The Bro Debugger

Vlad Grigorescu
NCSA

> whoami

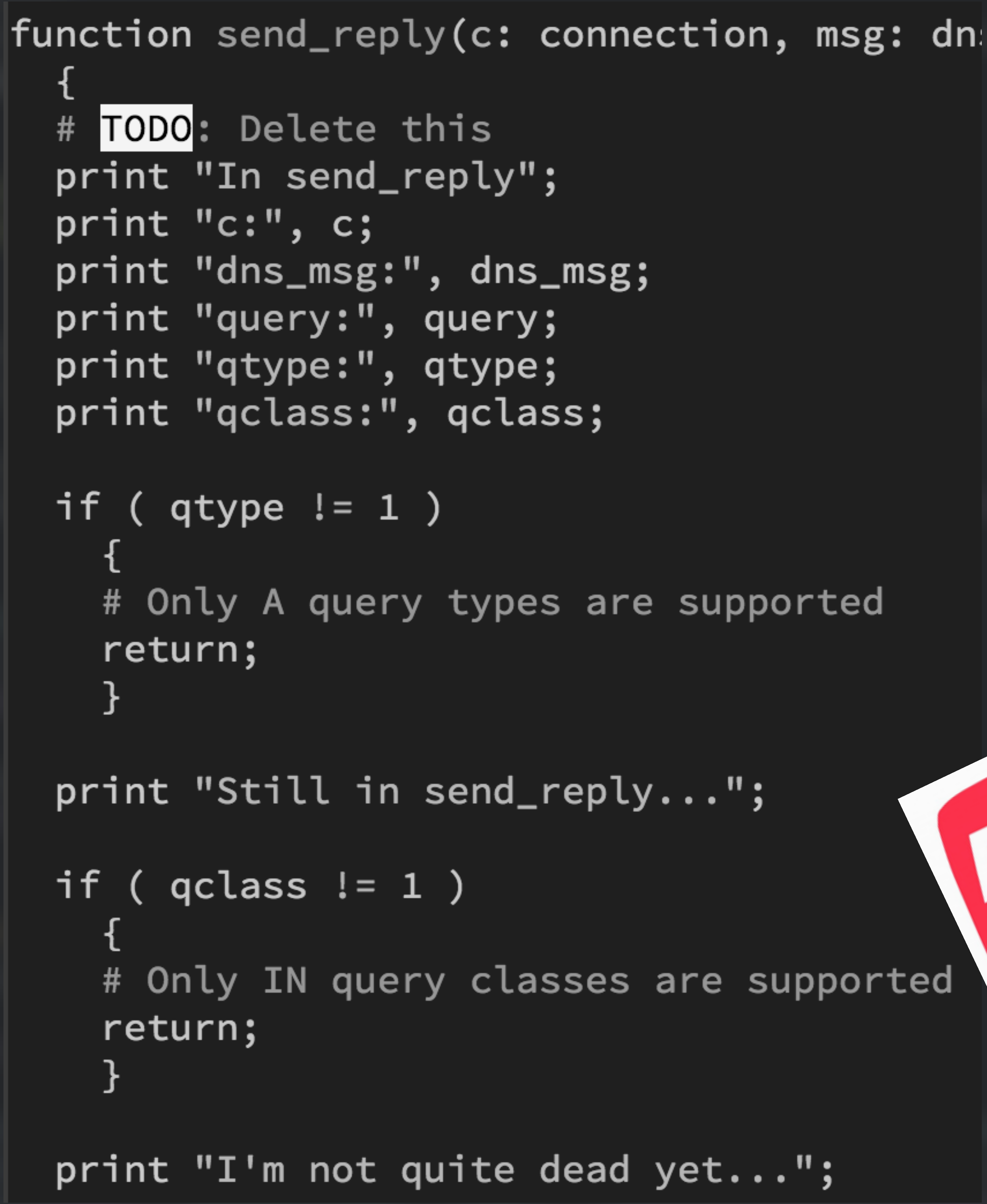
Member of the Bro development team

Security Engineer at the National Center for Supercomputing Applications (NCSA)

https://github.com/grigorescu

@0f010d

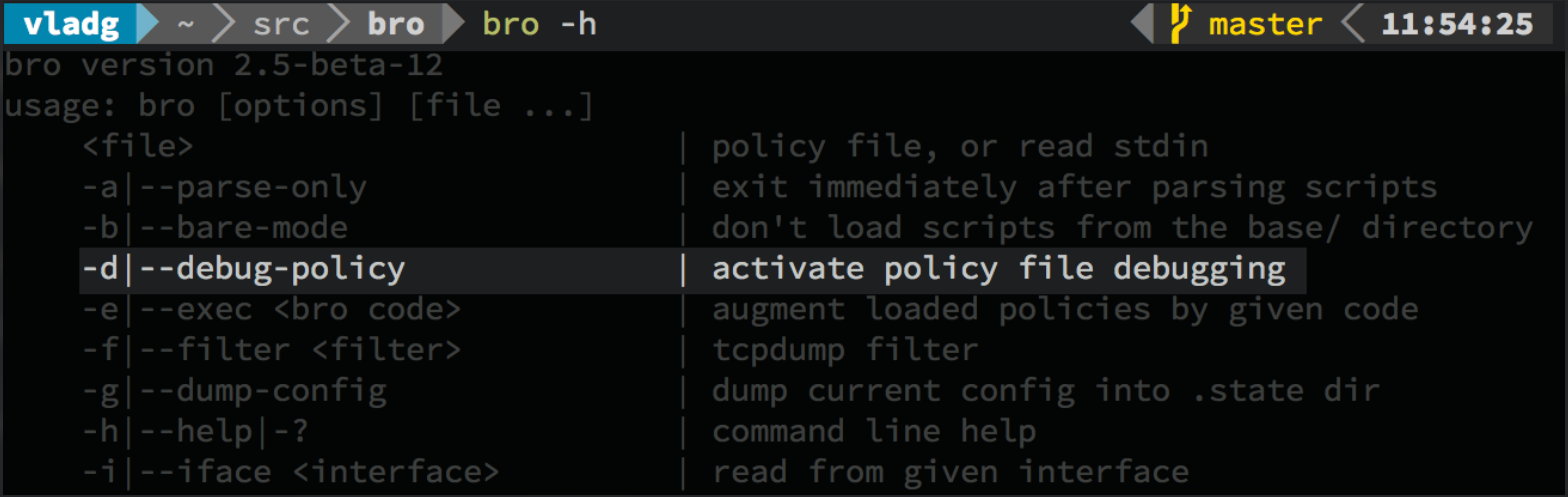
"Debugging" - originally published 1/14/2006
"Piled Higher and Deeper" by Jorge Cham
www.phdcomics.com



```
function send_reply(c: connection, msg: dn
  {
  # TODO: Delete this
  print "In send_reply";
  print "c:", c;
  print "dns_msg:", dns_msg;
  print "query:", query;
  print "qtype:", qtype;
  print "qclass:", qclass;

  if ( qtype != 1 )
    {
    # Only A query types are supported
    return;
    }

  print "Still in send_reply...";

  if ( qclass != 1 )
    {
    # Only IN query classes are supported
    return;
    }

  print "I'm not quite dead yet...";
```

There's a better way...

bro --debug-policy



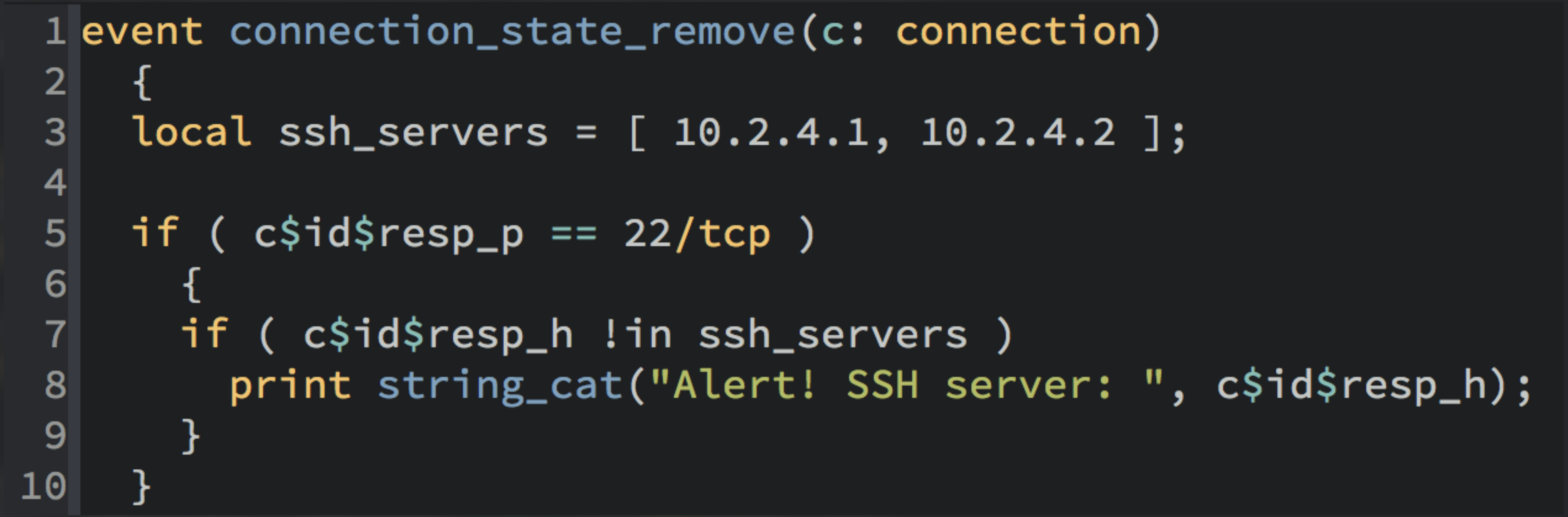
```
vladg  ~  src  bro  bro -h                                    master  11:54:25
bro version 2.5-beta-12
usage: bro [options] [file ...]
    <file>                          | policy file, or read stdin
    -a|--parse-only                 | exit immediately after parsing scripts
    -b|--bare-mode                  | don't load scripts from the base/ directory
    -d|--debug-policy               | activate policy file debugging
    -e|--exec <bro code>            | augment loaded policies by given code
    -f|--filter <filter>            | tcpdump filter
    -g|--dump-config                | dump current config into .state dir
    -h|--help|-?                    | command line help
    -i|--iface <interface>          | read from given interface
```

"GDB for Bro Scripts"

- Debugger for script-land
- No visibility into the "core layer" (C/C++ code)
- Breakpoints, flow control, examining values
- Executing Bro statements
- Can even be used on live traffic (not recommended)

Breakpoints

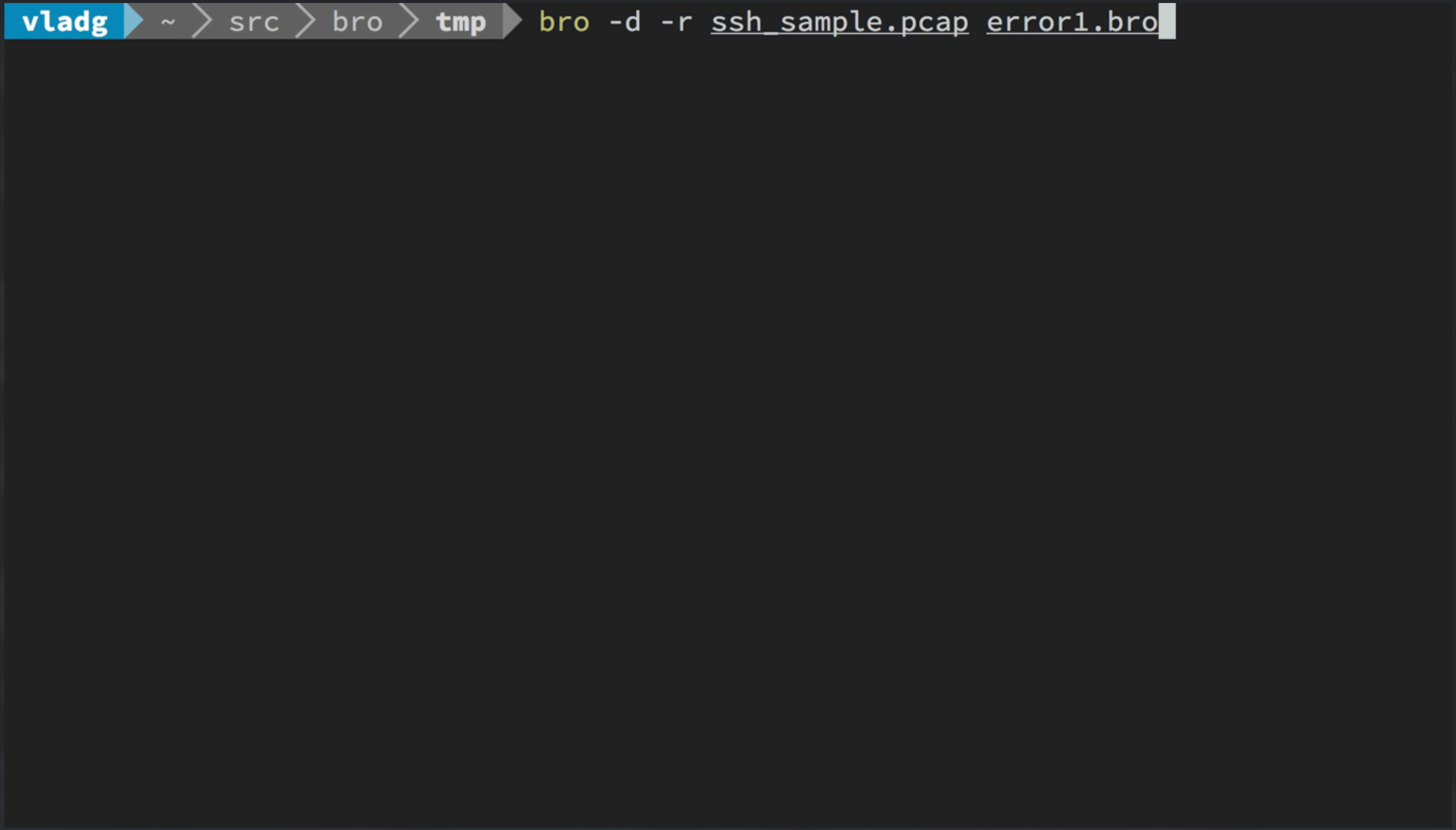
- Set breakpoints at script locations



```
event connection_state_remove(c: connection)
  {
  local ssh_servers = [ 10.2.4.1, 10.2.4.2 ];

  if ( c$id$resp_p == 22/tcp )
    {
    if ( c$id$resp_h !in ssh_servers )
      print string_cat("Alert! SSH server: ", c$id$resp_h);
    }
  }
```

```
vladg  ~ > src > bro > tmp > bro -r ssh_sample.pcap error1.bro
1445023194.281899 fatal error in <no location>: Val::CONST_ACCESSOR (addr/string) (141.142.112.248)
```

Breakpoints



```
vladg ~ src bro tmp bro -d -r ssh_sample.pcap error1.bro
```

Breakpoints

Command	Breakpoint at:
break	Current location
break 3	Line 3 of current file
break error1.bro:3	Line 3 of error1.bro
break bro_init	bro_init function/event
break irc_*	irc_* function/events

Breakpoints

Command	Description
info breakpoints	Show list of breakpoints
enable 1	Enable breakpoint #1
disable 1	Disable breakpoint #1
delete 1	Delete breakpoint #1
continue (c)	Resume execution
C-c	Stop execution

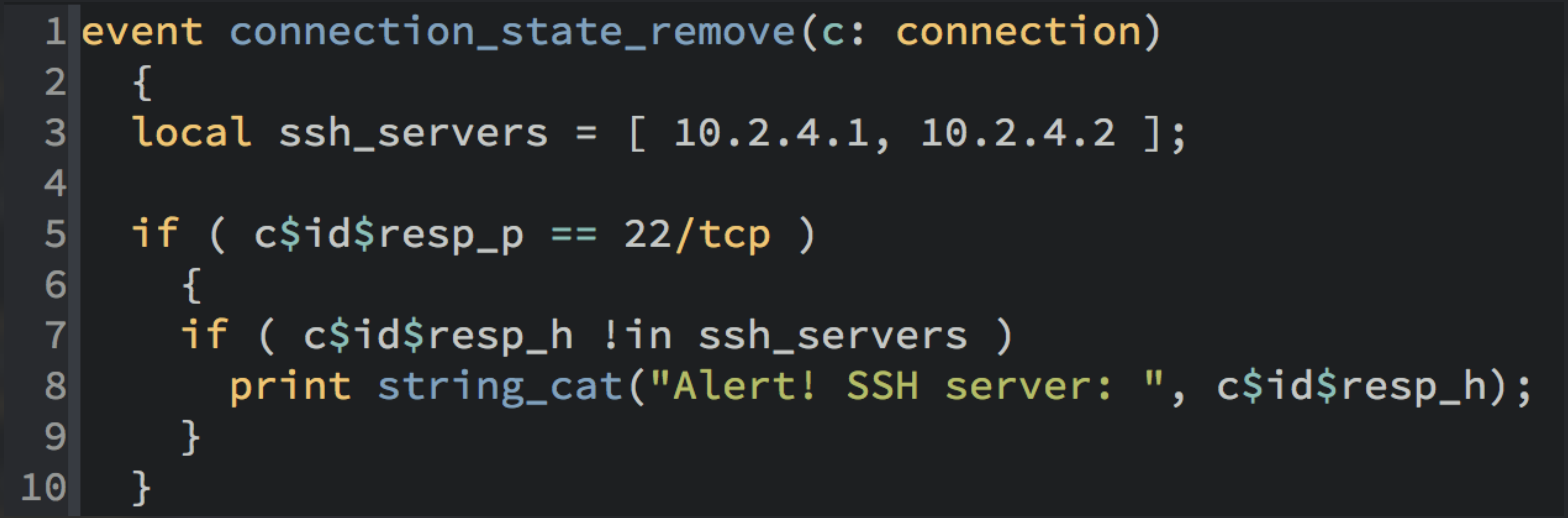
Examining State

```
(Bro [6])
```

Examining State

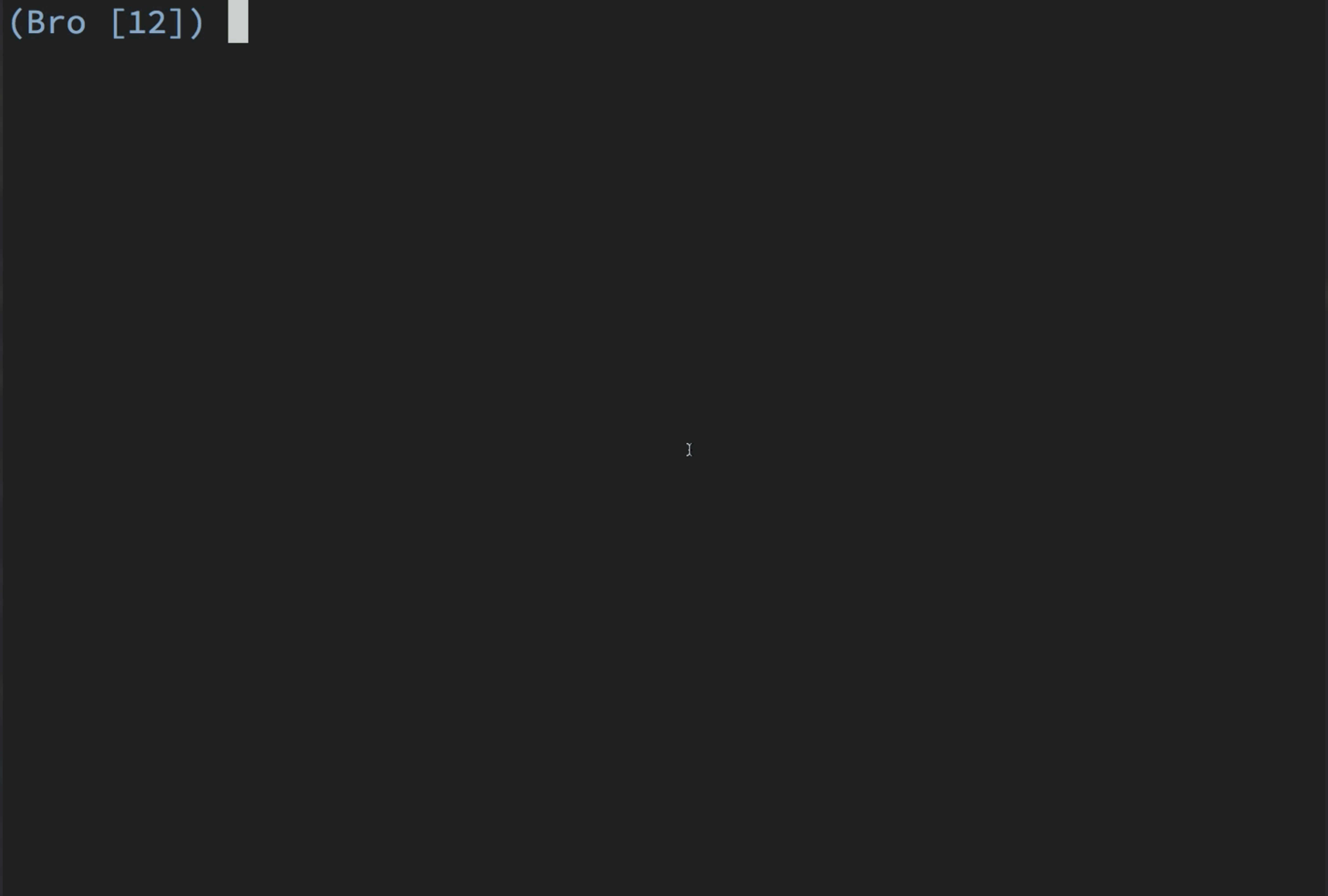
Command	Description
list	Show up to 10 lines of code
list 3	Show ±5 lines around line 3
list error1.bro:3	...around error1.bro:3
list bro_init	...around the bro_init event
print $exp (p)	Evaluate and print $exp

Flow Control



```
event connection_state_remove(c: connection)
  {
  local ssh_servers = [ 10.2.4.1, 10.2.4.2 ];

  if ( c$id$resp_p == 22/tcp )
    {
    if ( c$id$resp_h !in ssh_servers )
      print string_cat("Alert! SSH server: ", c$id$resp_h);
    }
  }
```

```
vladg ~ src bro tmp bro -r ssh_sample.pcap error1.bro
1445023194.281899 fatal error in <no location>: Val::CONST_ACCESSOR (addr/string) (141.142.112.248)
```



```
(Bro [12])
```

Flow Control

Command	Description
cond 1 c$?id	Add condition to breakpoint 1
next (n)	Next line, don't enter funcs
step	Next line, do enter funcs
finish	Run until end of current func

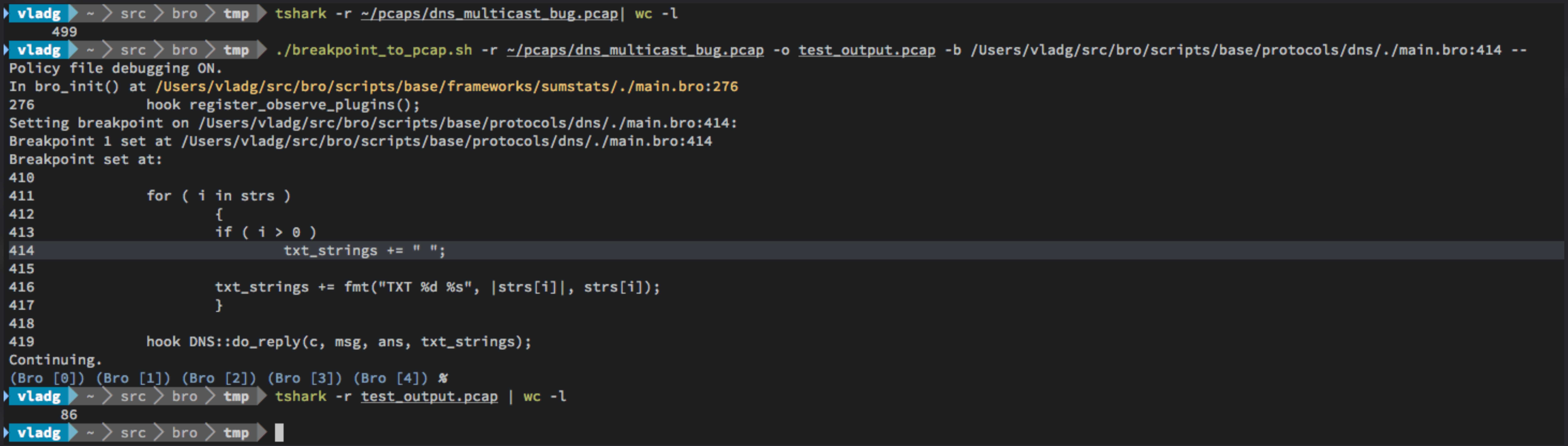
Extra Credit

- Setting condition breakpoints can be very powerful
- syslog(string)
- system(command)
- dump_current_packet(file_name)

breakpoint_to_pcap.sh

- Can filter a PCAP file
- Filters all connections that hit a certain point in the code
- Can pinpoint traffic that causes protocol errors, weirds, crashes, etc.

breakpoint_to_pcap.sh



```
vladg ~ src bro tmp tshark -r ~/pcaps/dns_multicast_bug.pcap| wc -l
     499
vladg ~ src bro tmp ./breakpoint_to_pcap.sh -r ~/pcaps/dns_multicast_bug.pcap -o test_output.pcap -b /Users/vladg/src/bro/scripts/base/protocols/dns/./main.bro:414 --
Policy file debugging ON.
In bro_init() at /Users/vladg/src/bro/scripts/base/frameworks/sumstats/./main.bro:276
276             hook register_observe_plugins();
Setting breakpoint on /Users/vladg/src/bro/scripts/base/protocols/dns/./main.bro:414:
Breakpoint 1 set at /Users/vladg/src/bro/scripts/base/protocols/dns/./main.bro:414
Breakpoint set at:
410
411             for ( i in strs )
412                     {
413                     if ( i > 0 )
414                             txt_strings += " ";
415
416                     txt_strings += fmt("TXT %d %s", |strs[i]|, strs[i]);
417                     }
418
419             hook DNS::do_reply(c, msg, ans, txt_strings);
Continuing.
(Bro [0]) (Bro [1]) (Bro [2]) (Bro [3]) (Bro [4]) %
vladg ~ src bro tmp tshark -r test_output.pcap | wc -l
      86
vladg ~ src bro tmp
```

http://go.ncsa.illinois.edu/breakpoint_to_pcap