# Idea

> ### Add meta-shell commands
>
> Shell sessions typically expose a direct connection to a remote shell, but are lacking a number of nice features such as the ability to stop a remote command, background a command (this could be advanced or depend on the underlying session), or to even lock the session. This project would implement some pre-processing hooks to shell sessions so that job control could be added by default (allowing backgrounding of commands), meta-commands like 'background' and 'sessions' could be added as well.

# Personal Information


- Surname: Wang
- Name: Yihang
- University: Harbin Institute of Technology
- Region: China
- Email: wangyihanger@gmail.com
- GitHub: https://github.com/WangYihang
- Freenode nickname: WangYihang


# Skills

- **Programming Language**
  - (★★★★☆) C
  - (★★★★☆) Python
  - (★★★☆☆) PHP
- **Vulnerability Mining**
  - Web Applications
- **Open Source Project of my own:**
  - Webshell-Sniper

    ```
    A webshell manager via terminal.
    Demo video: https://www.youtube.com/watch?v=iAUwb8SSS4s
    ```

  - Reverse-Shell-Manager

    ```
    A reverse shell manager via terminal. This tool is familiar to Meterpreter, It
    has a lot of build-in commands like 'shell' to pop up a interactive shell.
    Demo video: https://asciinema.org/a/143640
    ```

  - Exploit-Framework

```
A exploit framework pay tribute to Metasploit.
Demo video: https://asciinema.org/a/152418
```

- **Open Source Project contributed to:**
  - Codiad

    ```
    Codiad is an open source, web-based, cloud IDE and code editor with minimal
    footprint and requirements.
    Help to found 2 RCE vulnerabilities:
    ['CVE-2017-11366', 'CVE-2017-15689']
    Help to found 1 Any file read/write vulnerbility:
    [CVE-2017-15690]
    ```

  - GitBook

    ```
    Help to found 1 any file read vulnerability:
    [CVE-2017-15688]
    ```

# Plan

A normal shell session created by reverse_shell payload compared to meterpreter, which has all of the commands like `background`, `sessions`, `resource`

Secondary bits around handling 'Shell' Job control (Ctrl+C and Ctrl+Z) more properly in Metasploit could also be added to shell sessions, but the primary goal of the 'meta shell' project isn't necessarily to fix that.

The final capability that should be added to the meta shell would be the ability to 'upgrade' the shell session so it has a PTY, encryption

1. Implement the build-in command: `sessions`, `background`, `resource`
2. Handle signals in the reverse shell, such as: `CTRL+C`, `CTRL+Z`...
3. Upgrade the `shell session` to a `terminal session(PTY)`
4. Implement the `download` and the `upload` Build-in commands.

## Files to change:

```
plugins/session_tagger.rb
lib/rex/proto/ipmi/open_session_reply.rb
lib/msf/scripts/meterpreter.rb
lib/msf/core/handler/reverse_tcp_ssl.rb
lib/msf/core/handler/reverse_tcp_allports.rb
lib/msf/core/db_manager/session.rb
lib/msf/core/db_manager/http/servlet/session_event_servlet.rb
lib/metasploit/framework/data_service/stubs/session_data_service.rb
lib/metasploit/framework/data_service/remote/http/remote_session_event_data_service.rb
...
```

## Milestone:

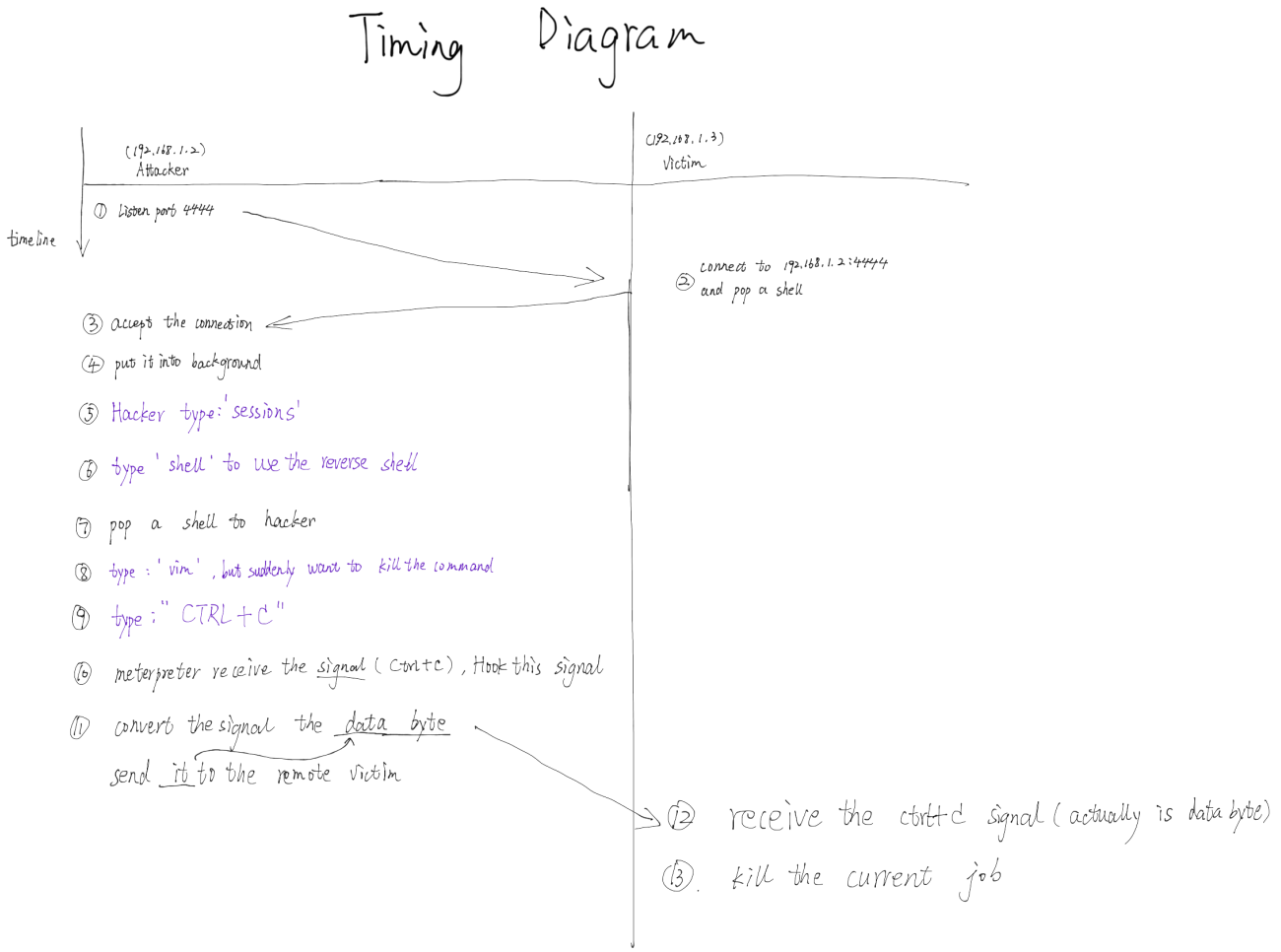
| Target | Start Time | End Time | Weeks | Hardness |
| --- | --- | --- | --- | --- |
| Build development and Read source code | 2018/04/17 | 2018/05/01 | 2 | ★★☆☆☆ |
| Try to implement an tiny reverse shell signal handler based on Reverse-Shell-Manager, Upgrade it to TTY | 2018/05/01 | 2018/05/15 | 2 | ★★★☆☆ |
| Implement of 'background' command | 2018/05/15 | 2018/05/22 | 1 | ★★★★☆ |
| Implement of 'sessions' and 'resource' command | 2018/05/22 | 2018/06/12 | 3 | ★★★★☆ |
| Implement of Signal Catcher and Sender | 2018/06/12 | 2018/06/26 | 3 | ★★★★★ |
| implement of 'ls' command like meterpreter does | 2018/06/26 | 2018/07/10 | 2 | ★★☆☆☆ |
| implement download/upload functions | 2018/07/10 | 2018/07/17 | 1 | ★★☆☆☆ |
| implement of crypto method and upgrade session to a complete metepreter session | 2018/07/17 | 2018/07/24 | 1 | ★★★★☆ |
| Upgrade documentation and close-out | 2018/07/24 | 2018/08/6 | 1 | ★☆☆☆☆ |

### Check points:

1. Implement of 'background' command
2. Implement of 'sessions' command
3. Implement of 'resource' command
4. Catch the mainly signals (CTRL+C, CTRL+Z) and send them to remote server
5. support PTY
6. support crypto methods like metepreter
7. implement of download and upload functions
8. implement of 'ls' command like meterpreter instead of 'ls' in the target system

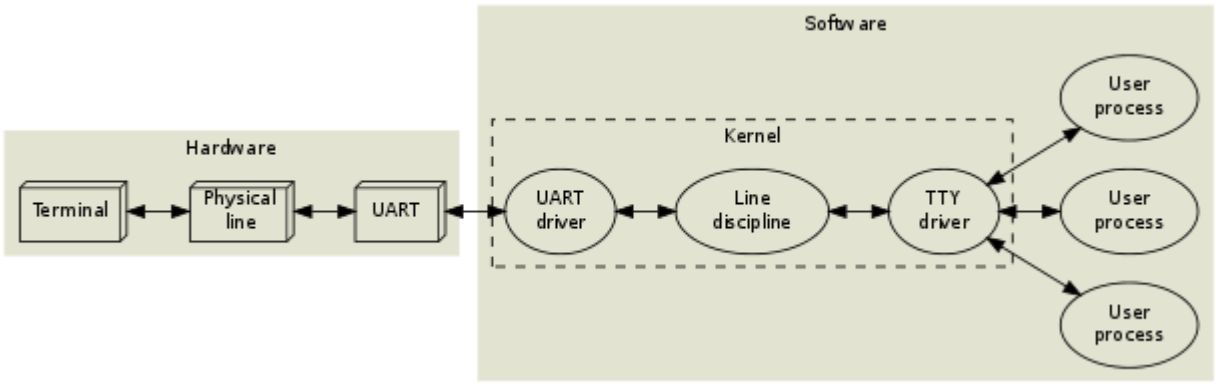
## Rough Sketch:

## Signal handler:

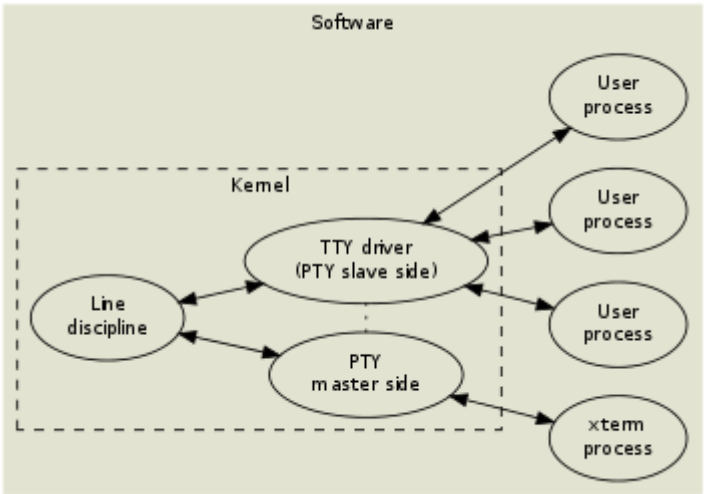



The picture above show how a program catch the signal and handle it then send it to the remote device. The signals in socket will be the control characters which is defined in the accii table. What we should do is send the correct control characters to the remote devices.

| Binary | Oct | Dec | Hex | Abbreviation | | | [b] | [c] | [d] | Name ('67) |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | '63 | '65 | '67 | | | | |
| 000 0000 | 000 | 0 | 00 | NULL | NUL | | NUL | ^@ | \0 | Null |
| 000 0001 | 001 | 1 | 01 | SOM | SOH | | SOH | ^A | | Start of Heading |
| 000 0010 | 002 | 2 | 02 | EOA | STX | | STX | ^B | | Start of Text |
| 000 0011 | 003 | 3 | 03 | EOM | ETX | | ETX | ^C | | End of Text |
| 000 0100 | 004 | 4 | 04 | EOT | | | EOT | ^D | | End of Transmission |
| 000 0101 | 005 | 5 | 05 | WRU | ENQ | | ENQ | ^E | | Enquiry |
| 000 0110 | 006 | 6 | 06 | RU | ACK | | ACK | ^F | | Acknowledgement |
| 000 0111 | 007 | 7 | 07 | BELL | BEL | | BEL | ^G | \a | Bell |
| 000 1000 | 010 | 8 | 08 | FE0 | BS | | BS | ^H | \b | Backspace[c][f] |
| 000 1001 | 011 | 9 | 09 | HT/SK | HT | | HT | ^I | \t | Horizontal Tab[g] |
| 000 1010 | 012 | 10 | 0A | LF | | | LF | ^J | \n | Line Feed |
| 000 1011 | 013 | 11 | 0B | VTAB | VT | | VT | ^K | \v | Vertical Tab |
| 000 1100 | 014 | 12 | 0C | FF | | | FF | ^L | \f | Form Feed |
| 000 1101 | 015 | 13 | 0D | CR | | | CR | ^M | \r | Carriage Return[h] |
| 000 1110 | 016 | 14 | 0E | SO | | | SO | ^N | | Shift Out |
| 000 1111 | 017 | 15 | 0F | SI | | | SI | ^O | | Shift In |
| 001 0000 | 020 | 16 | 10 | DC0 | DLE | | DLE | ^P | | Data Link Escape |
| 001 0001 | 021 | 17 | 11 | DC1 | | | DC1 | ^Q | | Device Control 1 (often XON) |
| 001 0010 | 022 | 18 | 12 | DC2 | | | DC2 | ^R | | Device Control 2 |
| 001 0011 | 023 | 19 | 13 | DC3 | | | DC3 | ^S | | Device Control 3 (often XOFF) |
| 001 0100 | 024 | 20 | 14 | DC4 | | | DC4 | ^T | | Device Control 4 |
| 001 0101 | 025 | 21 | 15 | ERR | NAK | | NAK | ^U | | Negative Acknowledgement |
| 001 0110 | 026 | 22 | 16 | SYNC | SYN | | SYN | ^V | | Synchronous Idle |
| 001 0111 | 027 | 23 | 17 | LEM | ETB | | ETB | ^W | | End of Transmission Block |
| 001 1000 | 030 | 24 | 18 | S0 | CAN | | CAN | ^X | | Cancel |
| 001 1001 | 031 | 25 | 19 | S1 | EM | | EM | ^Y | | End of Medium |
| 001 1010 | 032 | 26 | 1A | S2 | SS | SUB | SUB | ^Z | | Substitute |
| 001 1011 | 033 | 27 | 1B | S3 | ESC | | ESC | ^[ | \e[i] | Escape[j] |
| 001 1100 | 034 | 28 | 1C | S4 | FS | | FS | ^\ | | File Separator |
| 001 1101 | 035 | 29 | 1D | S5 | GS | | GS | ^] | | Group Separator |
| 001 1110 | 036 | 30 | 1E | S6 | RS | | RS | ^^[k] | | Record Separator |
| 001 1111 | 037 | 31 | 1F | S7 | US | | US | ^_ | | Unit Separator |




```
Metasploit shell just like a terminal emulator, catch the useful signal and justify
whether it should be sent to the remote device.
```




# Example

## Reverse-Shell-Manager:

```
https://asciinema.org/a/143640
```

```
def interactive_shell(self):
    self.interactive = True
    t = threading.Thread(target=transfer, args=(self.node_hash, ))
    t.start()
    try:
        while True:
            command = raw_input()
            if command == "exit":
                self.interactive = False
                self.socket_fd.send("\n")
                break
            self.socket_fd.send(command + "\n")
    except:
```

```
            self.remove_node()
        self.interactive = False
        time.sleep(0.125)
```

```
def signal_handler(ignum, frame):
    print ""
    Log.info("Enter : 'q|quit|exit' to shutdown the program!")

def main():
    signal.signal(signal.SIGINT, signal_handler)
    signal.signal(signal.SIGTERM, signal_handler)
```

### Signal handling in Ruby:

```
pid = fork do
  Signal.trap("USR1") do
    $debug = !$debug
    puts "Debug now: #$debug"
  end
  Signal.trap("TERM") do
    puts "Terminating..."
    shutdown()
  end
  # . . . do some work . . .
end

Process.detach(pid)

# Controlling program:
Process.kill("USR1", pid)
# ...
Process.kill("USR1", pid)
# ...
Process.kill("TERM", pid)
```

## Self-Asking-Questions

> **Why did this project appeal to you?**

```
I want to be a cyber security research, and metasploit is the most excellent framework
in this field, and I use it frequently, I think that is excited to contribute to the
tools I love.
```

> **How will you benefit from it?**

```
Make friends with excellent guys(Hackers)
Knowledge of Terminal
Knowledge of Ruby
Knowledge of 'How to build a high scale framework'

Knowledge of Socket
```

> **Why are you particularly suited to work on this?**

```
I have written some tools like Webshell-Sniper and Reverse-Shell-Manager, these tools
used the tech of socket and signal handle, so I think maybe I could do something to
improve it.
```

> **What will you do once the project is "finished"?**

```
1. Focus on the functions that I implemented, and once someone report a bug or mistaken,
disscuss and try to fix it.
2. Pay attention to the company 'Rapid7' and 'Metasploit'
```

> **Of the skills that you will need to complete the project, which do you already have?**

```
C programming language
Socket programming in C
Python programming language
```

> **What will you need to lean?**

```
Basic syntax of Ruby
Signal handle of Ruby
```

# Reference:


> https://github.com/rapid7/metasploit-framework/issues/9714
>
> https://blog.stalkr.net/2015/12/from-remote-shell-to-remote-terminal.html
>
> http://comp.unix.programmer.narkive.com/ow31N0bX/sending-ascii-eot-signal-over-tcp-ip-sockets
>
> http://thor-sec.com/cheatsheet/oscp/tty_spawnage/ https://blog.ropnop.com/upgrading-simple-shells-to-fully-interactive-ttys/
>
> https://github.com/rapid7/metasploit-framework/issues/9630
>
> https://www.wikiwand.com/en/ASCII
>
> http://ruby-doc.org/core-2.2.0/Signal.html