KiTrap0D od Tavisa Ormandy (Google)
- CVE-2012-0217 (i młodszy brat CVE-2006-0744) — Intel SYSRET znalezione w 2012 przez Rafała Wojtczuka (InvisibleThingsLab)
- CVE-2018-8897 — POPSS/MOVSS

Przeciwdziałanie

- Implementacja polityki patchowania
- Hardening
    - Dobre praktyki (e.g. CIS Benchmarks)
    - Dodatkowe mechanizmy obronne (e.g. Linux - grsecurity, LKRG; Windows - EMET oraz reinkarnacja w postaci WDEG)

📚

- https://dirtycow.ninja/
- http://seclists.org/fulldisclosure/2010/Jan/341
- https://www.cisecurity.org/cis-benchmarks/
- https://grsecurity.net/
- http://www.openwall.com/lkrg/
- https://support.microsoft.com/en-us/help/2458544/the-enhanced-mitigation-experience-toolkit
- https://docs.microsoft.com/en-us/powershell/module/processmitigations/?view=win10-ps
- https://docs.microsoft.com/en-us/windows/security/threat-protection/windows-defender-exploit-guard/windows-defender-exploit-guard
- https://blog.xenproject.org/2012/06/13/the-intel-sysret-privilege-escalation/
- https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-8897
- http://everdox.net/popss.pdf

Hypervisory

VMware (ESXi)

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2012	12	9	6	5	1							4			
2013	9	5	3	1	2			1				2			
2014	4	3										1			
2015	2	2										1			
2016	4	1			1		1		1			2			
2017	9	1	6	5			1				1				
2018	5														
Total	45	21	15	11	4		2	1	1		1	10			
% Of All		46.7	33.3	24.4	8.9	0.0	4.4	2.2	2.2	0.0	2.2	22.2	0.0	0.0	

XEN

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2007	2														
2008	2		1	1											
2009	2	1													
2012	35	31	3	3	5						1	5			
2013	42	29	2	9	3						6	8			
2014	45	42	2	10	1						3	8			
2015	41	29	4	5	1						6	3			
2016	28	18	1	3							7	10			
2017	62	37	6	4	3						15	17			
2018	23	14	2	1						1	3	6			
Total	282	201	21	36	13					1	41	57			
% Of All		71.3	7.4	12.8	4.6	0.0	0.0	0.0	0.0	0.4	14.5	20.2	0.0	0.0	

Przykłady

- Cloudburst — *guest escape* w VMware z 2009 roku (via SVGA)
- Pwn2Own — zawody w hackowaniu aplikacji 🤓
  - 2016 — Włączenie wirtualizacji do zakresu
  - 2017 — 2 drużyny dokonały udanej ucieczki z VMware
  - 2019 — VirtualBox escape 2, VMware escape 1

Przeciwdziałanie

🤷‍♂️

📚

- https://en.wikipedia.org/wiki/Virtual_machine_escape
- https://www.blackhat.com/presentations/bh-usa-09/KORTCHINSKY/BHUSA09-Kortchinsky-Cloudburst-PAPER.pdf
    - https://vimeo.com/6595148
- https://blogs.vmware.com/security/2017/03/security-landscape-pwn2own-2017.html
- https://www.blackhat.com/docs/eu-17/materials/eu-17-Mandal-The-Great-Escapes-Of-Vmware-A-Retrospective-Case-Study-Of-Vmware-G2H-Escape-Vulnerabilities.pdf
- https://keenlab.tencent.com/en/2018/04/23/A-bunch-of-Red-Pills-VMware-Escapes/

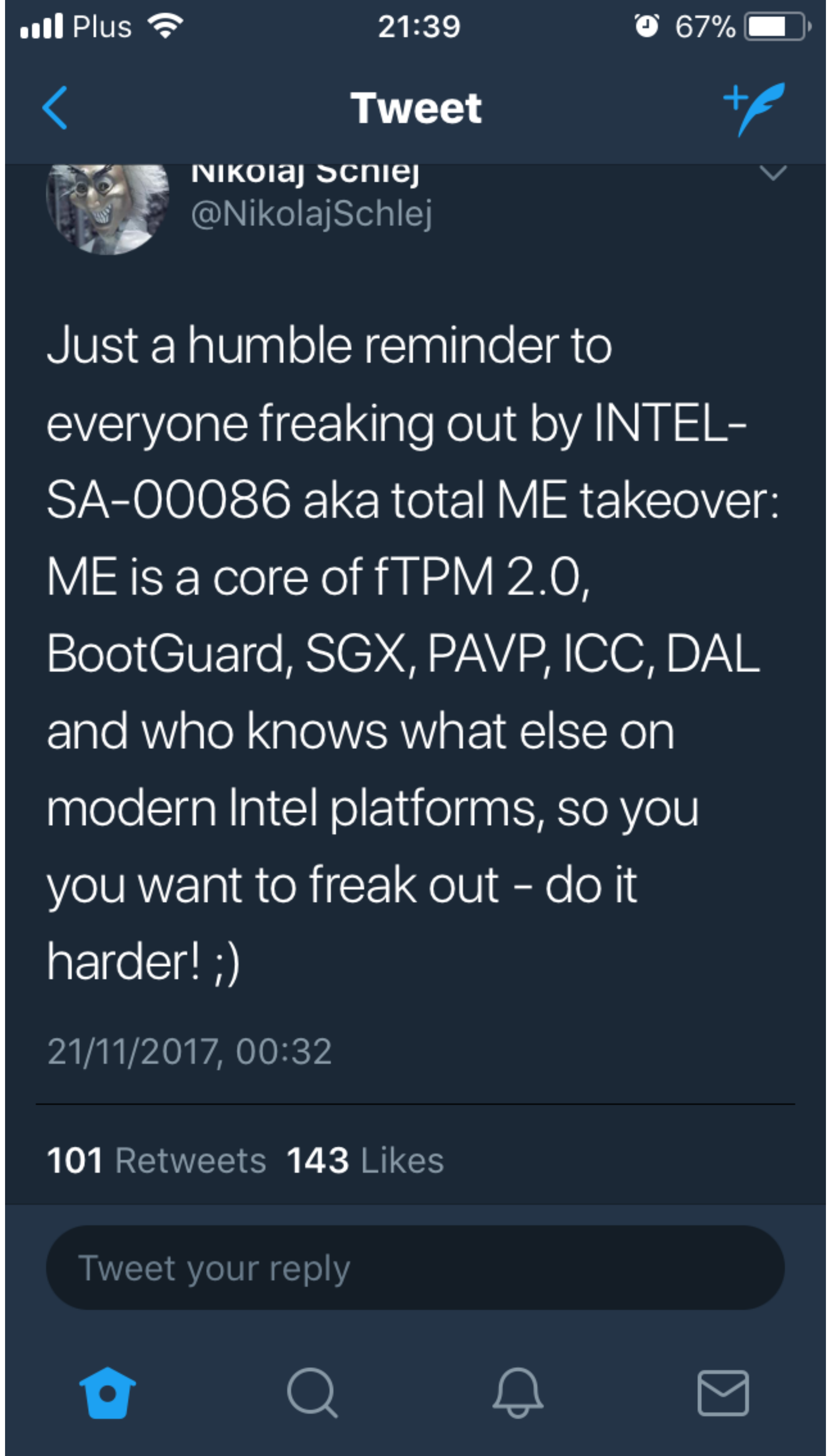
Sprzęt

Przykłady — CPU 1/2

- Bugi
  - Pentium FDIV bug — Intel - $$$ = 😢
  - AMD microcode security update — Robert Święcki podczas fuzzowania kernela na domowej stacji
  - Meltdown & Spectre — Jann Horn (Project Zero) i inni

Przykłady — CPU 2/2

- Ficzery 🤔
  - sandsifter — Fuzzing CPU na BlackHat 2017 przez Christophera Domas
  - Intel-SA-00086 — bugi w Intel Management Engine (ME)

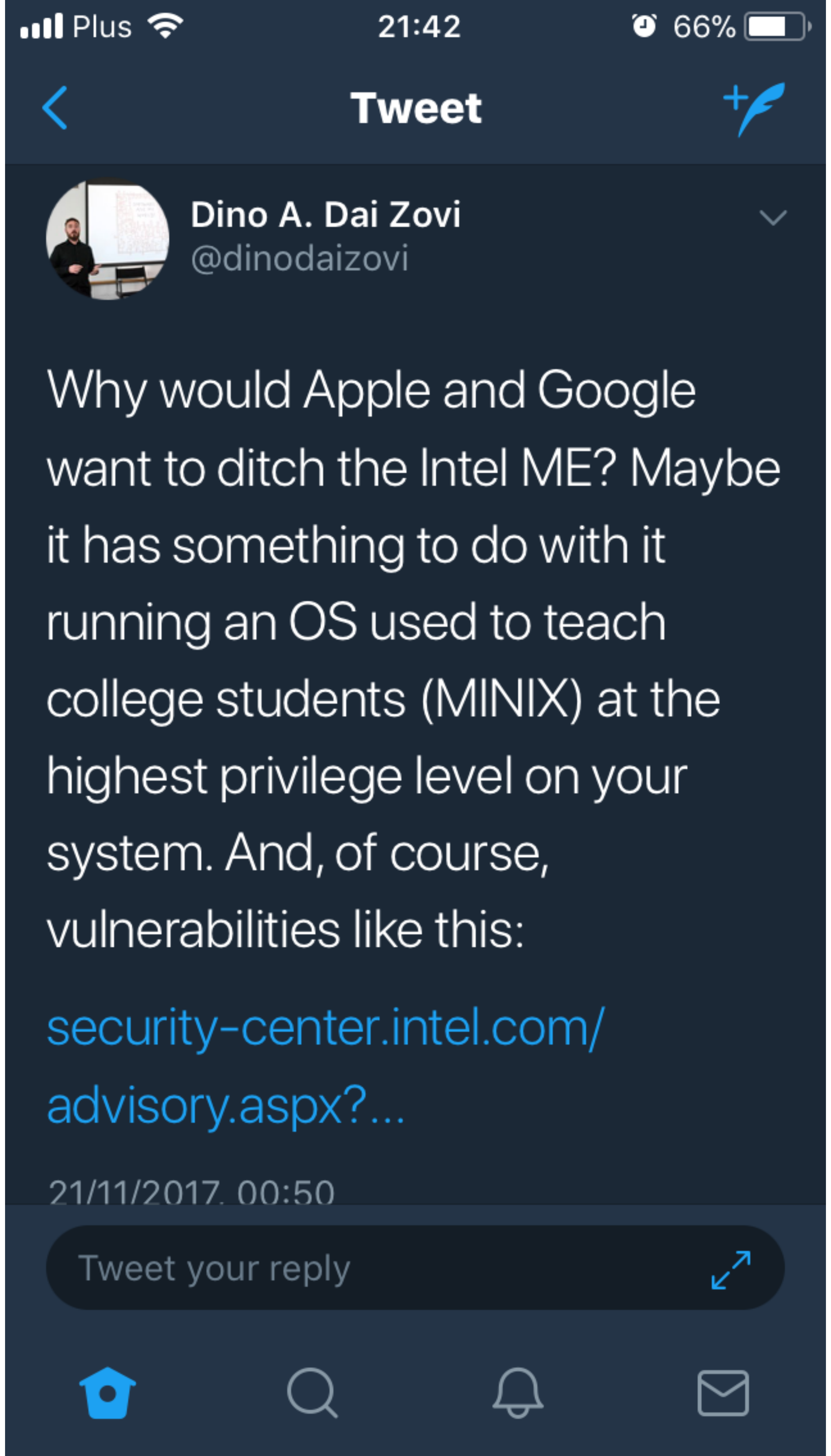


Nikolaj Schlej
@NikolajSchlej

Just a humble reminder to everyone freaking out by INTEL-SA-00086 aka total ME takeover: ME is a core of fTPM 2.0, BootGuard, SGX, PAVP, ICC, DAL and who knows what else on modern Intel platforms, so you you want to freak out - do it harder! ;)

21/11/2017, 00:32

101 Retweets 143 Likes



Dino A. Dai Zovi
@dinodaizovi

Why would Apple and Google want to ditch the Intel ME? Maybe it has something to do with it running an OS used to teach college students (MINIX) at the highest privilege level on your system. And, of course, vulnerabilities like this:

security-center.intel.com/advisory.aspx?...

21/11/2017, 00:50

Przykłady — RAM

- RowHammer — oryginalny pomysł i research Thomas Dullien et al (Project Zero); dalsze działania prowadzone przez różne grupy akademickie
  - Na początku (2015) — desktopy (lokalnie)
  - Później (2016) — urządzenia mobilne (lokalnie) oraz VM-to-VM attacks (“lokalnie”)
  - Teraz (2018) — urządzenia mobilne (zdalnie!) serwery w chmurze (zdalnie!)

Przeciwdziałanie

📚

- http://scholar.harvard.edu/files/mickens/files/theslowwinter.pdf
- https://wiki.osdev.org/CPU_Bugs
- https://danluu.com/cpu-bugs/
- https://lists.debian.org/debian-security/2016/03/msg00084.html
- https://cyber.wtf/2017/07/28/negative-result-reading-kernel-memory-from-user-mode/
- https://meltdownattack.com/
- https://www.blackhat.com/docs/us-17/thursday/us-17-Domas-Breaking-The-x86-Instruction-Set-wp.pdf
- https://github.com/xoreaxeaxeax/sandsifter
- https://www.intel.com/content/www/us/en/support/articles/000025619/software.html
- https://blog.rapid7.com/2017/11/21/intel-sa-00086-security-bulletin-for-intel-management-engine-me-and-advanced-management-technology-amt-vulnerabilities-what-you-need-to-know/
- https://www.blackhat.com/docs/eu-17/materials/eu-17-Goryachy-How-To-Hack-A-Turned-Off-Computer-Or-Running-Unsigned-Code-In-Intel-Management-Engine.pdf
- https://en.wikipedia.org/wiki/Row_hammer
- https://googleprojectzero.blogspot.com/2015/03/exploiting-dram-rowhammer-bug-to-gain.html
- https://www.vusec.net/projects/flip-feng-shui/
- https://www.vusec.net/projects/glitch/

📚📚

- https://www.cs.vu.nl/~herbertb/download/papers/throwhammer_atc18.pdf

- https://arxiv.org/abs/1805.04956

Podsumowanie

- Software jest popsuty pod każdym kątem
- Hardware jest popsuty i to dopiero wierzchołek góry lodowej
- Dobre praktyki na każdym stopniu zmniejszają ryzyko, ale nigdy go nie wyeliminują
- Bezpieczeństwo to proces, nie produkt

👋

https://dyjak.me

Twitter: @andrzejdyjak

Github: @dyjakan