FOX IT

Creating the Bro RFB (VNC) parser

Martin van Hensbergen, Fox-IT

FOX IT

Agenda

- Introduction
- Context: How we use Bro
- The dangers of VNC
- VNC protocol
- Dev
- Deploy
- Future work

FOX IT

Introduction

- Martin van Hensbergen - Fox-IT
  - Studied Mathematics at University of Delft
  - Worked at Fox-IT 2001-2011 + 2016-?
  - Mostly as developer but also in few other areas
- 2007-2011, worked on FoxReplay
  - Software for full-content reconstruction of network data
  - Lawful interception & forensics purposes
  - Required network protocol knowledge

FOX IT

Bro at Fox-IT

FOX IT

Bro at Fox-IT

- We use Bro in three major services:
  - Passive Audits - 🤔
  - Compromise Assessments - 😨
  - Incident Response - 😵

FOX IT

Bro at Fox-IT

- We use Bro in three major services:
  - Passive Audits - network 🤔
  - Compromise Assessments - 😨
  - Incident Response - 😵

FOX IT

Bro at Fox-IT

- We use Bro in three major services:
  - Passive Audits - network 🤔
  - Compromise Assessments - network+hosts 😨
  - Incident Response - 😵

FOX IT

Bro at Fox-IT

- We use Bro in three major services:
  - Passive Audits - network 🤔
  - Compromise Assessments - network+hosts 😰
  - Incident Response - network+hosts 😵

FOX IT

Bro at Fox-IT - Passive Audit

- We take a ‘photograph’ of the network by passively monitoring 4 weeks of network traffic
- Combination of:
  - Bro
  - Suricata
  - Custom tooling

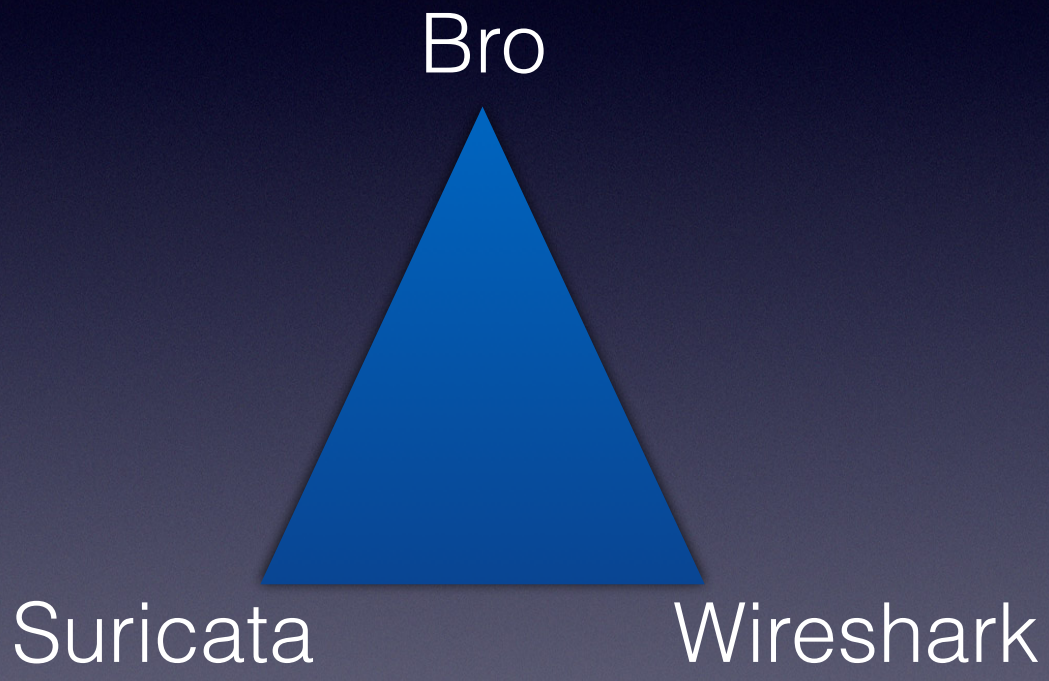
FOX IT

Bro at Fox-IT - Passive Audit

- Bro gives us a very detailed run-down on:
  - Protocols used in a network
  - Flow data
- Suricata gives us alerting on known-bad

FOX IT

Bro at Fox-IT - Passive Audit



- Use strengths of multiple products

FOX IT

Bro at Fox-IT - Passive Audit

- Mix: Automated and manual analysis
- Deliver report on security of the network

FOX IT

Bro at Fox-IT - Passive Audit

- Some things we look for:
  - Weak protocols (security wise) / SSL configs / Plaintext passwords
  - ‘Weird’ traffic / Context surrounding alerts
  - Network segmentation
  - Services exposed to e.g. outside world
  - Remote administration tools
    - RDP ... why not RFB/VNC?

FOX IT

VNC basics

Virtual Network Computing

From Wikipedia, the free encyclopedia

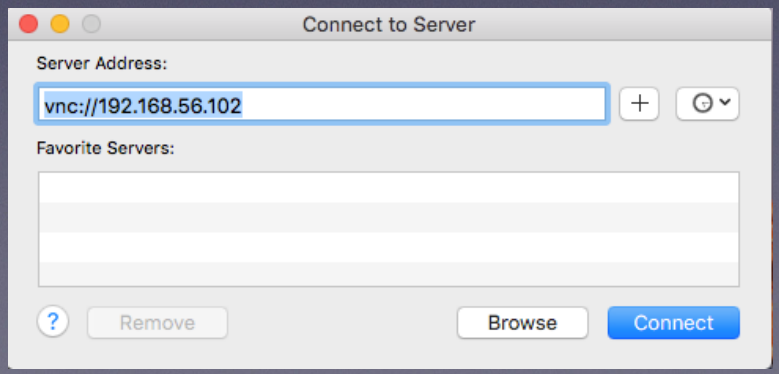
In computing, **Virtual Network Computing** (**VNC**) is a graphical desktop sharing system that uses the Remote Frame Buffer protocol (RFB) to remotely control another computer. It transmits the keyboard and mouse events from one computer to another, relaying the graphical screen updates back in the other direction, over a network.[1]

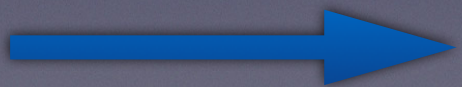
- Original spec (v3.3) by Olivetti Research Lab in 1998, later maintained by RealVNC: v3.7 in 2003 and v3.8 in 2007.
- Protocol published under RFC6143 by RealVNC in 2011

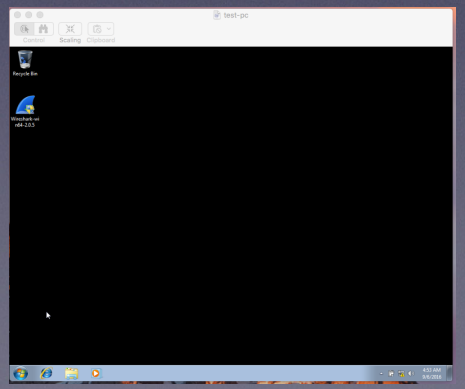
FOX IT

VNC basics

- Server runs RFB server (e.g. RealVNC server); listens on (default) TCP port 5900
  - RFB client connects over network
  - Client can control server over network







FOX IT

The dangers of VNC

FOX IT

The dangers of VNC

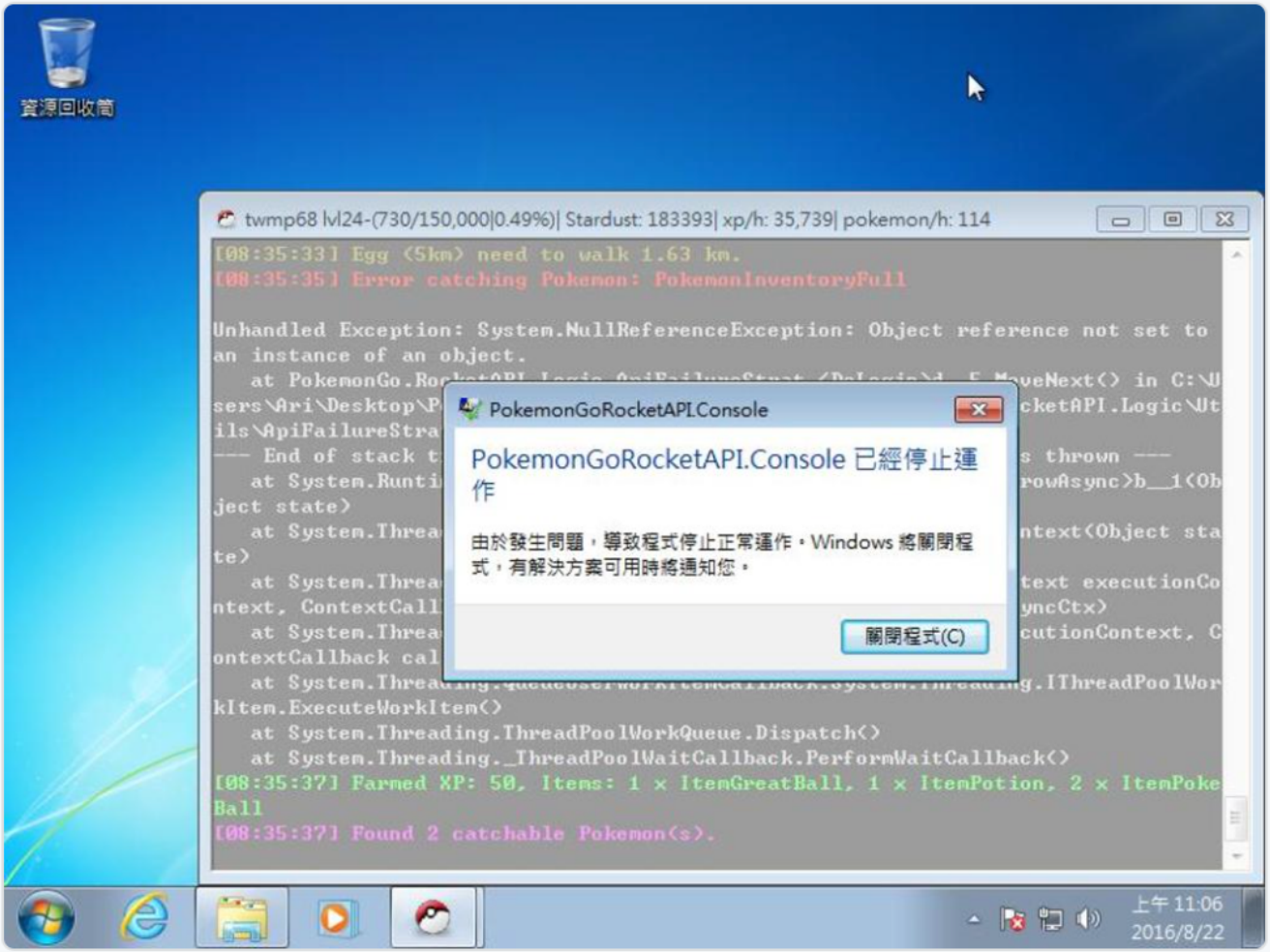
- My colleague Yonathan Klijnsma did some research on publicly reachable VNC servers
- It's 2016 .... VNC IS EVERYWHERE!

FOX IT

The dangers of VNC

Yonathan Klijnsma @ydklijnsma · 6h

Watching people play Pokemon GO over VNC, apparently there are a bunch of bots around now. shodan.io/host/210.61.14…

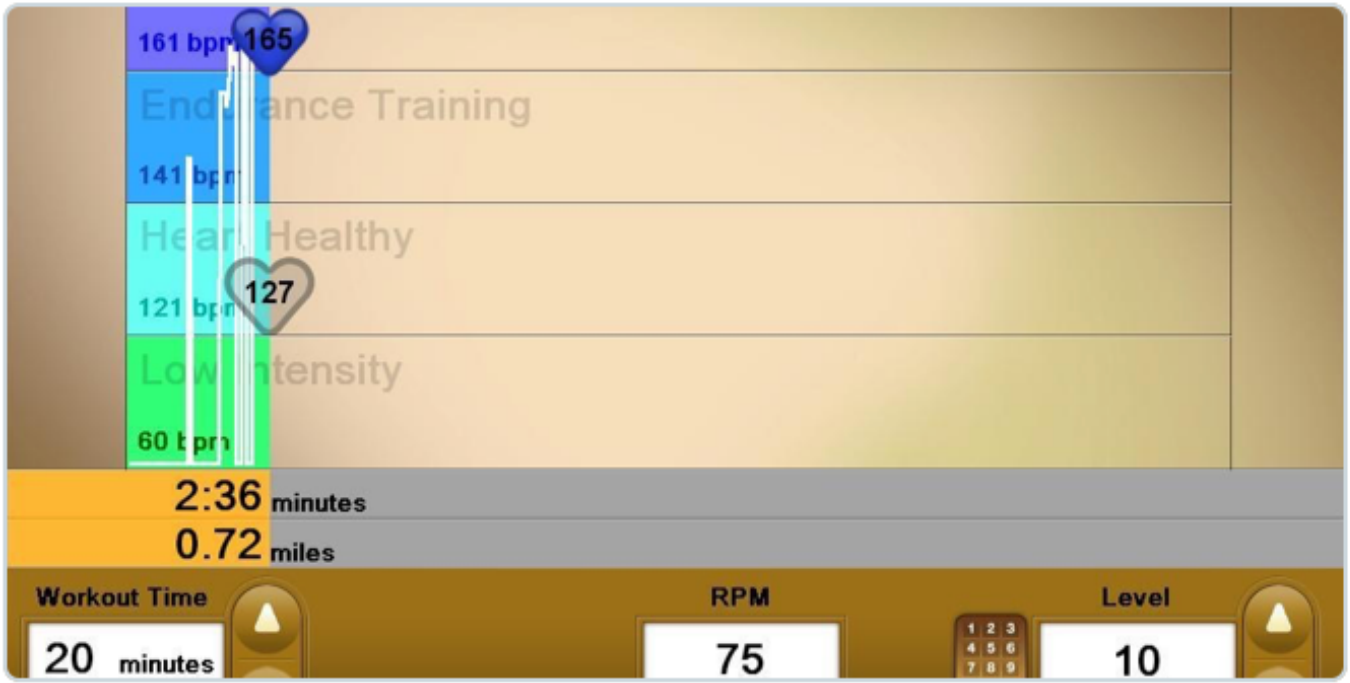


FOX IT

The dangers of VNC

Yonathan Klijnsma @ydklijnsma 9h

Watching people work out at the University of Hawaii because the IoT is amazing. shodan.io/host/128.171.2...



9 6

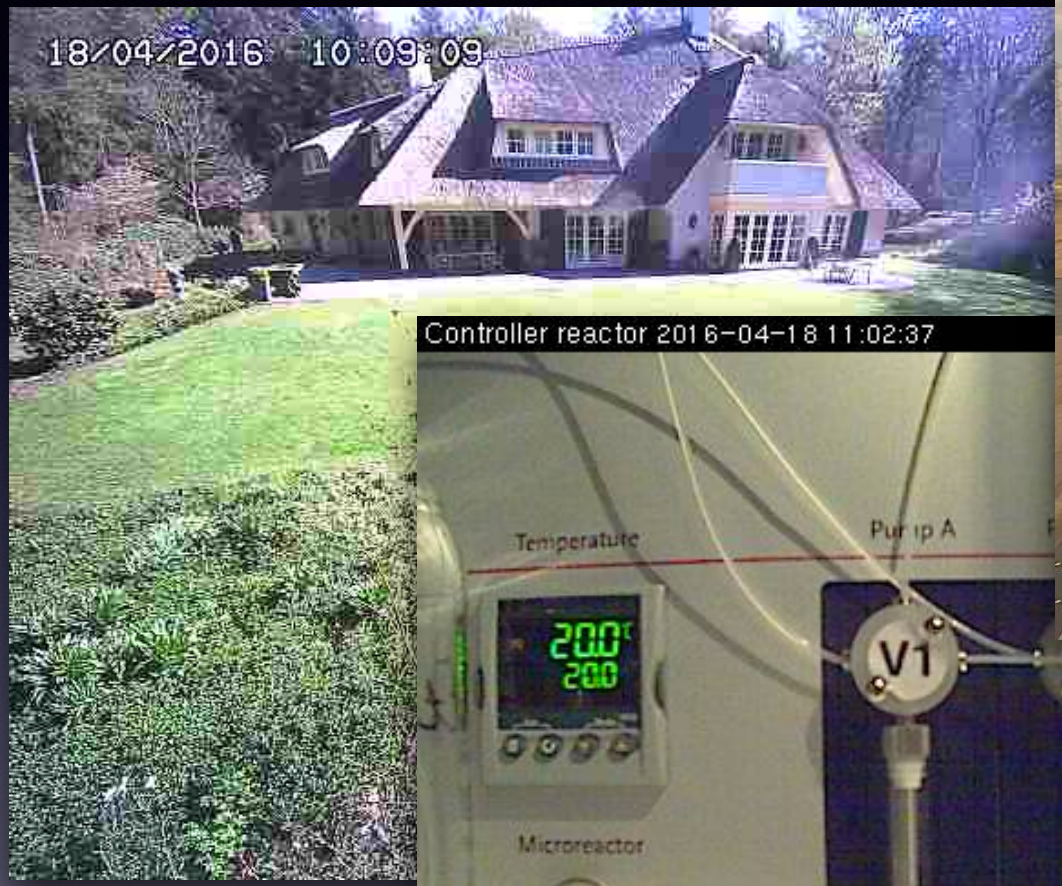
FOX IT

Dangers of VNC

- All good and fun until…

FOX IT

The dangers of VNC - IoT



FOX IT

Dangers of VNC

Yonathan Klijnsma @ydklijnsma · Feb 12

More open & unauthenticated VNC on medical devices: a cardiac imaging device: shodan.io/host/201.231.2… (cc @shodanhq)

22 16

FOX IT

Dangers of VNC

- VNC connections open to:
  - Medical devices
  - SCADA systems
  - Factories
  - Homes

FOX IT

Dangers of VNC

- VNC:
  - no- or weak authentications
  - unencrypted

FOX IT

Bro Wishlist

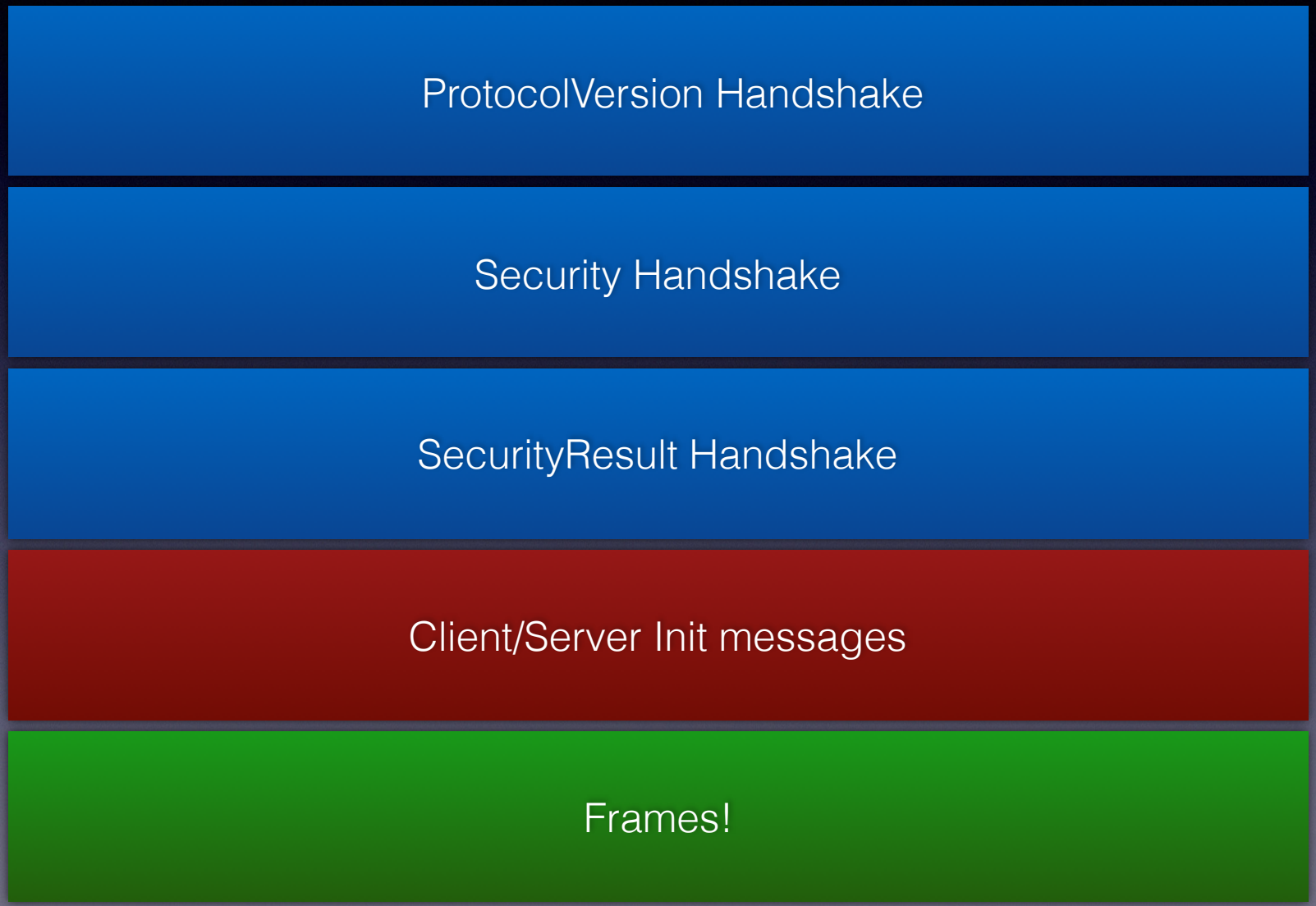
- What would we want to see from a security perspective:
  - are there RFB servers in the network?
  - from where and when are they accessed, for how long?
  - which software is used?
  - what kind of authentication is used, was it successful?
  - other useful information?
- Bonus exercise: can we get a screenshot? 😇

FOX IT

VNC protocol

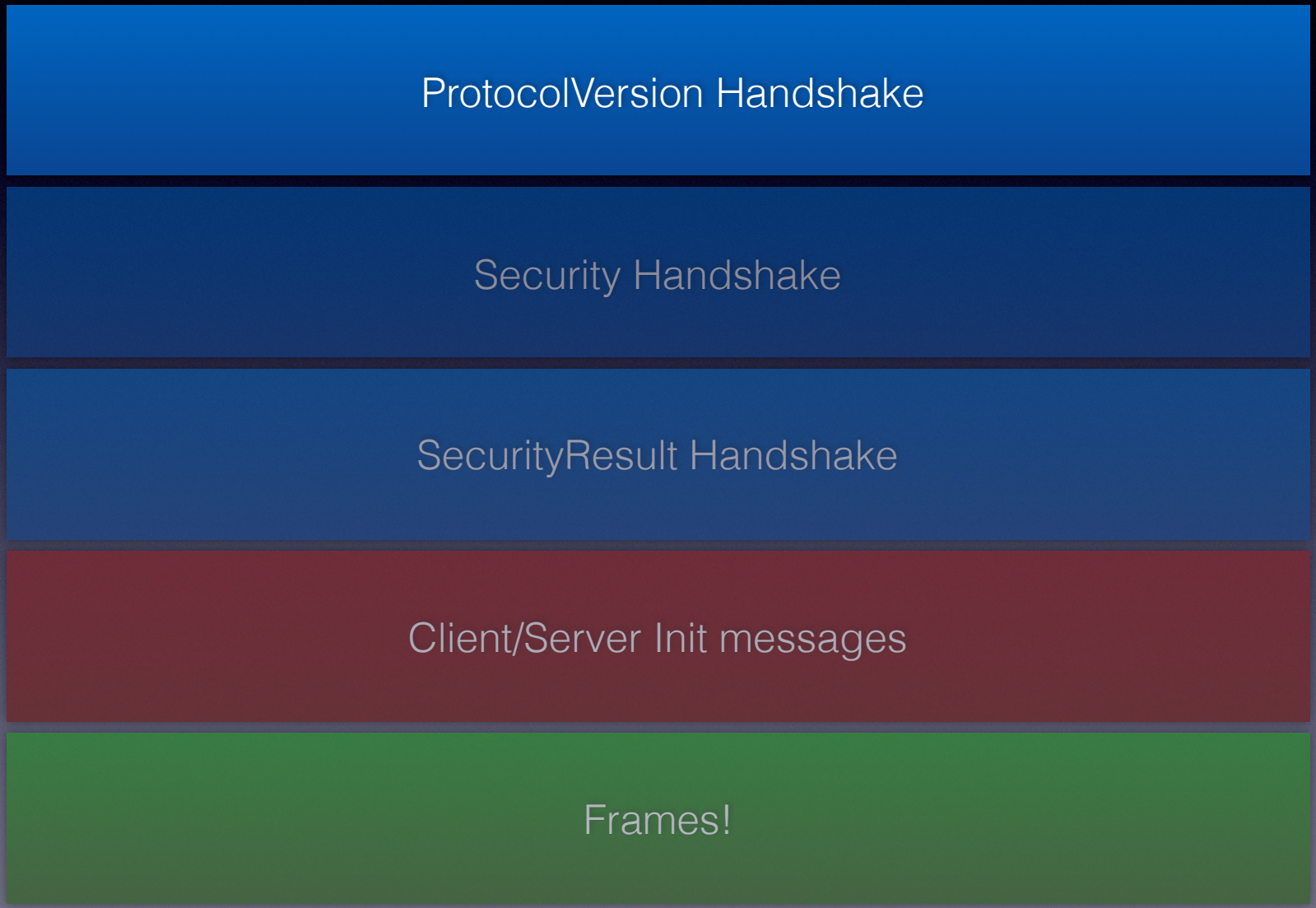
FOX IT

VNC protocol



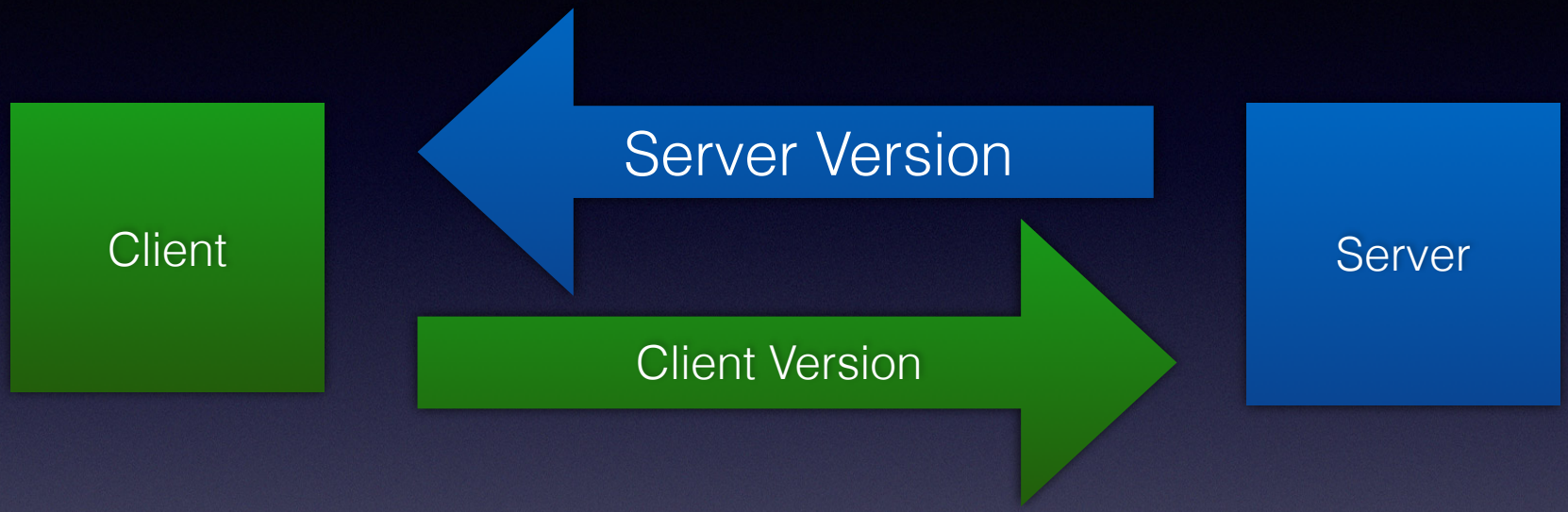
FOX IT

VNC protocol



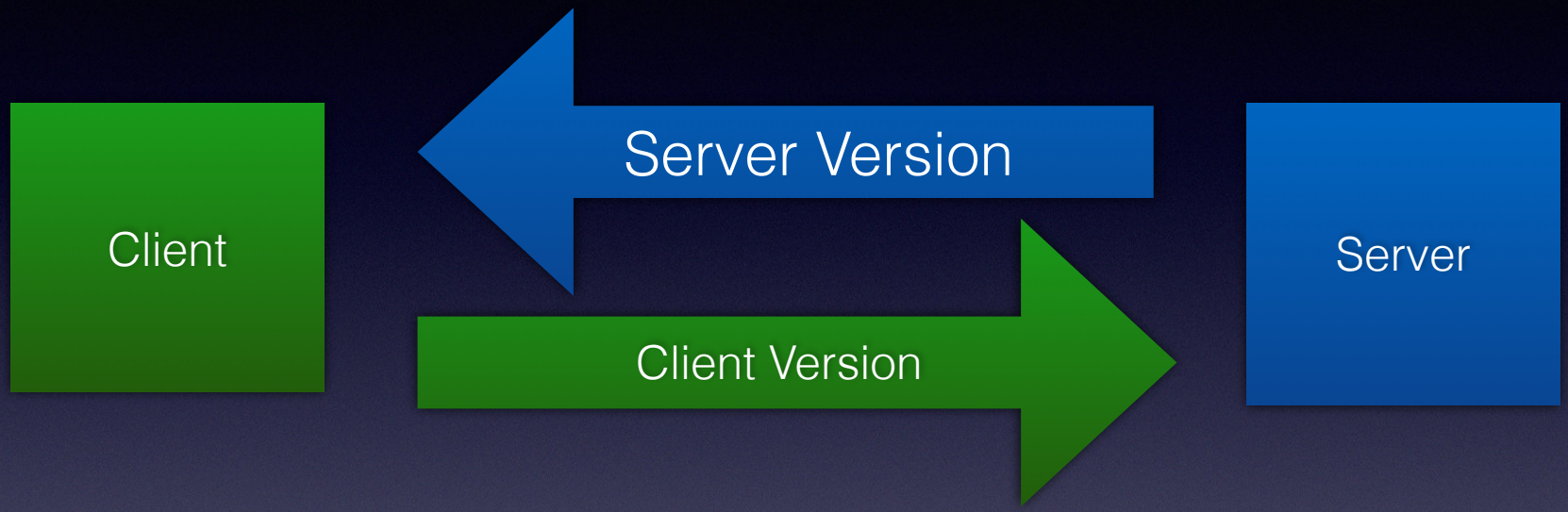
FOX IT

VNC protocol - Identification



FOX IT

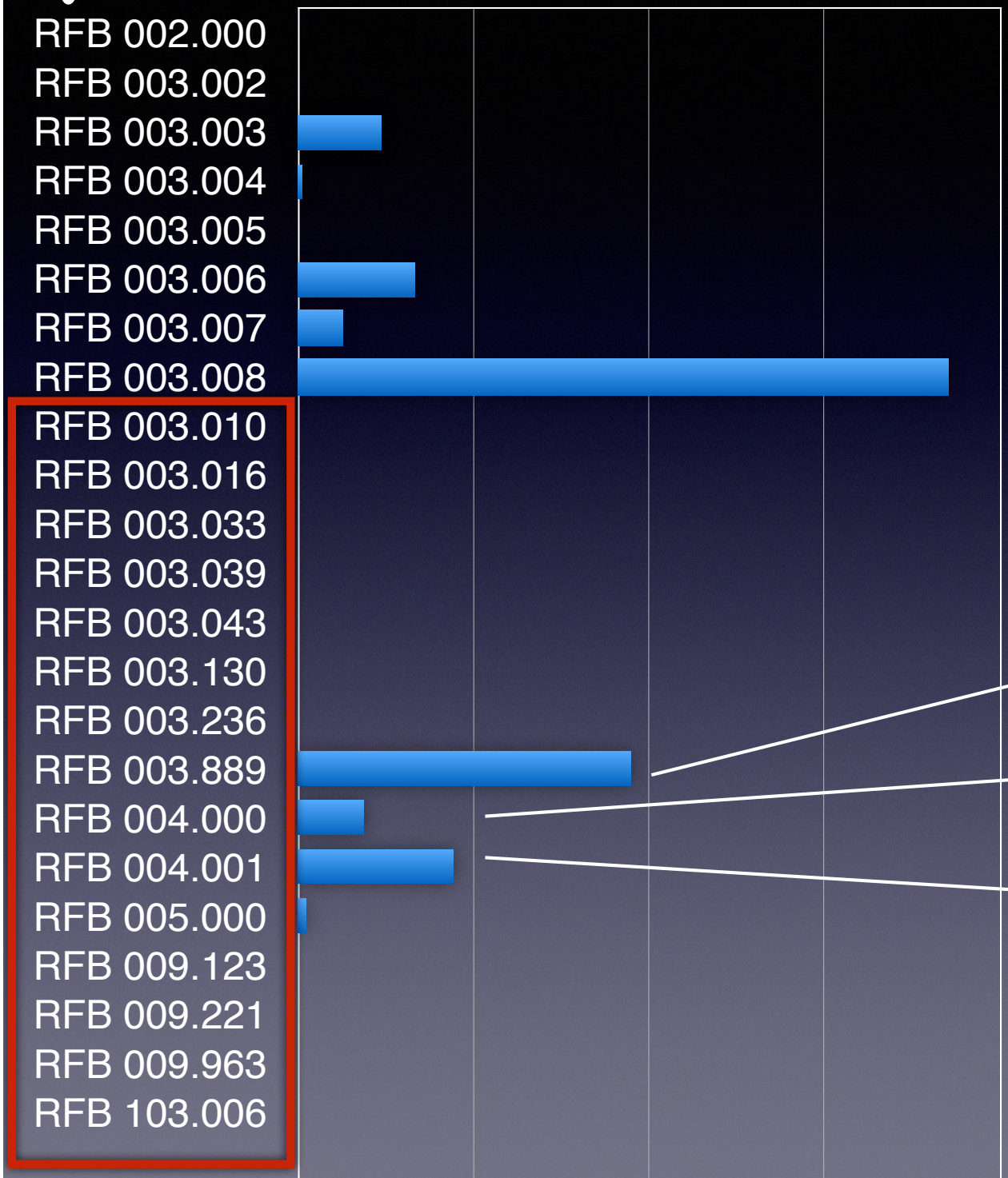
VNC protocol - Identification



12 byte string “RFB xxx.yyy\n”

RFB 003.003 - RFB 003.007 - RFB 003.008

FOX IT



Identified RFB headers in the wild.

Apple Remote Desktop

RealVNC Personal

RealVNC Enterprise

Source: Y. Klijnsma

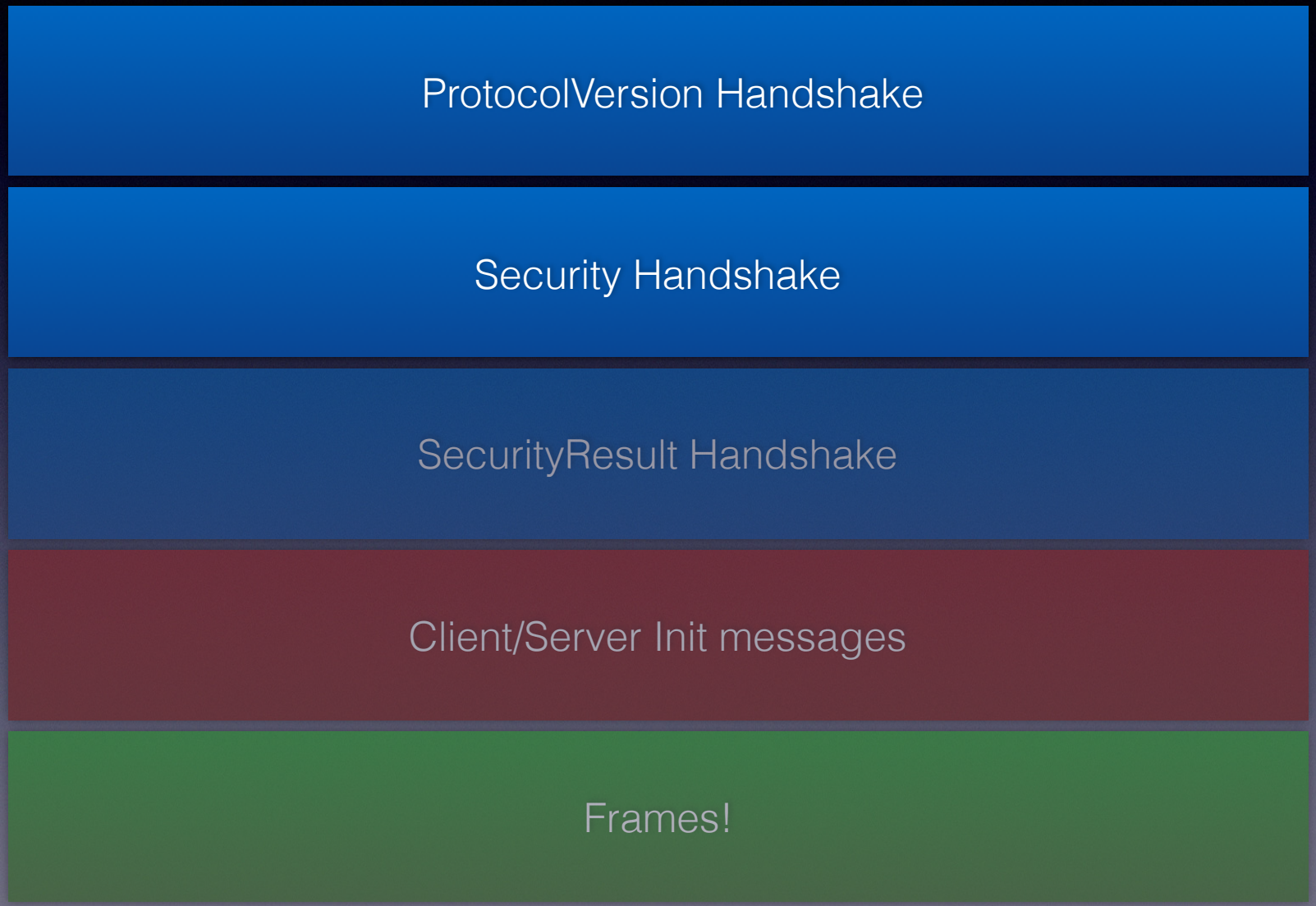
FOX IT

VNC protocol - Identification

- Certain version numbers can be attributed to certain software

FOX IT

VNC protocol



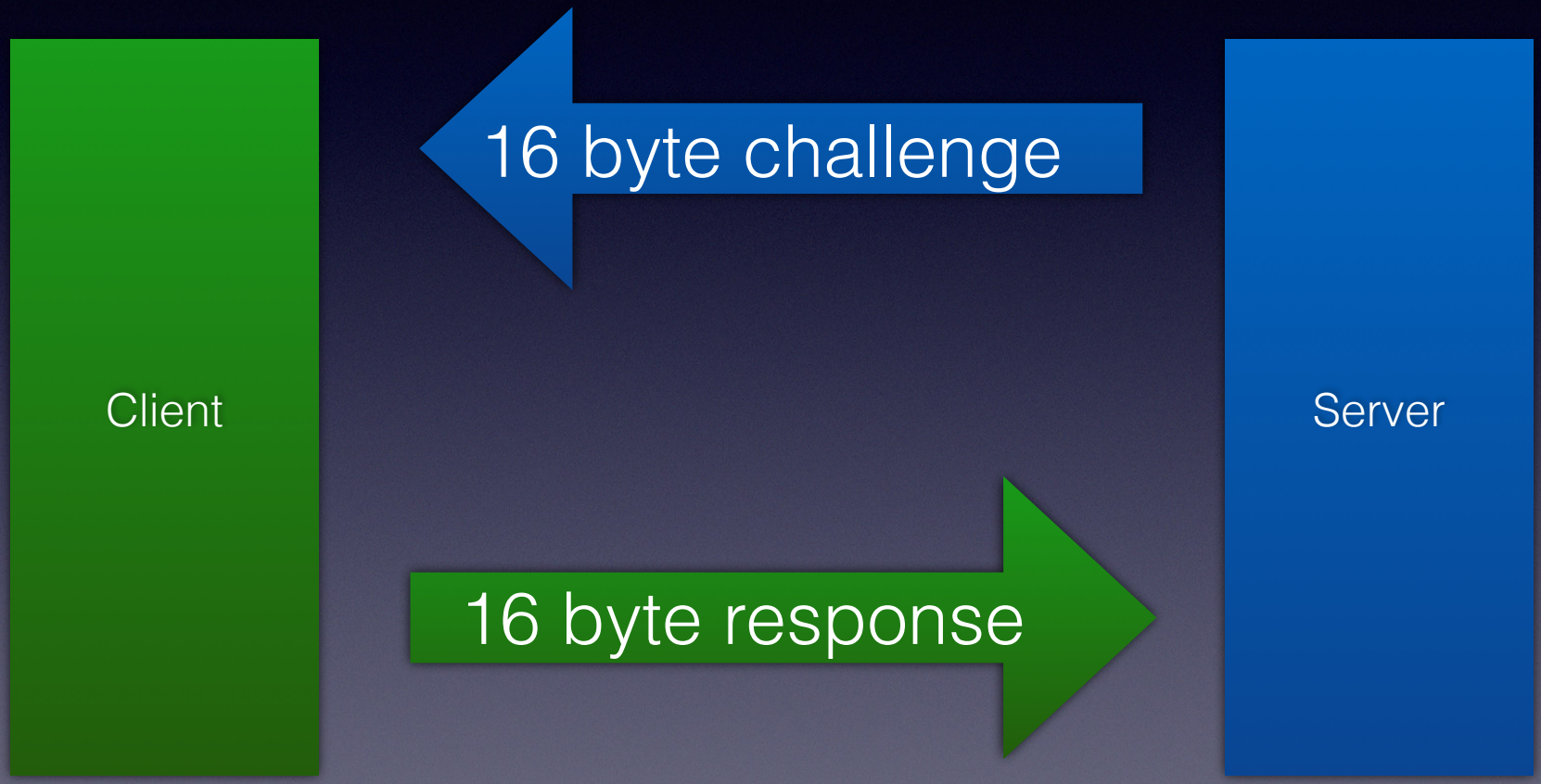
FOX IT

VNC protocol security

- Server sends a list of supported ‘security types’
- These determine form of authentication (examples):
  - 1 = No authentication
  - 2 = VNC authentication
  - 30 = Apple Remote Desktop authentication

FOX IT

VNC protocol - VNC authentication



DES(challenge) with password derived key

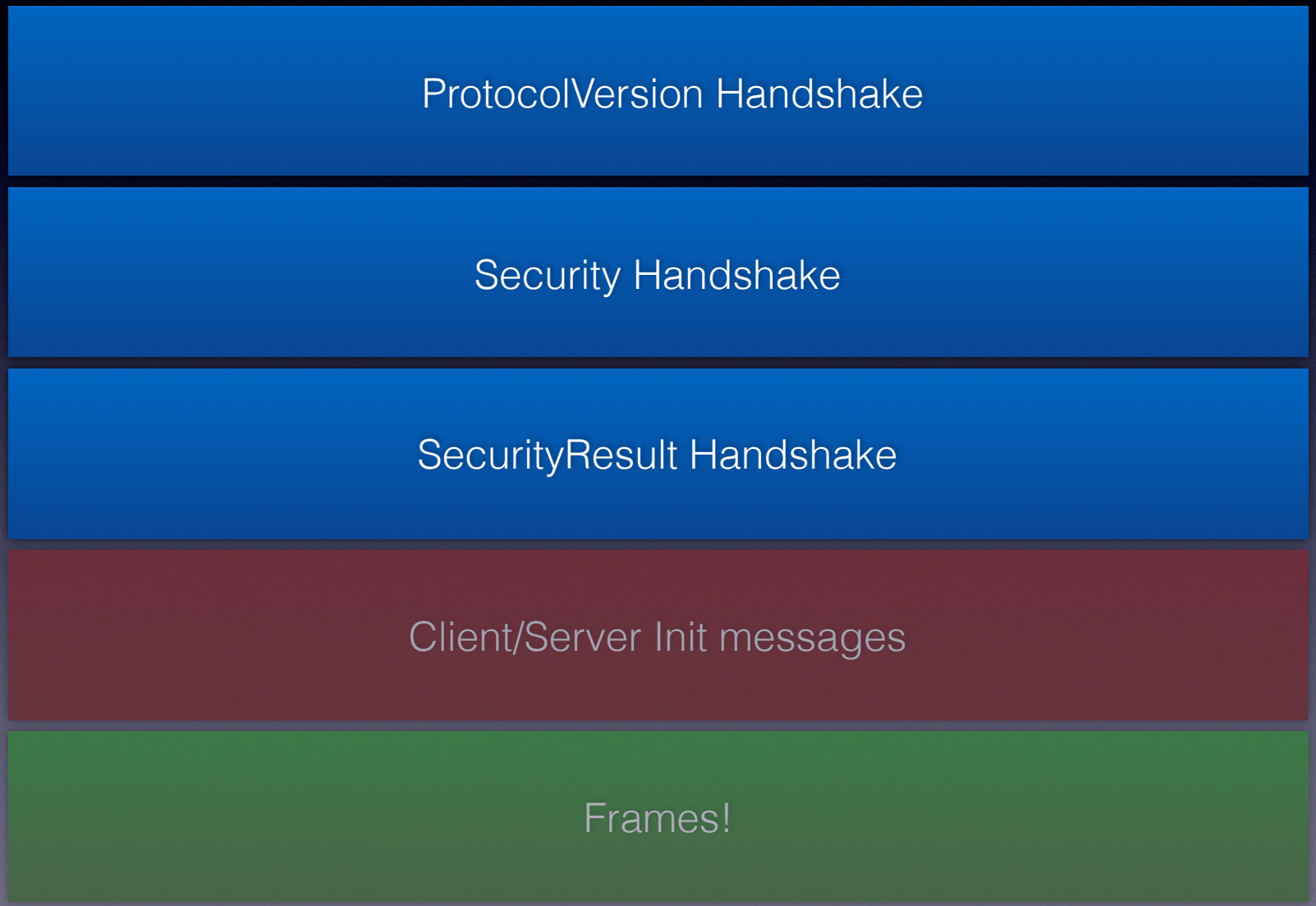
FOX IT

VNC protocol - VNC authentication

- Custom authentication types possible
- Found VNC server implementation that **does** send username/password in cleartext over wire
- 😳

FOX IT

VNC protocol



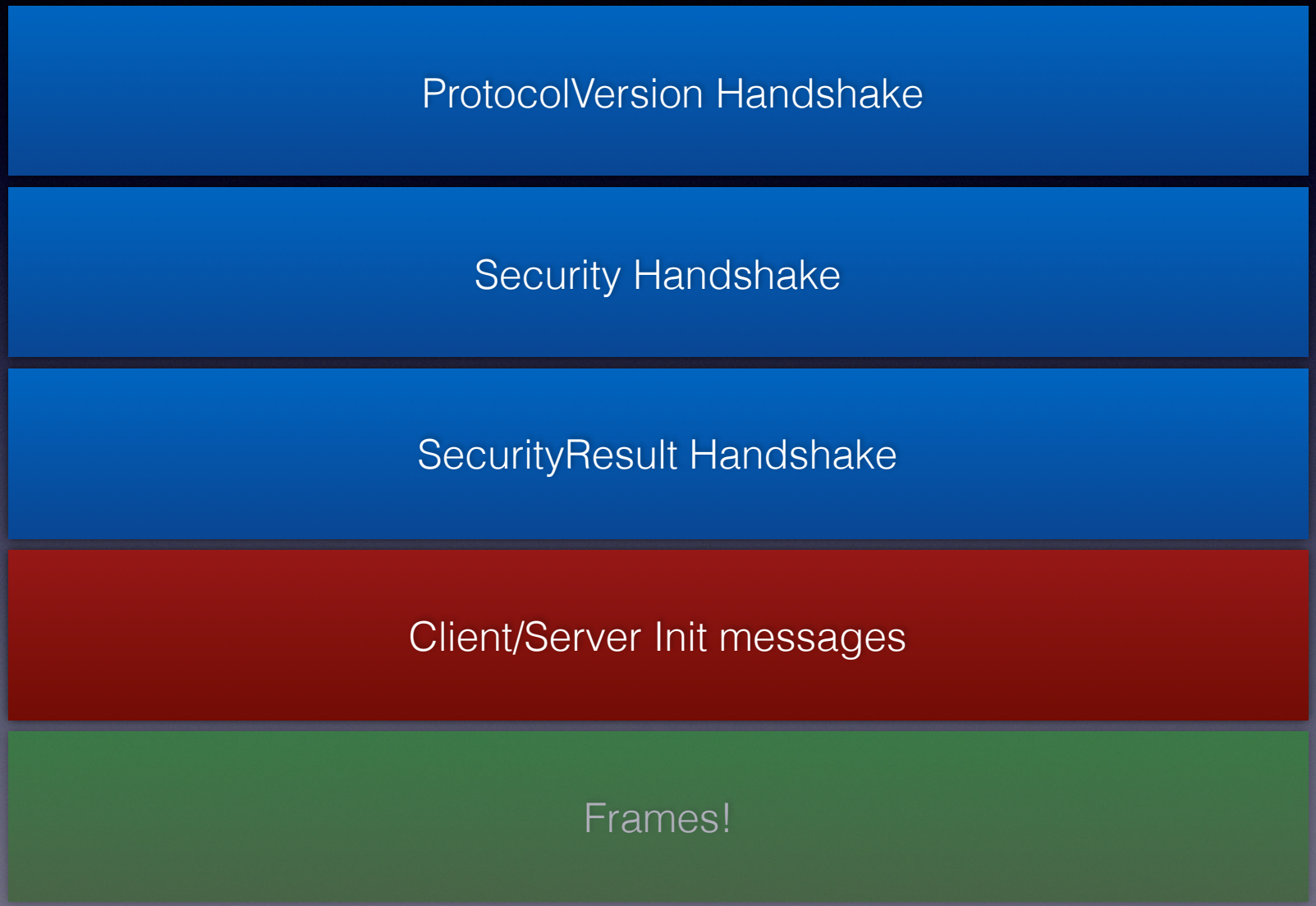
FOX IT

VNC protocol - Security result

- Server always sends an explicit acknowledgment if authentication succeeded.
- If not successful: connection aborted

FOX IT

VNC protocol



FOX IT

VNC protocol - Init messages

- Client sends ClientInit message with a ‘shared_flag’
- Shared flag determines mode of operation:
  - 1 = Allow other connections to remain if present
  - 0 = Disconnect other connections for exclusive access

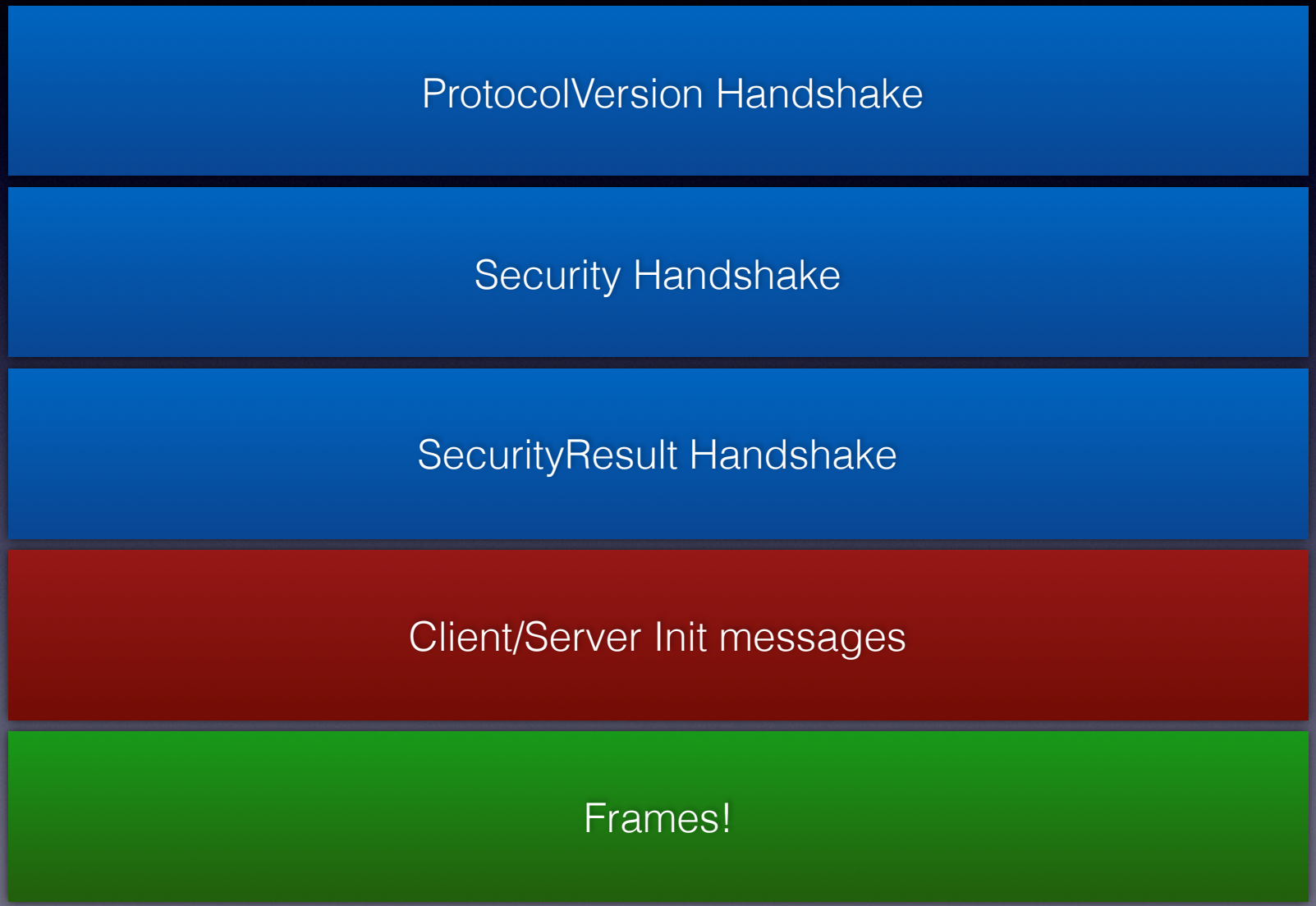
FOX IT

VNC protocol - Init messages

- Server sends ServerInitMsg, containing:
  - name of the server
  - width/height of shared screen in pixels
  - 16 bytes of pixel information encoding information

FOX IT

VNC protocol



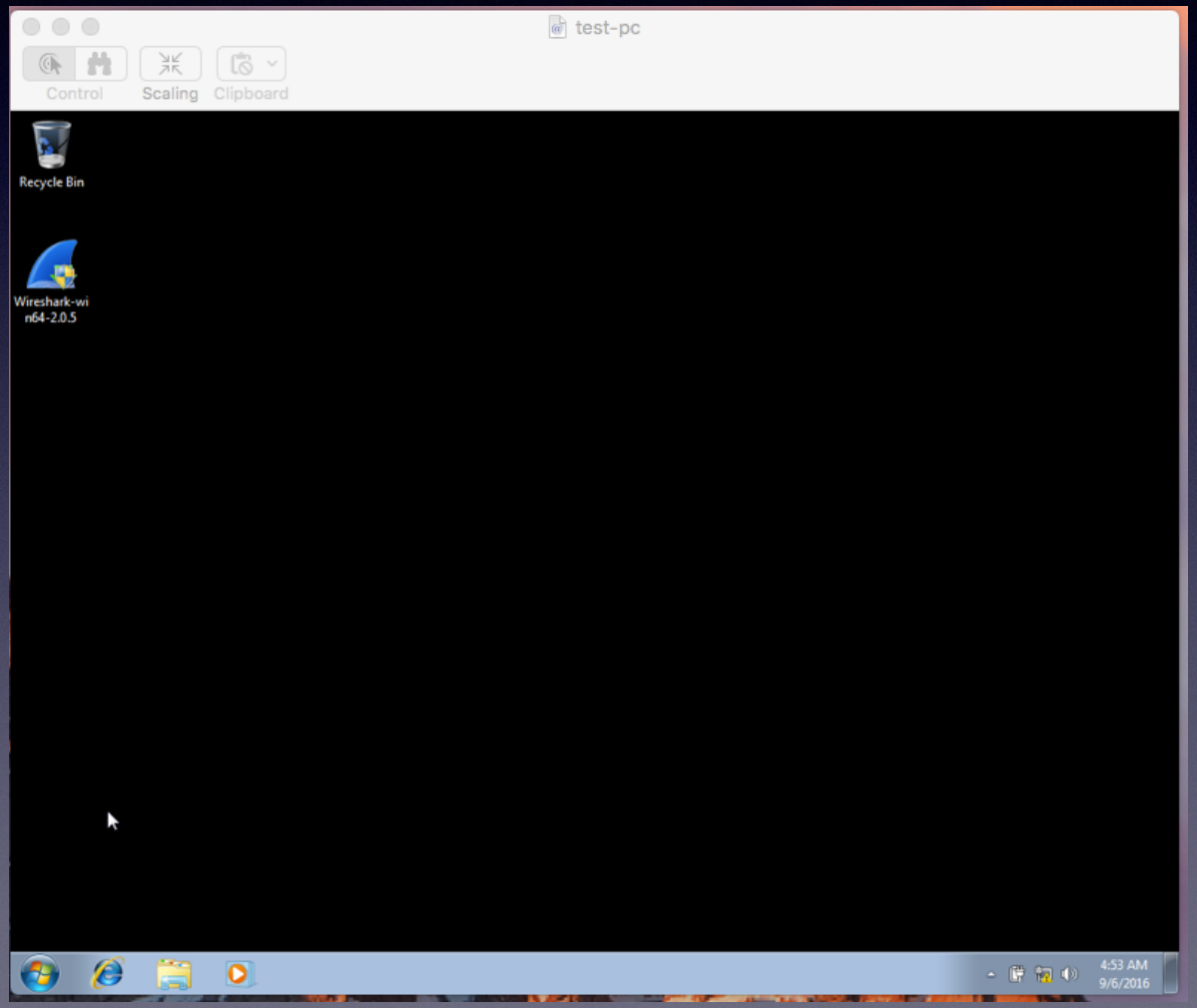
FOX IT

VNC protocol - frame messages

- After the initial handshake, the server sends a complete representation of the server's screen to the client
- One should be able to reconstruct a complete screenshot from the screen using this first message!

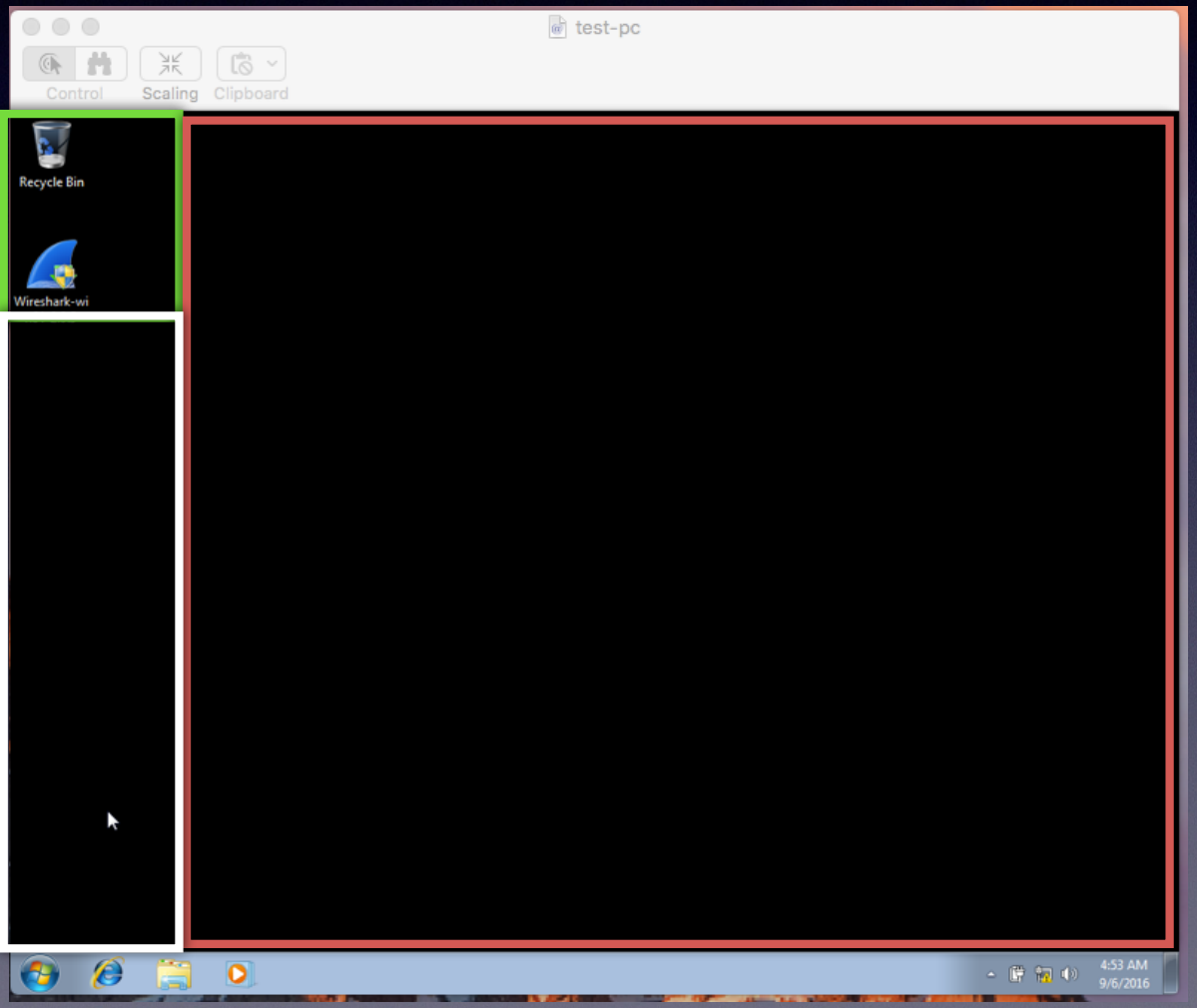
FOX IT

VNC protocol - frame messages



FOX IT

VNC protocol - frame messages



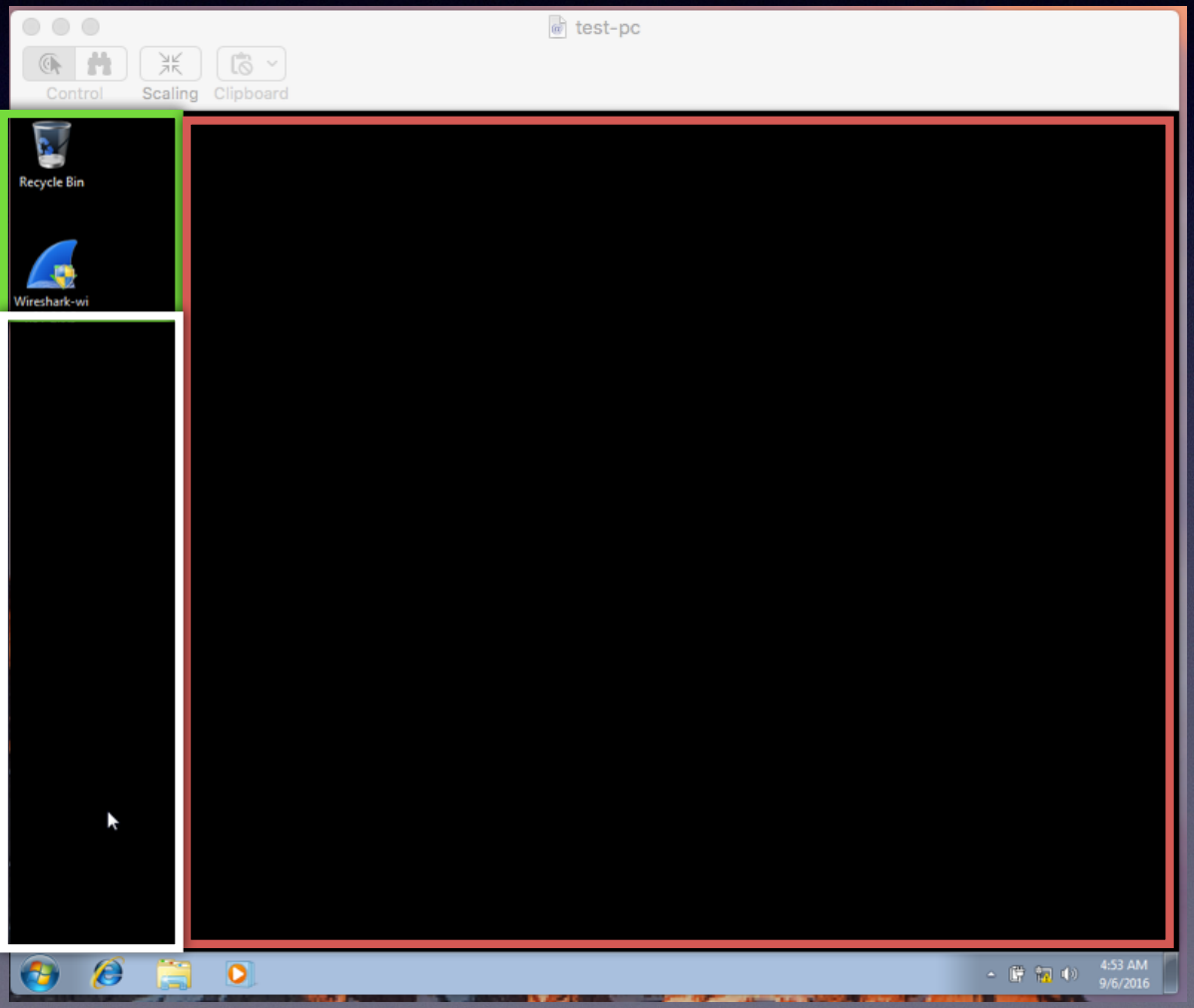
120x120

1160x960

120x840

FOX IT

VNC protocol - frame messages



120x120

1160x960

120x840

Compress & Encode

FOX IT

VNC protocol - frame messages

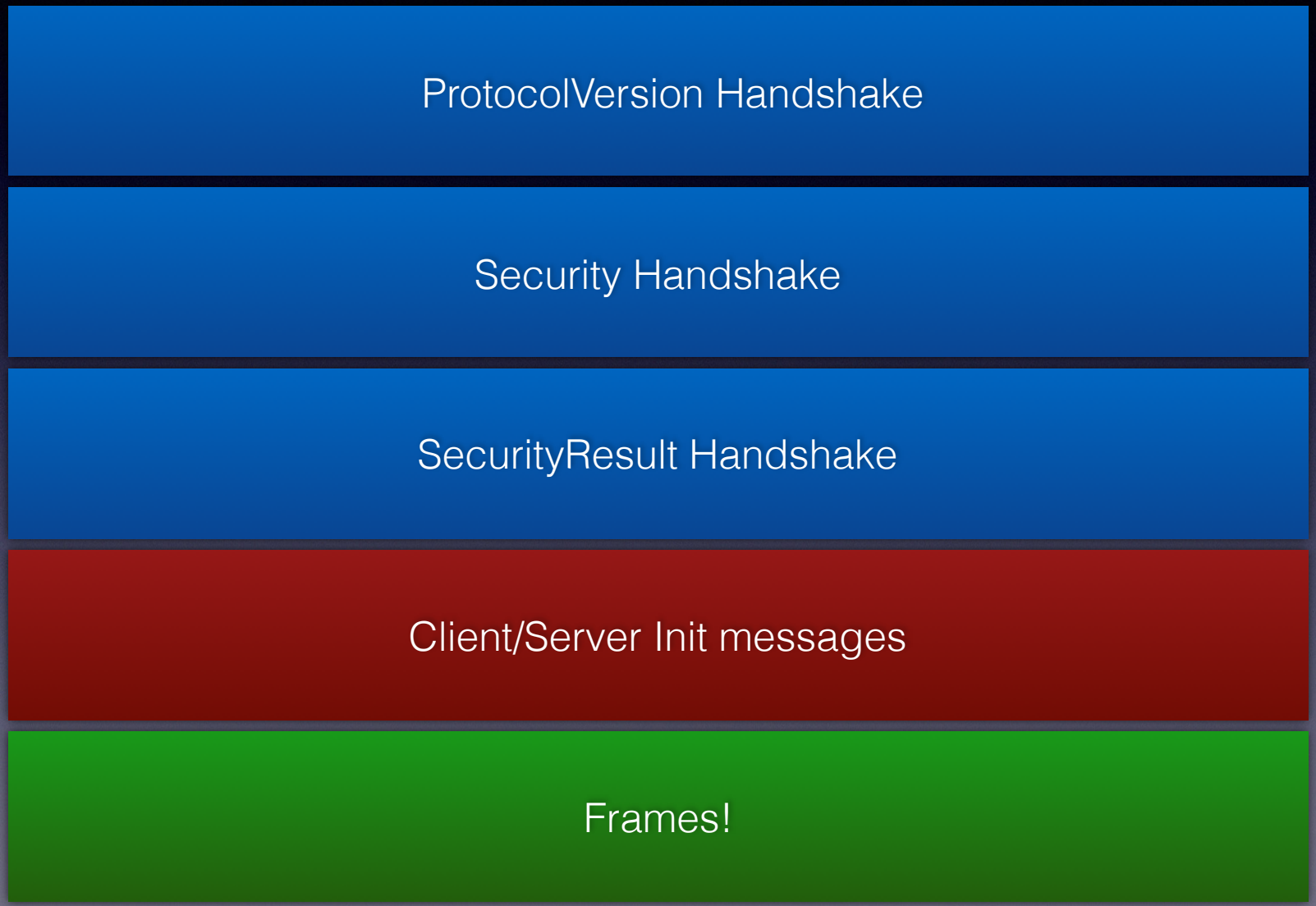
FOX IT

VNC protocol - frame messages

- Complete screen update first!
- Then: Client and Server can send messages at will:
  - containing keystrokes, mouse pointer movements, screen updates.
- For our purpose too much effort at this stage

FOX IT

VNC protocol - Recap



FOX IT

Bro Wishlist

- What would we want to see from a security perspective:
  - are there RFB servers in the network?
  - from where and when are they accessed, for how long?
  - which software is used?
  - what kind of authentication is used, was it successful?
  - ~~other useful information~~ Server name, screen dimensions?
- Bonus exercise: can we get a screenshot? 😇

FOX IT

Dev/test/deploy

FOX IT

Dev/test/deploy

- Ingredients for creating a protocol parser:
  - wireshark and loads of sample PCAPs
  - knowledge of BinPac and Bro policy writing
  - knowledge of the protocol (obviously)

FOX IT

Dev

BinPac (protocol parsing)

- Define events to emit
- Define protocol messages
- BinPac creates C++ parser

Scripts

- Define DPD to identify streams to process
- Connect events from parser to log output

Testing

- Create tests based on pcaps
- Supply suspected output of your parser

FOX IT

Dev - where to start

- documentation on-line
- learn from existing protocol parsers
- https://github.com/grigorescu/binpac_quickstart
  - creates some boilerplate code for you to get your parser up and running
- bro-dev mailinglist
  - great supportive community!

FOX IT

Dev - be prepared

- #1 - No matter how simple the protocol, there's always a catch
- #2 - No matter how well your protocol parser is, someone will always present you with a pcap that doesn't parse

FOX IT

Dev - be prepared

- #1 - No matter how simple the protocol, there's always a catch

FOX IT

BinPac (protocol parsing)

- Ideally, we would like to have something like this:

```
type RFB_PDU {
    header: "RFB";
    type: uint8;
    length_of_payload: uint16;
    payload: case of type {
        1-> client_version = RFBClientVersion;
        2-> authentication_msg = RFBAuthenticationMsg;
    } &length=length_of_payload;
};
```

- Each message self-descriptive (SMB!)

FOX IT

Dev

BinPac

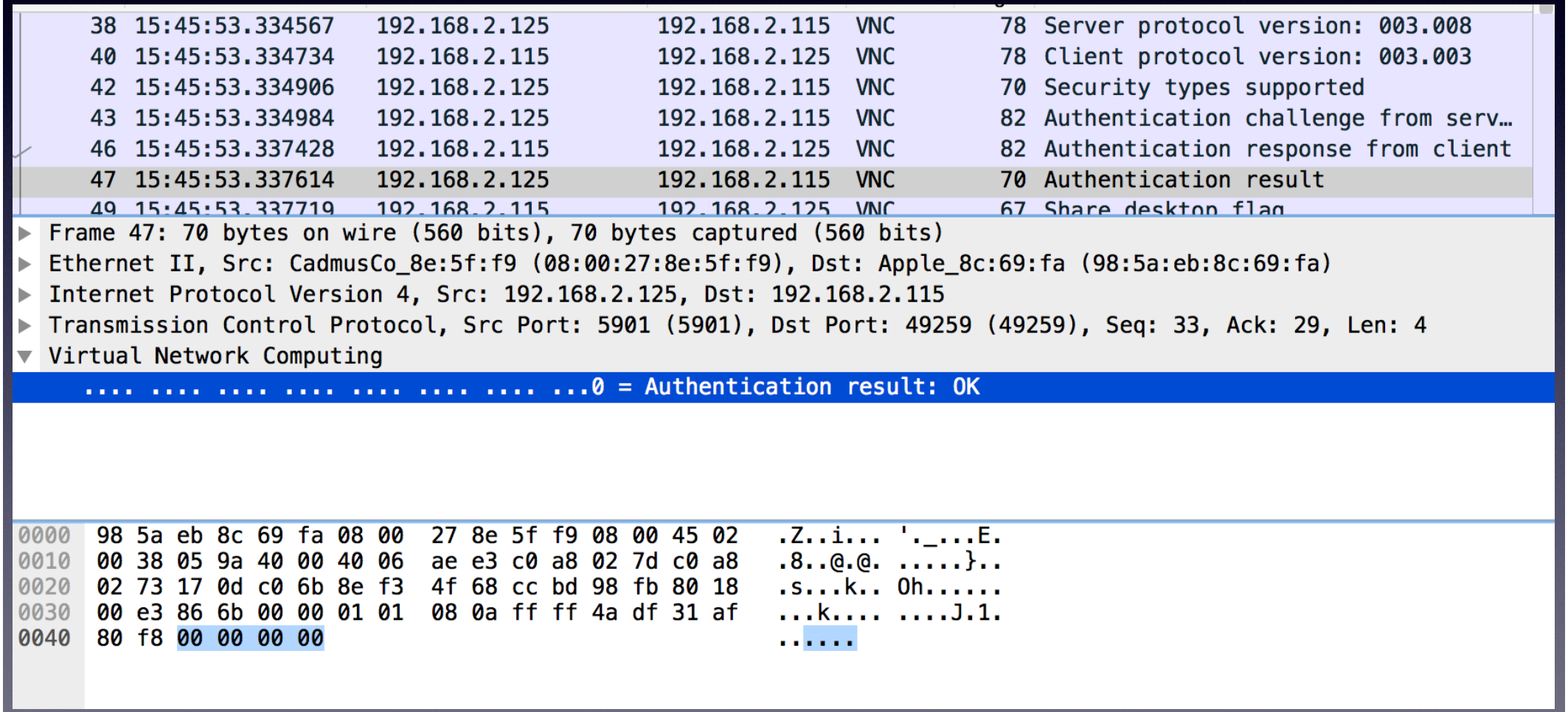
```
    38 15:45:53.334567   192.168.2.125        192.168.2.115  VNC        78 Server protocol version: 003.008
    40 15:45:53.334734   192.168.2.115        192.168.2.125  VNC        78 Client protocol version: 003.003
    42 15:45:53.334906   192.168.2.125        192.168.2.115  VNC        70 Security types supported
    43 15:45:53.334984   192.168.2.125        192.168.2.115  VNC        82 Authentication challenge from serv…
    46 15:45:53.337428   192.168.2.115        192.168.2.125  VNC        82 Authentication response from client
    47 15:45:53.337614   192.168.2.125        192.168.2.115  VNC        70 Authentication result
    49 15:45:53.337719   192.168.2.115        192.168.2.125  VNC        67 Share desktop flag
▶ Frame 42: 70 bytes on wire (560 bits), 70 bytes captured (560 bits)
▶ Ethernet II, Src: CadmusCo_8e:5f:f9 (08:00:27:8e:5f:f9), Dst: Apple_8c:69:fa (98:5a:eb:8c:69:fa)
▶ Internet Protocol Version 4, Src: 192.168.2.125, Dst: 192.168.2.115
▶ Transmission Control Protocol, Src Port: 5901 (5901), Dst Port: 49259 (49259), Seq: 13, Ack: 13, Len: 4
▼ Virtual Network Computing
    Security type: VNC (2)

0000  98 5a eb 8c 69 fa 08 00  27 8e 5f f9 08 00 45 02   .Z..i... '._...E.
0010  00 38 05 98 40 00 40 06  ae e5 c0 a8 02 7d c0 a8   .8..@.@. .....}..
0020  02 73 17 0d c0 6b 8e f3  4f 54 cc bd 98 eb 80 18   .s...k.. OT......
0030  00 e3 86 6b 00 00 01 01  08 0a ff ff 4a de 31 af   ...k.... ....J.1.
0040  80 f5 00 00 00 02                                 ......
```

FOX IT

Dev

BinPac



FOX IT

BinPac

Dev

- RFB messages do not contain e.g. a command identifier, or total size of the message
- How to interpret a set of bytes depends on the messages before it
- rfb-protocol-analyzer.pac implements state machine

FOX IT

State machine

BinPac

```
type RFB_PDU_request = record {
    request: case state of {
        AWAITING_CLIENT_BANNER -> version: RFBProtocolVersion(true);
        AWAITING_CLIENT_RESPONSE -> response: RFBVNCAuthenticationResponse;
        AWAITING_CLIENT_SHARE_FLAG -> shareflag: RFBClientInit;
        AWAITING_CLIENT_AUTH_TYPE_SELECTED37 -> authtype: RFBAuthTypeSelected;
        AWAITING_CLIENT_ARD_RESPONSE -> ard_response: RFBSecurityARDResponse;
        RFB_MESSAGE -> ignore: bytestring &restofdata &transient;
        default -> data: bytestring &restofdata &transient;
    } &requires(state);
    } &let {
        state: uint8 = $context.connection.get_state(true);
};

type RFB_PDU_response = record {
    request: case rstate of {
        AWAITING_SERVER_BANNER -> version: RFBProtocolVersion(false);
        AWAITING_SERVER_AUTH_TYPES -> auth_types: RFBSecurityTypes;
        AWAITING_SERVER_AUTH_TYPES37 -> auth_types37: RFBSecurityTypes37;
        AWAITING_SERVER_CHALLENGE -> challenge: RFBVNCAuthenticationRequest;
        AWAITING_SERVER_AUTH_RESULT -> authresult : RFBSecurityResult;
        AWAITING_SERVER_ARD_CHALLENGE -> ard_challenge: RFBSecurityARDChallenge;
        AWAITING_SERVER_PARAMS -> serverinit: RFBServerInit;
        RFB_MESSAGE -> ignore: bytestring &restofdata &transient;
        default -> data: bytestring &restofdata &transient;
    } &requires(rstate);
    } &let {
        rstate: uint8 = $context.connection.get_state(false);
};
```

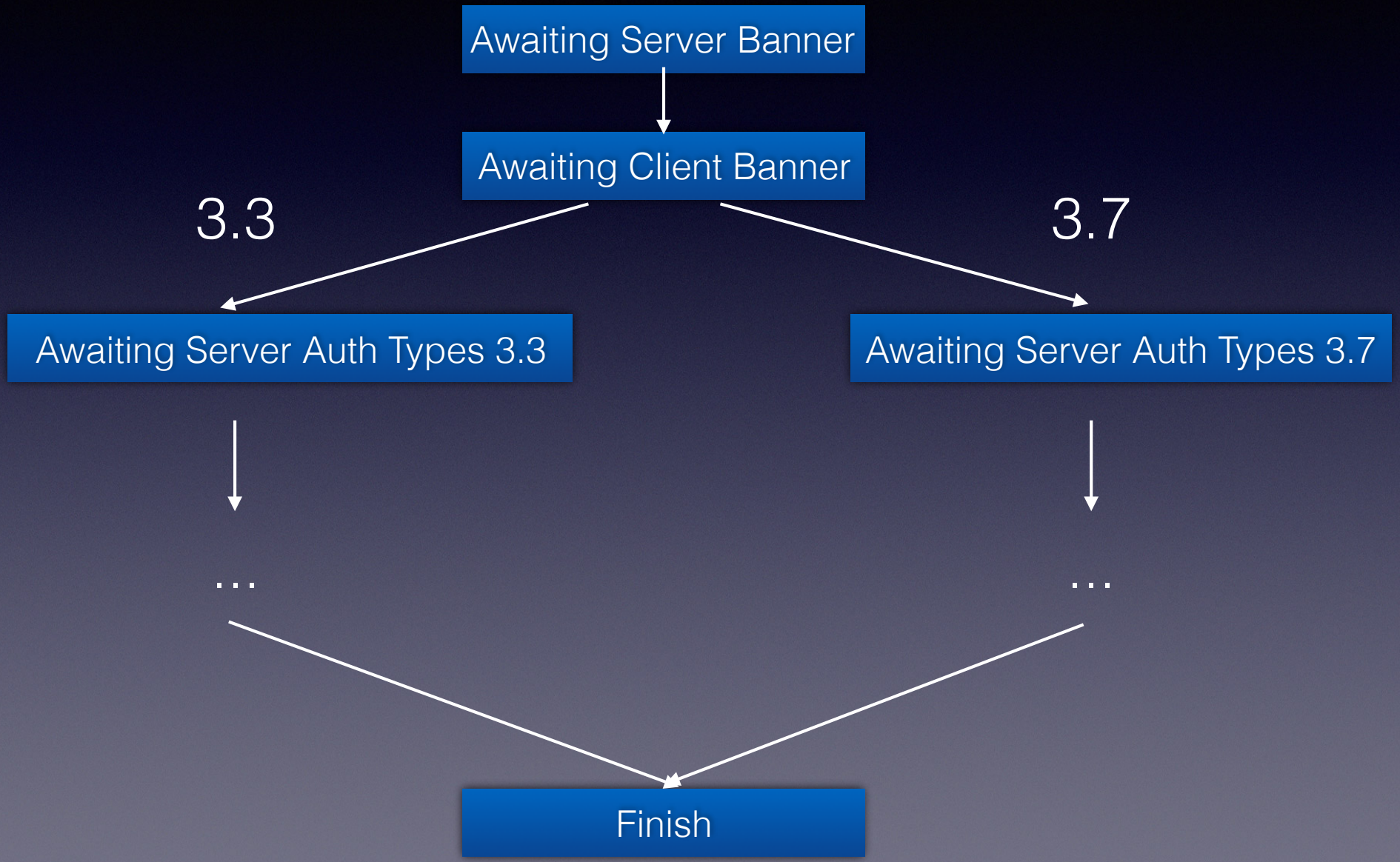
‘state’ - defines step in our protocol.

After successfully parsing a message, ‘state’ gets updated accordingly.

FOX IT

State machine

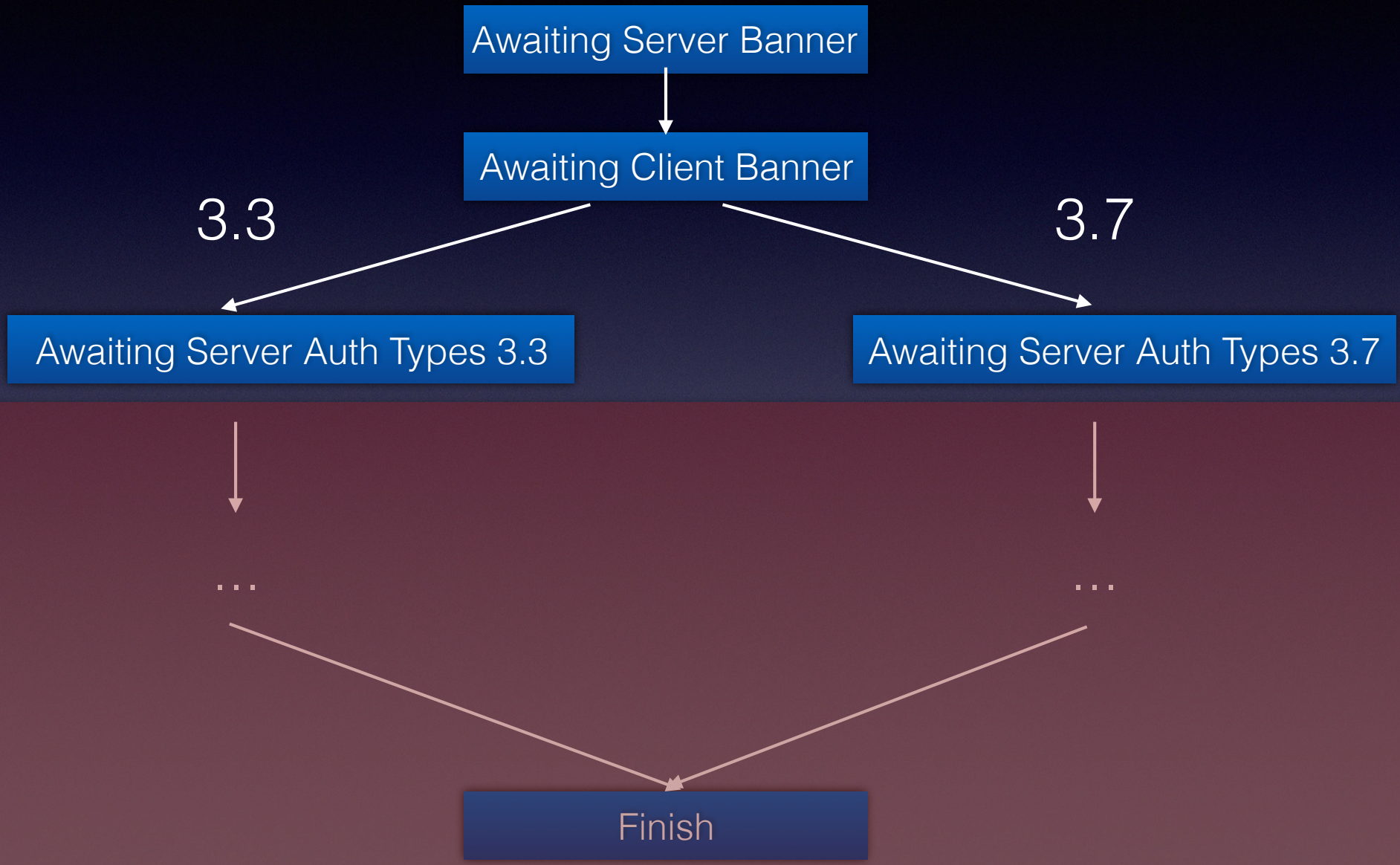
BinPac



FOX IT

State machine

BinPac



FOX IT

Dev - be prepared

- #2 - No matter how well your protocol parser is, someone will always present you with a pcap that doesn't parse

FOX IT

BinPac

Reality...

- Many different dialects, custom features and specific implementations hamper parsing
- E.g. custom authentication protocols

FOX IT

Scripts

Dev - events

```
event rfb_event%(c: connection%);

event rfb_authentication_type%(c: connection, authtype: count%);

event rfb_auth_result%(c: connection, result: bool%);

event rfb_share_flag%(c: connection, flag: bool%);

event rfb_client_version%(c: connection, major_version: string, minor_version: string%);

event rfb_server_version%(c: connection, major_version: string, minor_version: string%);

event rfb_server_parameters%(c: connection, name: string, width: count, height: count%);
```

Logical breakdown
of events

FOX IT

Dev - DPD

Scripts

- Supply DPD signature

```
signature dpd_rfb_server {
    ip-proto == tcp
    payload /^RFB/
    requires-reverse-signature dpd_rfb_client
    enable "rfb"
}

signature dpd_rfb_client {
    ip-proto == tcp
    payload /^RFB/
    tcp-state originator
}
```

FOX IT

Testing

Dev - test

- Test framework allows you to submit a sample pcap with expected output for (regression) testing

```
./testing/btest/Baseline/scripts.base.protocols.rfb.vnc-mac-to-linux/rfb.log
./testing/btest/Traces/rfb/vnc-mac-to-linux.pcap
./testing/btest/scripts/base/protocols/rfb/vnc-mac-to-linux.test
```

FOX IT

Testing

Dev - test

- Simple test:

```
# @TEST-EXEC: bro -C -r $TRACES/rfb/vnc-mac-to-linux.pcap
# @TEST-EXEC: btest-diff rfb.log

@load base/protocols/rfb
```

- Execute:

```
mbp-retina:btest mhens$ ../../aux/btest/btest scripts/base/protocols/rfb/*
[  0%] scripts.base.protocols.rfb.rfb-apple-remote-desktop ...
[ 50%] scripts.base.protocols.rfb.vnc-mac-to-linux ...
all 2 tests successful
```

FOX IT

Dev

BinPac (protocol parsing)

Scripts

Testing

- Deploy!

FOX IT

Dev - deploy

```
martin@martin-VirtualBox:~/bro$ bro -C -r testing/btest/Traces/rfb/vnc-mac-to-linux.pcap
martin@martin-VirtualBox:~/bro$ head rfb.log
#separator \x09
#set_separator   ,
#empty_field    (empty)
#unset_field    -
#path   rfb
#open   2016-09-11-17-08-04
#fields ts      uid     id.orig_h       id.orig_p       id.resp_h       id.resp_p       c
lient_major_version    client_minor_version server_major_version    server_minor_version
    authentication_method   auth    share_flag      desktop_name width   height
#types  time    string  addr    port    addr    port    string  string  string  string  s
tring  bool    bool    string      count   count
1459093553.334734       Cba8ke1TukW2T8C6qa      192.168.2.115   49259   192.168.2.125   5
901    003     003     003 008     VNC     T       T       root's X desktop (martin-Virt
ualBox:1)  1024    768
1459093548.745805       C7d7UK31LWnc0S2XPa      192.168.2.115   49256   192.168.2.125   5
901    003     003     003 008     VNC     -       -       -       -       -
martin@martin-VirtualBox:~/bro$
```

FOX IT

Dev - deploy

ts	1459093553.334734
uid	Cfyr7o4lZoenYZZNG6
id.orig_h	192.168.2.115
id.orig_p	49259
id.resp_h	192.168.2.115
id.resp_p	5901
client_major_version	003
client_minor_version	003
server_major_version	003
server_minor_version	008
authentication_method	VNC
auth	T
share_flag	T
desktop_name	root's X desktop (martin-VirtualBox:1)
width	1024
height	768

FOX IT

Dev - deploy

- What would we want to see from a security perspective:
  - are there RFB servers in the network?
  - from where and when are they accessed, for how long?
  - which software is used?
  - what kind of authentication is used, was it successful?
  - Server name, screen dimensions?
- Bonus exercise: can we get a screenshot? 😇

FOX IT

Dev - deploy

- Are there RFB servers in the network?
- bro-cut id.resp_h < rfb.log | sort | uniq

FOX IT

Dev - deploy

- From where and when are RFB servers accessed, for how long?
- bro-cut -d ts id.orig_h id.resp_h service duration < conn.log | grep rfb

```
$ bro-cut -d ts id.orig_h id.resp_h service duration < conn.log | grep rfb
2016-03-27T17:45:51+0200 192.168.2.115 192.168.2.125   rfb 1.775081
2016-03-27T17:45:53+0200 192.168.2.115 192.168.2.125   rfb 2.778796
2016-03-27T17:45:48+0200 192.168.2.115 192.168.2.125   rfb 2.813754
```

FOX IT

Dev - deploy

- Which software is used?
- bro-cut client_major_version client_minor_version < rfb.log | sort | uniq -c | sort -nr
- bro-cut server_major_version server_minor_version < rfb.log | sort | uniq -c | sort -nr
- Look for server/client versions: e.g. 3.889 = most likely Apple Remote Desktop

FOX IT

Dev - deploy

- What kind of authentication is used, was it successful?
- bro-cut id.resp_h authentication_method auth < rfb.log

```
$ bro-cut id.resp_h authentication_method auth  < rfb.log
192.168.2.125  VNC  T
192.168.2.125  VNC  F
192.168.2.125  VNC  -
```

FOX IT

Dev - deploy

- What kind of Server name, screen dimensions are used, was the connection exclusive?
- bro-cut id.resp_h desktop_name name width height share_flag < rfb.log

```
$ bro-cut id.resp_h desktop_name name width height share_flag < rfb.log
192.168.2.125  root's X desktop (martin-VirtualBox:1)    1024   768    T
192.168.2.125  -    -  -  -
192.168.2.125  -    -  -  -
```

FOX IT

Recap

- We have seen **why** it is interesting to parse RFB
- We have seen **how** RFB works and what information we can get from parsing the protocol
- We have seen **what** steps to take to build and test a protocol parser
- We have seen **how** we can answer our research questions

FOX IT

Recap

- First version of RFB parser commit:

```
commit 849875e8be73d0e0b5a6ebca74ed56fdabba464b
Author: Martin van Hensbergen <martin.vanhensbergen@fox-it.com>
Date:   Mon Apr 11 10:35:00 2016 +0200

    Analyzer and bro script for RFB protocol (VNC)

    This analyzer parses the Remote Frame Buffer
    protocol, usually referred to as the 'VNC protocol'.

    It supports several dialects (3.3, 3.7, 3.8) and
    also handles the Apple Remote Desktop variant.

    It will log such facts as client/server versions,
    authentication method used, authentication result,
    height, width and name of the shared screen.

    It also includes two testcases.

    Todo: Apple Remote Desktop seems to have some
    bytes prepended to the screen name. This is
    not interepreted correctly.
```

- Will be in 2.5 release

FOX IT

Future work

- Handle different dialects/authentication types/ implementations (pcaps welcome!)
- TLS over VNC support
- Generating screenshot files from initial screen update 😎
- martin.vanhensbergen@fox-it.com

FOX IT

Thanks

- Thanks for listening!