Yandex.Root Final

Igor Gnatenko (Linuxorgru)

April 23, 2015

Contents

Abstract

Part I

Приготовления и запуск

2 образа. Debian, OpenIndiana. Всё как обычно, меняем пароль руту, грузимся.

Part II

Задачи

Chapter 1

Backup

Our old admin supported small site with some very useful texts. Unfortunately, he quit some time ago and we do not know how to restore the site. Please make it run again with the most recent data available.

true_admin:
Сразу смотрим /.bash_history пока его не затёрли (HISTSIZE=10!).
Оттуда понятно что бэкапы лежат в /var/backups и так же виден ключ и то что шифровалось через openssl. Попутно выясняется что ключ подходит только к первому архиву. А тааам... ключ ко второму архиву в котором... ну вы поняли.

```
HISTSIZE=10000 иии...
cd /var/backups
KFILE=backup.key
TMPFILE=/tmp/temp.arch
# самый первый ключ из .bash_history
echo 27CBB6C8170EF54C3235AF3F8ED2A106 > $KFILE

# небольшой уикл
for f in `ls *encrypted | sort -r`; do
  newname=`basename $f .encrypted`;
  openssl aes-128-cbc -d -kfile $KFILE -in $f -out $newname;
  tar --strip-components=1 -xf $newname root/backup.key
done

for f in `ls *tgz | sort` ; do tar -xf $f; done
# копируем usr/share/nginx/www в корень.
mv usr/ /
```

Chapter 2

HTTPS MITM

We continue with our Internel filtration topic. Now you need to set up HTTP proxy on port 3129 which will intercept all HTTPS traffic coming through and do it properly: we expect the proxied sites to provide valid certificates for their associated host-names. You may sign it with our own MITM certificate authority (the CA key and certificate are available in /root), we will use a client that trusts this CA.
To additionally prove that you are ready to intercept traffic, please substitute the value for Server: header of all requests with the string "root.yandex.com".

Ставим mitmproxy, запускаем и останавливаем для того чтобы сгенерировались сертификаты
потом cd /root ; cat mitm.key mitm.crt > .mitmproxy/mitmproxy-ca.pem
Затем:

```
1  def response(context, flow):
2      flow.response.headers["Server"] = ["root.yandex.com"]
```

Listing 1: add_header.py

```
mitmproxy -p 3129 -s add_header.py
```

Жмём | и запускаем game

Chapter 3

CI

We got a very old continious integration system (on OpenIndiana) and we want to upgrade it.
Please upgrade project serverMVC to use Docker.
Out checker works with your repo at ssh://root@${YOUR_IP}/root/app.git and for start deploy checker will use url:
http://${YOUR_IP}:8080/hudson/job/serverMVC_docker/build?delay=0sec
Moreover, checker expects Docker running on ${YOUR_IP}.
Our id_rsa.pub: ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDoGfF7Dvs5H0aeMsfm9MMasUWY12rdphM410FJdJ aoAjGFA8X6PeKq06qEbZpbmXRs0Yfrcwn1hRONAQ
FHjHvtfp4ViVOtfmt1byubG9BaFRJe+L3+l7MkAodYyrC93/jisk3xi/veAkuRFa4F7qUioBOuRYXE KSg4eF+tMouqbzKoM2O9vsAHrBRaIhV+yTiiDjN2UswzmQl4n4m/wRZ/OKISiewUzol
i3Lpl5IdaNMiYdsi9D8Mgb7N2x4DKZKTXOVnHmMN79yL1u2WUlp3vhWAmz8Af4Sux7jh

Chapter 4

netflow

Set up a netflow receiver on port 9996 and make a traffic billing.
For each user you should write bytes count and make it available via http://<your-ip>/billing.html which looks like:

```
<tr>
  <td>IP</td>
  <td>bytes count</td>
</tr>
```

Update period: 1 minute
Sort: by bytes DESC

Ставим flow-tools. Настраиваем. Теряем 3 часа, чтобы понять, что нам нужны только 220.123.31.0/24. Готово.

```
-w /var/flow/ -n 1339 -N 0 0/0/9996 -R /root/bin/flowrep.sh
```

Listing 2: flow-capture.conf

```
stat-report localnet
    type ip-address
    output
    format ascii
    options -header,-xheader,-totals
    fields -flows,+octets,-packets,-duration

stat-definition localnet
    report localnet
```

Listing 3: report.conf

```
#!/bin/bash
FILE="/var/www/flow/billing.html.new"
echo "<html><head></head><body><table>" > $FILE
flow-cat /var/flow/ft-* | flow-report -s /root/bin/report.conf -S localnet | \
  grep -v '\#' | sort -r -n -k 2 -t ',' | grep "220.123.31." | \
  awk -F',' '{print "<tr>\n  <td>"$1"</td>\n  <td>"$2"</td>\n</tr>"}' >> $FILE
echo "</table></body></html>" >> $FILE
mv /var/www/flow/billing.html.new /var/www/flow/billing.html
```

Listing 4: flowrep.sh

Chapter 5

Repo

We got a repositary at /root/repo, but it doesn't work with youm < 3.0.0. Fix it and make aviable via http://<ip>/repo.

Сразу смотрим в ман по createrepo:
The older default was "sha", which is actually "sha1", however explicitly using "sha1" doesn't work on older (3.0.x) versions of yum, you need to specify "sha".
На всякий случай бэкапим. С дуру попробовал пересоздать repodata с указанием алгоритма sha, сломал. Восстановил, открыл руками repomd.xml и поменял тип и хэш-суммы.

```
cd /root/repo/repodata/

for i in *.bz2; do
  sha_old=$(sha256sum $i);
  sha_new=$(shasum $i);
  sed -i -e "s/$sha_old/$sha_new/g" repomd.xml;
done

for i in *.bz2; do
  bzcat $i > ${i%.bz2};
  sha_old=$(sha256sum $i);
  sha_new=$(shasum $i);
  sed -i -e "s/$sha_old/$sha_new/g" repomd.xml;
  rm -f ${i%.bz2};
done

sed -i -e "s/\"sha256\"/\"sha\"/g" repomd.xml
```

Chapter 6

Infected binary

Your image has been touched by a cracker who replaced one of standard system binaries with his own. We thought that the program contains some secret string and it will output it when properly executed. Please find that string.

Ну это совсем просто. Ищем изменённые бинарники (проверяем чексумму), затем смотрим в него.

```
apt-get install debsums -y
debsums -a -s
strings /usr/bin/\[ | more
```

Ага. Записать строчку в binary.txt и расшарить по http. Готово.

Сделано быстрее всех

Chapter 7

DNS MITM

Let's suppose you're an administrator of large corporate network. You have a list of hosts which should be blocked according to the company policy. You decide to set up your DNS server in such a way that it acts as normal DNS server for all the hosts except the ones from this list. For those hosts, it returns the address of itself to all A queries. Later you plan to set up a special web server that displays the page about your company Internet restrictions on the same machine.
Now the task is to set up such DNS server. You should find the list of hosts in /root.

Chapter 8

SVN

You have svn repository in /root/repo. Delete (like svn rm) all files which are greater than 5MB in all revisions and make them available via svn://ip/
Big file should be deleted only in the revision when it became >5MB.

К сожалению, мега-супер-пупер-крутой скрипт был утерян ввиду анализа Infected binary через gdb, так что вкратце алгоритм:

1. Ставим более свежие версии svn и svn-utils из wheezy-backports
2. Запускаем svnserve на /root/repo
3. Делаем локальный клон через svn co
4. Дальше скрипт:
   - Дампить все ревизии через svn dump (1 - полная, остальные с –incremental)
   - Смотреть какой файл превышает 5МиБ через svn ls –verbose
   - Добавлять к списку exclude
   - Прогонять через svndumpfilter
5. Получили 101 отфильтрованную ревизию
6. Теперь через svnadmin create создаём репозиторий и через svn load грузим фильтрованный дамп
7. Сервим новый реп
8. PROFIT!

Chapter 9

NFS

Make nfs://10.10.10.11/dir available as http://<yourip>/nfs. Use this credentials user@YA.ROOT:password. NFS server's host name is localhost.

Chapter 10

Nginx Lua

We have inherited from previous admin several modules for nginx, you can find it out at /etc/nginx/lua.
No documentation, no examples of configuration files. But we have some examples, how web-server on port 8000 must work:

- /static/local/jquery.min.js should return content of file /var/www/static/jquery.min.js
- /static/local/<size>/dog.png should return thumbnail of image /var/www/static/dog.png, with width and height limit equals to X. <size> is a verbal description of size ("small", "medium" and so on, full list is unknown). We have values for X for different sizes: 50, 100, 500, 1000, 2000.
- All requests to /static/local/* except requests for css- and js- files, should return error 403 unless user has a special authrization cookie
- Request to /auth/local/jquery.min.js should set authorization cookie with name "auth_local/jquery.min.js", which is accepted by /static/local/jquery.min.js

This list is not complete! But it's all we have.
Don't forget to setup caching. It is said these modules support them.

```
#!/usr/bin/env python3
# ACHTUNG! THIS IS NOT HOW I USUALLY WRITE SCRIPTS!

# This supposed to be code in nginx+lua, but I found
# python more convenient for handling tricky URLs and
# image manipulations. Sorry :)

from bottle import route, get, run, template, \
  static_file, request, response, HTTPError

import Image
import time
import re

SIZES = dict(xsmall=50,  small=100, medium=500, large=1000, xlarge=2000)

@get('<uri:path>')
def static(uri):
  # HERE IS WHERE YANDEX EXPETS CACHE TO BE
  if uri.find('/static/remote/') != -1:
    # quick and dirty way to parse URL, do not do in so in prod, sons
    toks = uri.split('/')
    size = toks[3]
    fname = toks[-1]
    if fname.endswith('png'):
      fname = 'dog.png'
    else:
      fname = 'turtle.jpg'
    uri = '/static/local/%s/%s' %(size, fname)
    return static_file(uri, root='.')

  # AUTH REQUEST, LET'S SET COOKIE
  elif uri.startswith('/auth'):
    fname = uri.split('/')[2:]
    fname = "/".join(fname)
    # bits and pieces from reverse engineered lua code
    response.set_header('Set-Cookie', '%s=%s; path=/; expires=%s' % ('auth_' + fname, 'xxx', int(time.time())+24*3600 ))
    return ""  # doesn't really matter what to return

  # STATIC FILES!
  elif uri.endswith('.css') or uri.endswith('.js'):
    if uri == "/auth/local/jquery.min.js":
      response.set_cookie("auth_local", "jquery.min.js")
      response.set_header('Set-Cookie', 'auth_local/jquery.min.js')
      uri  = "/static/local/jquery.min.js"
    # static_files() screws headers, so I use open() :(((
    return open('.'+ uri, 'rb')  # DO NOT DO THIS IN PROD!!1

  # IMAGES
  elif uri.endswith(".png") or uri.endswith('.jpg'):
    # check all cookies if there is auth cookie....
    for c in request.cookies:
      if c.startswith('auth_'):
        # oh no, what a dirty code...
        uri = '/static/' + c.split('auth_')[1]
        break
    else:
      # no auth cookie, get out!
      raise HTTPError(403)

    # got auth cookie, let's resize img and put in on disk
    r = re.match('/static/local/([a-zA-Z]+)/(.+)', uri)
    vsize, fname = r.groups()
    img = Image.open('static/local/' + fname)
    size = SIZES[vsize]
    img = img.resize((size,size))
    img.save('.'+uri)
    return static_file(uri, root='.')

  # all other URLs are forbidden (doesn't really matter)
  raise HTTPError(404)


run(host='0.0.0.0', port=8000, debug=True, reloader=True)
```

Listing 5: nginx_lua.py

Part III

Итоги

6 место. Огромное спасибо tazhate, madrouter, realloc, lumi, octocat, delirium, anonymous_sama, trofk, feofan, tailgunner, всем кого забыл, ну и мне %)

Backup	HTTPS MITM	CI	netflow	Repo	Infected binary	DNS MITM	SVN	NFS	Nginx Lua
01:29:18	04:27:47	07:01:54	05:25:27	01:39:15	00:18:49	01:27:47	03:29:41	04:52:19	07:06:34