Party Time

A competitive game of subtle espionage at an exclusive cocktail party

About Me

Joey Geralnik Bar

- Cybersecurity
- CTFs @ pasten
- Independent Software Consultant @ Hypr

Prehistoric Times

- Before the internet

Prehistoric Times

- Before the internet
- 2014


Practice Spy mode: you vs nobody
Any 7 of 8 Missions
Bug Ambassador
Contact Double Agent
Transfer Microfilm
Swap Statue
Inspect 3 Statues
Seduce Target
Purloin Guest List
Fingerprint Ambassador
VERY EARLY WORK IN PROGRESS!!!
3:02
Ambassador
Double Agent
Toby the Waiter
Security Agent Damon
Fake Signal Double Agent
Begin Talking (1/2)
chat: /help, Enter/y/t, i, PageUp, Esc


Practice Sniper mode: you vs nobody
Complete 3 Known Missions
Bug Ambassador
Contact Double Agent
Seduce Target
VERY EARLY WORK IN PROGRESS!!!
2:30

Building a private server

- Want to play the game without an internet connection
- First step?

Wireshark

- Authentication handled over kerberos
- Lobby
- P2P

Kerberos

- Patching it out
- Creating our own kerberos server

Patching

- Patching it out is not easy
  - Developer abused kerberos in creative ways

Setting up Kerberos Server

- Success!

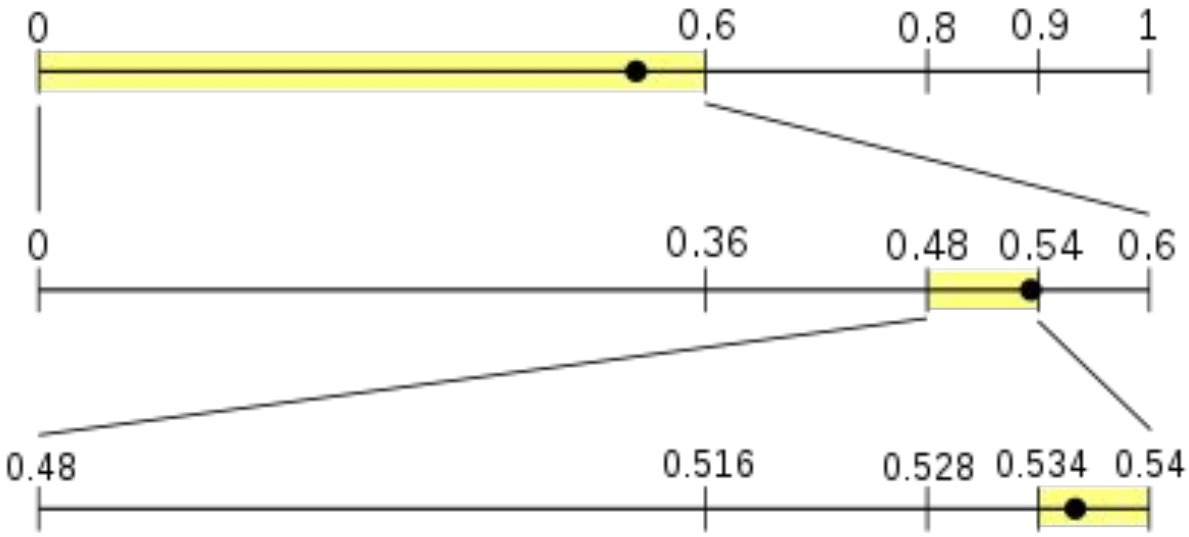
Lobby Packets

- Packets are not in a known format - looks compressed or encrypted
- Kerberos only handles Authentication, not encryption
- Find code in IDA that handles packets

- Identify Objects that compress/decompress
- Assertions contain the string “Arithmetic Coding”

```
if ( v8 > 32 )
  j_itk_generic_critical_func(L"bits <= 32", L"..\\misc\\arithmetic_coder.cpp", 437);
```

Arithmetic Coding



Where is the table (IDA)???

- FUCK IT, there’s no table
- Maybe it’s not Arithmetic Coding?
- Let’s just blindly implement it in python

It works!

But what about the table?

- There is none…

But what about the table?

- There is none…
- **This “compression” scheme actually expands the data**

Decoding packets

- Scrambled log
  - WinDBG script that hooks logs before scrambling and prints them
  - Also print data parsed from packets
- Start understanding the data structures being sent

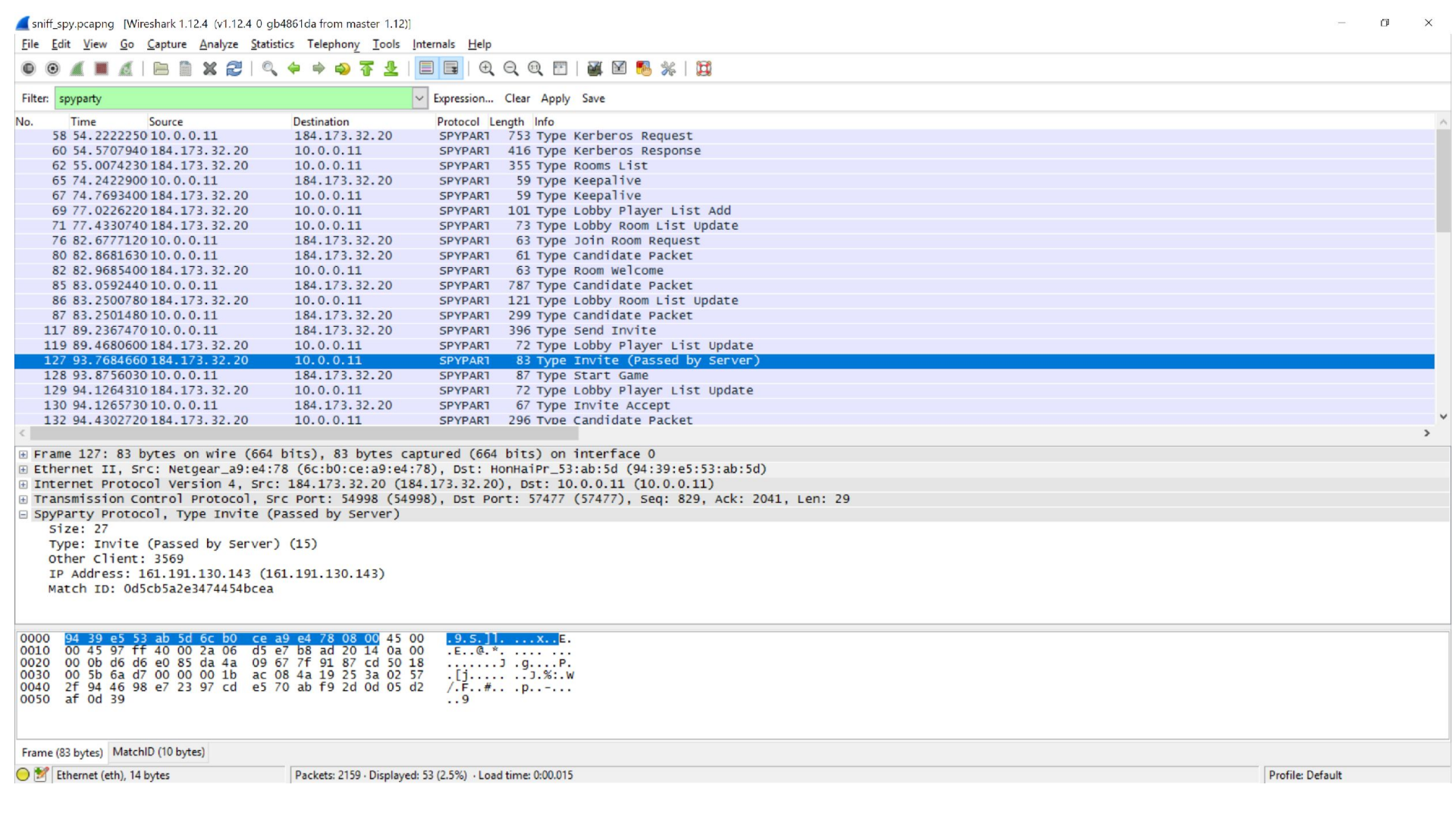
Windbg Script:
The following script can be used on the current client to get all kinds of nice prints:

```
bp 007E59F6 "r @$t0 = ecx; g;"
bp 007E59FB  "da @$t0; g;"

bp 00A449C0 ".echo Creating encoder; g;"
bp 00A44D2B ".echo 'Encode string of length'; rebx; .echo 'String data is'; db edi L0x100; r @$t3=1; g;"
bp 00A44E08 "r @$t3=0; g;"
bp 00A44BAC ".if (@$t3 == 0) { .echo 'Encode value:'; rebx; .echo 'And limit:'; redi; }; g;"

bp 00A44C36 ".echo Encoding bits; rebx; r @$t3=1; g;"
bp 00A44C6C ".echo Value; redi; g;"
bp 00A44CAA "r @$t3=0; g;"
bp 00A44D11 "r @$t3=0; g;"

bp 00A44400 ".echo Creating decoder; g;"
bp 00A44671 ".echo 'Decode string of length'; rebx; r @$t1 = esi; r @$t4=1; g;"
bp 00A446E4 ".echo 'String data:'; db @$t1 L0x100; r @$t4=0; g;"
bp 00A446AB ".echo 'String data:'; db @$t1 L0x100; r @$t4=0; g;"

bp 00A44505 ".if (@$t4 == 0) {.echo 'Decode limit:'; rebx; }; g;"
bp 00A44599 ".if (@$t4 == 0) {.echo 'Value'; reax; }; g; "
bp 00A44562 ".if (@$t4 == 0) {.echo 'Value'; reax; }; g; "

bp 00A445B5 ".echo 'Decoding bits'; rebx; r @$t4=1; g;"
bp 00A445EA ".echo 'Decoded bits result:'; reax; r @$t4=0; g;"
bp 00A4464D ".echo 'Decoded bits result:'; reax; r @$t4=0; g;"
```

Wireshark Plugin

```
typedef enum {
    LOBBY_QUERY = 0,
    LOBBY_QUERY_RESPONSE = 1,
    KERBEROS_REQUEST = 2,
    KERBEROS_RESPONSE = 3,
    JOIN_ROOM_REQUEST = 5,
    ROOM_WELCOME = 6,
    LOBBY_PLAYER_LIST = 8,
    RECEIVE_CHAT_MESSAGE = 9,
    SEND_CHAT_MESSAGE = 10,
    KEEPALIVE = 11,
    SEND_INVITE = 12,
    INVITE_ACCEPT = 13,
    INVITE_PASSED_BY_SERVER = 14,
    INVITE_ACCEPT_PASSED_BY_SERVER = 15,
    START_GAME = 16,
    CANDIDATE_PACKET = 17,
    REMOTE_ASSERTION_ERROR = 23,
    ROOMS_LIST = 28,
    LOBBYCLIENT_IDLE = 33,
    LOBBY_PLAYER_LIST_ADD = 34,
    LOBBY_PLAYER_LIST_UPDATE = 35,
    LOBBY_ROOM_LIST_UPDATE = 36,
    CANCEL_GAME = 38,
} spyparty_packet_type;

static const value_string packettypenames[] = {
    // TODO: Replace these values with enum above
    { 0, "Lobby Query??" },
    { 1, "Lobby Query Response??" },
    { 2, "Kerberos Request" },
    { 3, "Kerberos Response" },
    { 5, "Join Room Request" },
```

```
static int
dissect_spyparty_packet(tvbuff_t *tvb, packet_info *pinfo, proto_tree *tree, void *data _U_)
{
    gint offset = 0;
    guint16 packet_size = tvb_get_ntohs(tvb, 0);
    guint8 packet_type;
    int a, b;

    Decoder decoder;
    decoder_init(&decoder, tvb, packet_size);
    a = decoder_limited_decode(&decoder, 3);
    b = decoder_limited_decode(&decoder, 1009);
    if (a != 2 || b != 16) {
        return 0;
    }
    offset = decoder.input_byte_cursor;
    packet_type = decoder_limited_decode(&decoder, 101);

    col_set_str(pinfo->cinfo, COL_PROTOCOL, "SPYPARTY");
    /* Clear out stuff in the info column */
    col_clear(pinfo->cinfo,COL_INFO);
    col_add_fstr(pinfo->cinfo, COL_INFO, "Type %s",
        val_to_str(packet_type, packettypenames, "Unknown (%d)"));

    if (tree) { /* we are being asked for details */
        proto_item *ti = NULL;
        proto_tree *spyparty_tree = NULL;

        ti = proto_tree_add_item(tree, proto_spyparty, tvb, 0, -1, ENC_NA);
        proto_item_append_text(ti, ", Type %s",
            val_to_str(packet_type, packettypenames, "Unknown (%d)"));
        spyparty_tree = proto_item_add_subtree(ti, ett_spyparty);

        proto_tree_add_uint(spyparty_tree, hf_spyparty_size, tvb, 0, 2, packet_size);

        proto_tree_add_uint(spyparty_tree, hf_spyparty_type, decoder.tvb, offset, decoder.input_byte_cursor - offset , p
        offset = decoder.input_byte_cursor;

       /* The correct way to work is to define a new dissector for each packet type
          and call the next dissector with dissector_next. Screw that.*/
        switch (packet_type) {
```


sniff_spy.pcapng [Wireshark 1.12.4 (v1.12.4 0 gb4861da from master 1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: spyparty
Expression... Clear Apply Save
No. Time Source Destination Protocol Length Info
58 54.2222250 10.0.0.11 184.173.32.20 SPYPARTY 753 Type Kerberos Request
60 54.5707940 184.173.32.20 10.0.0.11 SPYPARTY 416 Type Kerberos Response
62 55.0074230 184.173.32.20 10.0.0.11 SPYPARTY 355 Type Rooms List
65 74.2422900 10.0.0.11 184.173.32.20 SPYPARTY 59 Type Keepalive
67 74.7693400 184.173.32.20 10.0.0.11 SPYPARTY 59 Type Keepalive
69 77.0226220 184.173.32.20 10.0.0.11 SPYPARTY 101 Type Lobby Player List Add
71 77.4330740 184.173.32.20 10.0.0.11 SPYPARTY 73 Type Lobby Room List Update
76 82.6777120 10.0.0.11 184.173.32.20 SPYPARTY 63 Type Join Room Request
80 82.8681630 10.0.0.11 184.173.32.20 SPYPARTY 61 Type Candidate Packet
82 82.9685400 184.173.32.20 10.0.0.11 SPYPARTY 63 Type Room Welcome
85 83.0592440 10.0.0.11 184.173.32.20 SPYPARTY 787 Type Candidate Packet
86 83.2500780 184.173.32.20 10.0.0.11 SPYPARTY 121 Type Lobby Room List Update
87 83.2501480 10.0.0.11 184.173.32.20 SPYPARTY 299 Type Candidate Packet
117 89.2367470 10.0.0.11 184.173.32.20 SPYPARTY 396 Type Send Invite
119 89.4680600 184.173.32.20 10.0.0.11 SPYPARTY 72 Type Lobby Player List Update
127 93.7684660 184.173.32.20 10.0.0.11 SPYPARTY 83 Type Invite (Passed by Server)
128 93.8756030 10.0.0.11 184.173.32.20 SPYPARTY 87 Type Start Game
129 94.1264310 184.173.32.20 10.0.0.11 SPYPARTY 72 Type Lobby Player List Update
130 94.1265730 10.0.0.11 184.173.32.20 SPYPARTY 67 Type Invite Accept
132 94.4302720 184.173.32.20 10.0.0.11 SPYPARTY 296 Type Candidate Packet
Frame 127: 83 bytes on wire (664 bits), 83 bytes captured (664 bits) on interface 0
Ethernet II, Src: Netgear_a9:e4:78 (6c:b0:ce:a9:e4:78), Dst: HonHaiPr_53:ab:5d (94:39:e5:53:ab:5d)
Internet Protocol Version 4, Src: 184.173.32.20 (184.173.32.20), Dst: 10.0.0.11 (10.0.0.11)
Transmission Control Protocol, Src Port: 54998 (54998), Dst Port: 57477 (57477), Seq: 829, Ack: 2041, Len: 29
SpyParty Protocol, Type Invite (Passed by Server)
Size: 27
Type: Invite (Passed by Server) (15)
Other Client: 3569
IP Address: 161.191.130.143 (161.191.130.143)
Match ID: 0d5cb5a2e3474454bcea
0000 94 39 e5 53 ab 5d 6c b0 ce a9 e4 78 08 00 45 00 .9.S.]l. ...x..E.
0010 00 45 97 ff 40 00 2a 06 d5 e7 b8 ad 20 14 0a 00 .E..@.*. ....
0020 00 0b d6 d6 e0 85 da 4a 09 67 7f 91 87 cd 50 18 ....J .g....P.
0030 00 5b 6a d7 00 00 00 1b ac 08 4a 19 25 3a 02 57 .[j..... ..J.%:.W
0040 2f 94 46 98 e7 23 97 cd e5 70 ab f9 2d 0d 05 d2 /.F..#.. .p..-...
0050 af 0d 39 ..9
Frame (83 bytes) MatchID (10 bytes)
Ethernet (eth), 14 bytes
Packets: 2159 · Displayed: 53 (2.5%) · Load time: 0:00.015
Profile: Default

Parse Packets

- Chat works!
- Joining rooms works!
- Invitations work!
- Accepting invitations work!
- … candidate packet?

```
# A bit of a guess
enc.encode_bits(32, sent_from.id)
enc.encode_bits(32, self.current_room.id)

enc.encode_limited_string(sent_from.username, 0x21)
```

How do two computers connect on the internet?

Candidate Packet (ICE)

- Describes a possible way for computers to connect (direct IP, NAT, both behind NATs, both behind same NATs, NAT holepunching, relay)
- Looks something like this

192.168.4.1:52668,SpyParty1.0,bG9senlvdXJzc28xMzM3

219.49.13.37:52668,SpyParty1.0,cGFzdGVud3Uxd3UzaTM3

- Data on wire is encrypted somehow on one end, sent to the server, changed somehow (?) before being sent to the second client
- Why is it encrypted? Why does it change between clients?
- I am stuck… for quite a while

And then… find encryption!

And then… find encryption!

```
dd offset aAes256CtsHmacSha196 ; "aes256-cts-hmac-sha1-96"
dd offset aAes256Cts    ; "aes256-cts"
dd 0
dd offset aAes256CtsModeWith96BitSha1Hmac ; "AES-256 CTS mode with 96-bit SHA-1 HMAC"
dd offset unk_E01450
dd offset aSha1         ; "SHA1"
dd 10h
dd offset sub_6E16D6
dd offset sub_6EF592
dd offset sub_6F66D5
dd offset sub_6F5140
dd offset sub_6FC143
```

- The function we're looking at is actually part of Kerberos
- The candidate is encrypted using an internal mechanism called **priv**
- After the client connects, the server can use the functions rd_priv and mk_priv to decrypt/encrypt respectively
- Client 1 defines a candidate:

  192.168.4.1:52668,SpyParty1.0,bG9senlvdXJzc28xMzM3
- Client 1 sends mk_priv(candidate)
- Server decrypts with rd_priv(candidate)
- Server sends to client2 mk_priv(candidate)
- Client2 decrypts with rd_priv

Using priv API

- Failure
- Invalid net ADDR
- What does that mean?

Using priv API

- Failure
- Invalid net ADDR
- What does that mean?
- Metadata when encrypting
- Check on decrypt
- When decrypting, kerberos checks that IP in metadata matches
- Kerberos logic is happening in .so wrapped in python code
- Attach gdb to python, skip the check

Using priv API

- Failure
- Invalid net ADDR
- What does that mean?
- Metadata when encrypting
- Check on decrypt
- When decrypting, kerberos checks that IP in metadata matches
- Kerberos logic is happening in .so wrapped in python code
- Attach gdb to python, skip the check
- The Candidate packet is decrypted

Why does IP check fail?

- Compares the IP in the packet - the correct IP is 10.0.0.1 at port 52668, it is compared to
  -
- The second side should be 10.0.0.5 on port 10254, instead it’s
  -

Why does IP check fail?

- Compares the IP in the packet - the correct IP is 10.0.0.1 at port 52668, it is compared to
  - 1.2.3.4 on port 0x0102
- The second side should be 10.0.0.5 on port 10254, instead it’s
  -

Why does IP check fail?

- Compares the IP in the packet - the correct IP is 10.0.0.1 at port 52668, it is compared to
  - 1.2.3.4 on port 0x0102
- The second side should be 10.0.0.5 on port 10254, instead it’s
  - 5.6.7.8 on port 0x0506

Why does IP check fail?

- Compares the IP in the packet - the correct IP is 10.0.0.1 at port 52668, it is compared to
  - 1.2.3.4 on port 0x0102
- The second side should be 10.0.0.5 on port 10254, instead it’s
  - 5.6.7.8 on port 0x0506
- WTF?!
- Developer misusing private API :(

Final touches

- Parse the rest of the packet types
- It’s working!
- End of story?

Later…

I reverse engineered your game

Inbox ×

Joey Dec 24, 2016, 2:18 PM

to support

Dear checker,

I started playing SpyParty a few years ago, and I really love your game. Back in 2014 a friend of mine hosted an offline LAN party with around 20 other players, and I really wanted all of them to try it out. But, even though I had purchased multiple copies of the game we couldn't play there because the game only works when connected to the internet…

SpyParty Support <support@spyparty.com> Dec 24, 2016, 6:31 PM

to me

What the hell? This is not cool, and obviously illegal. I'm suing you for everything you have, expect my lawyer to contact you sometime soon.

Thanks,
Chris

SpyParty Support <support@spyparty.... Sat, Dec 24, 2016, 6:31 PM

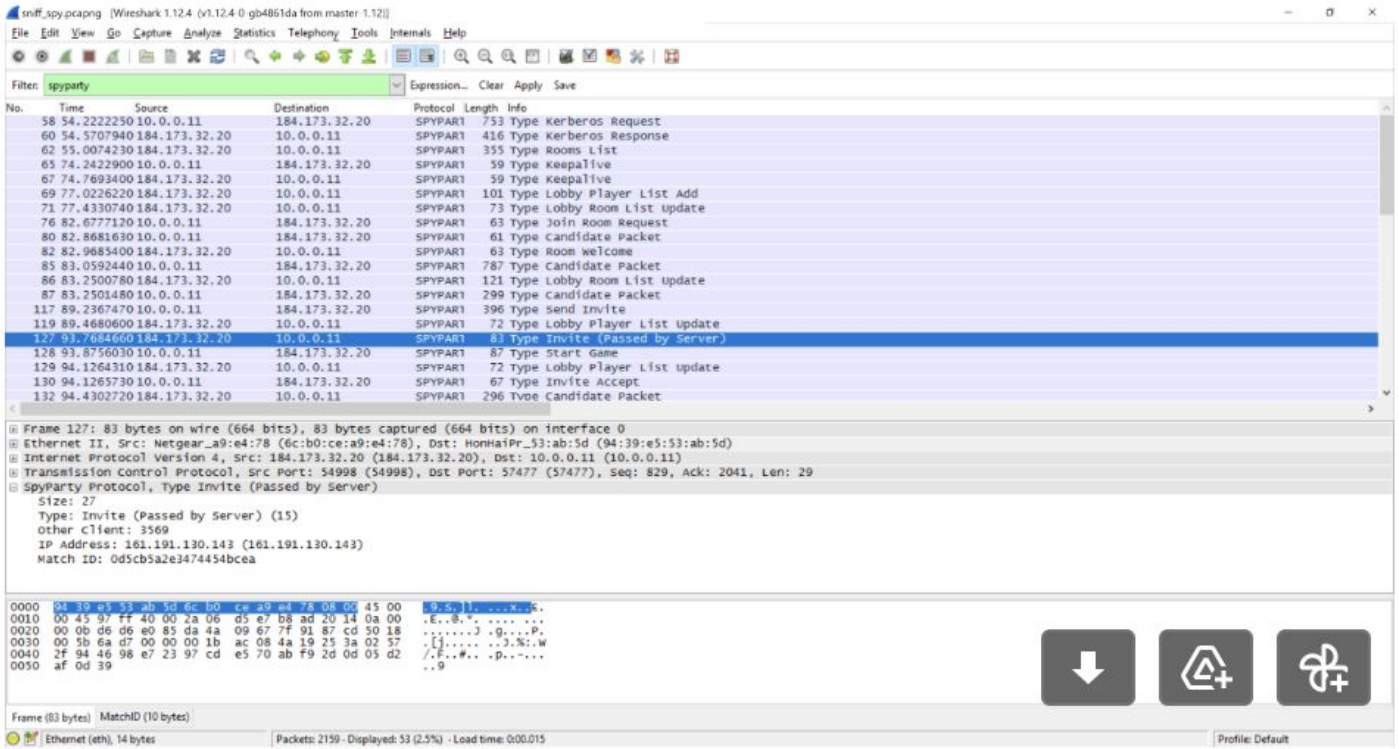
to me

Hah, awesome! I was wondering if somebody was going to try that! Thanks for the mail and the kind words, and glad you guys like the game!

Yeah, I'd love to get your notes and bugs and I'm happy to answer questions if I can!

Thanks,
Chris

Also - this part you're going to love - in addition to the server code I created a wireshark plugin that analyses and does basic parsing of spyparty game traffic. Here's a screenshot:



I'll be happy to share this with you as well :)

SpyParty Support <support@spyparty.com> Fri, Dec 30, 2016, 7:18 PM

to me

This is awesome; more later due to holiday craziness! I'll make a bit bucket account!

Chris

•••

The end

- Play the game?
  - Lobby story
- Fixing the internet

Questions?