VoIP in Flow
A Beginning

Nathan Dell
CERT/NetSA

Legal

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM-0000773

Outline

- General Principals to Finding and Understanding VoIP in flow
- Analysis of VoIP Real Time Communications
- Lab Example of Data Exfiltration

General Principals Finding VoIP

- Standard VoIP have well known ports and protocols
  - SIP uses port 5060, secure SIP uses port 5061
  - SCCP uses port 2000, secure SCCP uses port 2443
  - H. 323 uses port 1720 for call set up
- Finding Real Time traffic (phone call)
  - Session generated port pair for real time traffic (RTP, RTCP)
  - Work backwards follow VoIP signaling
  - SIP – Phone are required to periodically register with the registration server
  - Find the a particular end port and pull traffic until you see a phone call

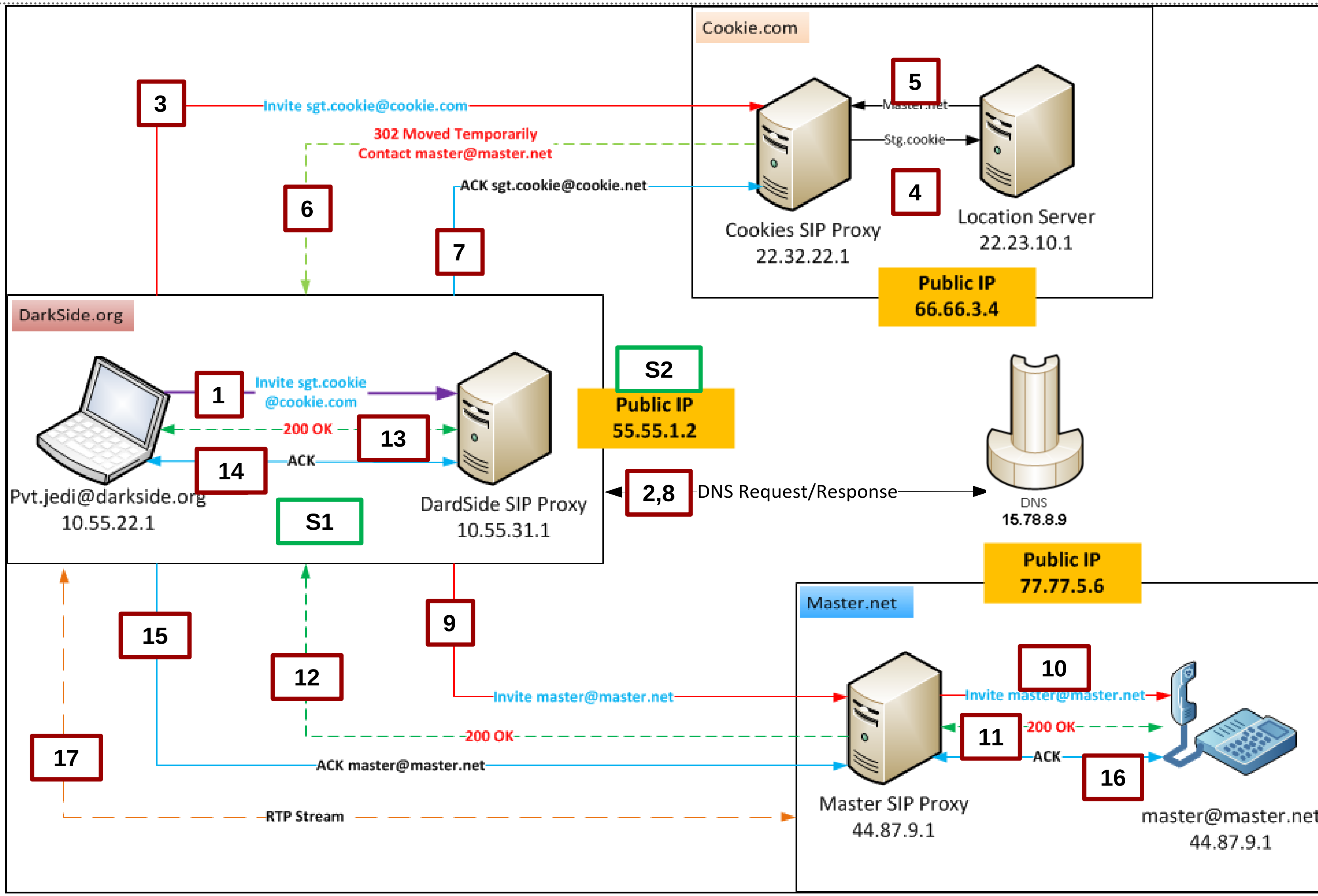
General Principals Finding VoIP

- None standard VoIP have unique signaling protocols and ports
  - Skype
    - Authentication and signaling over port 80 or 443 (TLS)
    - Use common network elements to locate traffic
    - Authentication Server is a static IP
  - Razer Comm Gaming VoIP Software
    - Authentication Ports 443 (TLS)
    - Use UDP Port 2000 for media sessions
    - Does not operate in peer-to-peer method uses a transcoder through a single IP

General Principals Understand VoIP

- Understanding the function of common VoIP network elements can help you understand observed VoIP traffic
  - **Redirect server** – allows phone call forwarding
  - **Transcoder** – Allow phone with unsupported codex the ability to communicate
  - **Mixer** – Combines two audio streams into a single audio stream
  - **User Agents** - entities within the VoIP network in a client server mode that act on the behalf of users (phone, PC, Conference Bridge)
  - **Location Server** – Maintain location of connected phones

SIP Redirect Server – Call Forwarding



SIP Redirect Flow – Call Forwarding

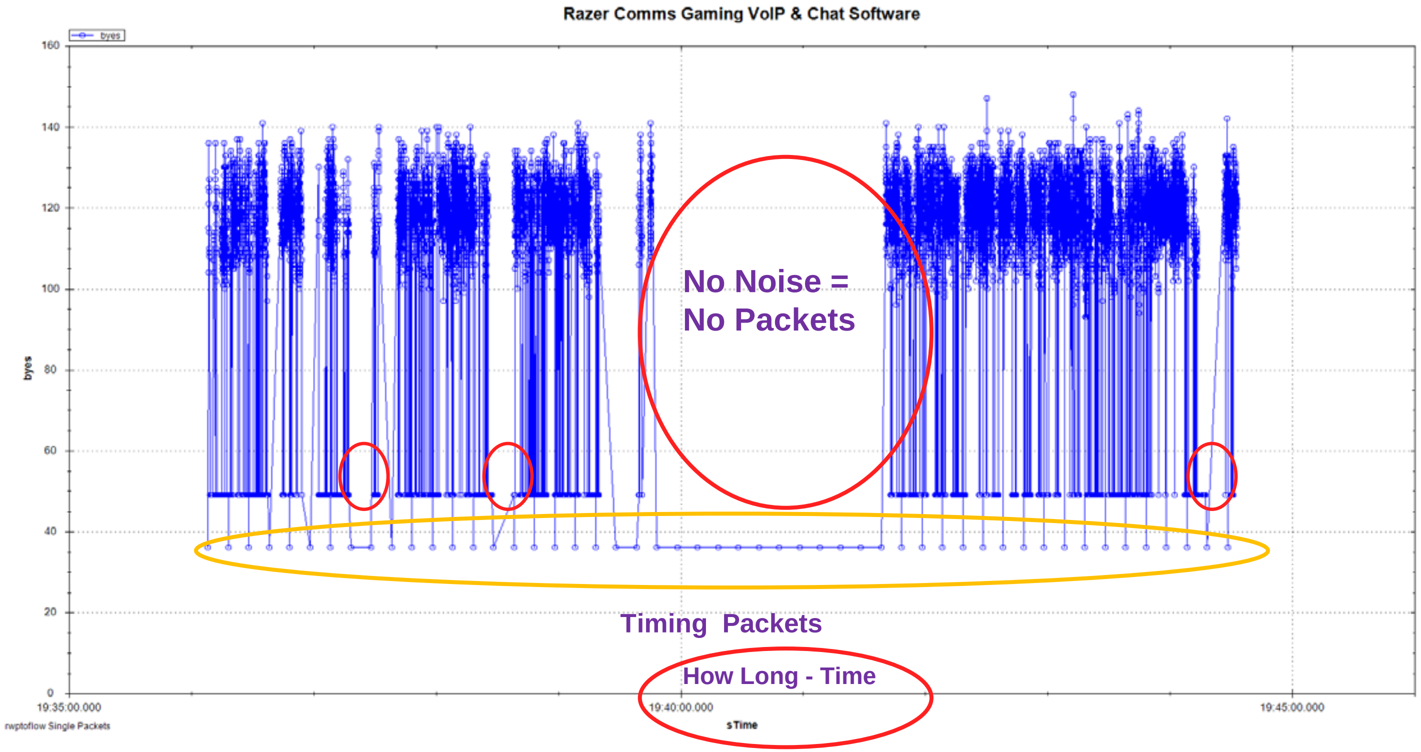
sIP	dIP	sPort	dPort	Pro	Packets	Bytes	Sensor	Type
10.55.22.1	10.55.31.1	17756	5060	17	26	10062	S1	OUT
55.55.1.2	15.78.8.9	37895	53	17	1	88	S2	OUT
15.78.8.9	55.55.1.2	53	37895	17	1	245	S2	IN
55.55.1.2	66.66.3.4	24868	5060	6	15	5175	S2	OUT
66.66.3.4	55.55.1.2	5060	24868	6	10	3450	S2	IN
55.55.1.2	16.58.43.6	37895	53	17	1	88	S2	OUT
16.58.43.6	55.55.1.2	53	37895	17	1	245	S2	IN
55.55.1.2	77.77.5.6	13467	5060	6	34	12852	S2	OUT
77.77.5.6	55.55.1.2	5060	13467	6	24	8328	S2	IN
10.55.31.1	10.55.22.1	5060	17756	6	34	13158	S1	IN
10.55.22.1	77.77.5.6	47598	35878	17	4876	277932	S1	OUT
55.55.1.2	77.77.5.6	47598	35878	17	4876	277932	S2	OUT
77.77.5.6	55.55.1.2	33572	47598	17	1625	92644	S2	IN
55.55.1.2	10.55.22.1	33572	47598	17	1625	92644	S1	IN
10.55.22.1	10.55.31.1	34527	5060	17	7	2275	S1	OUT
55.55.1.2	77.77.5.6	12354	5060	6	15	2646	S2	OUT
77.77.5.6	55.55.1.2	5060	12354	6	6	2088	S2	IN
10.55.31.1	10.55.22.1	5060	34527	17	8	2784	S1	IN

Real Time Communications

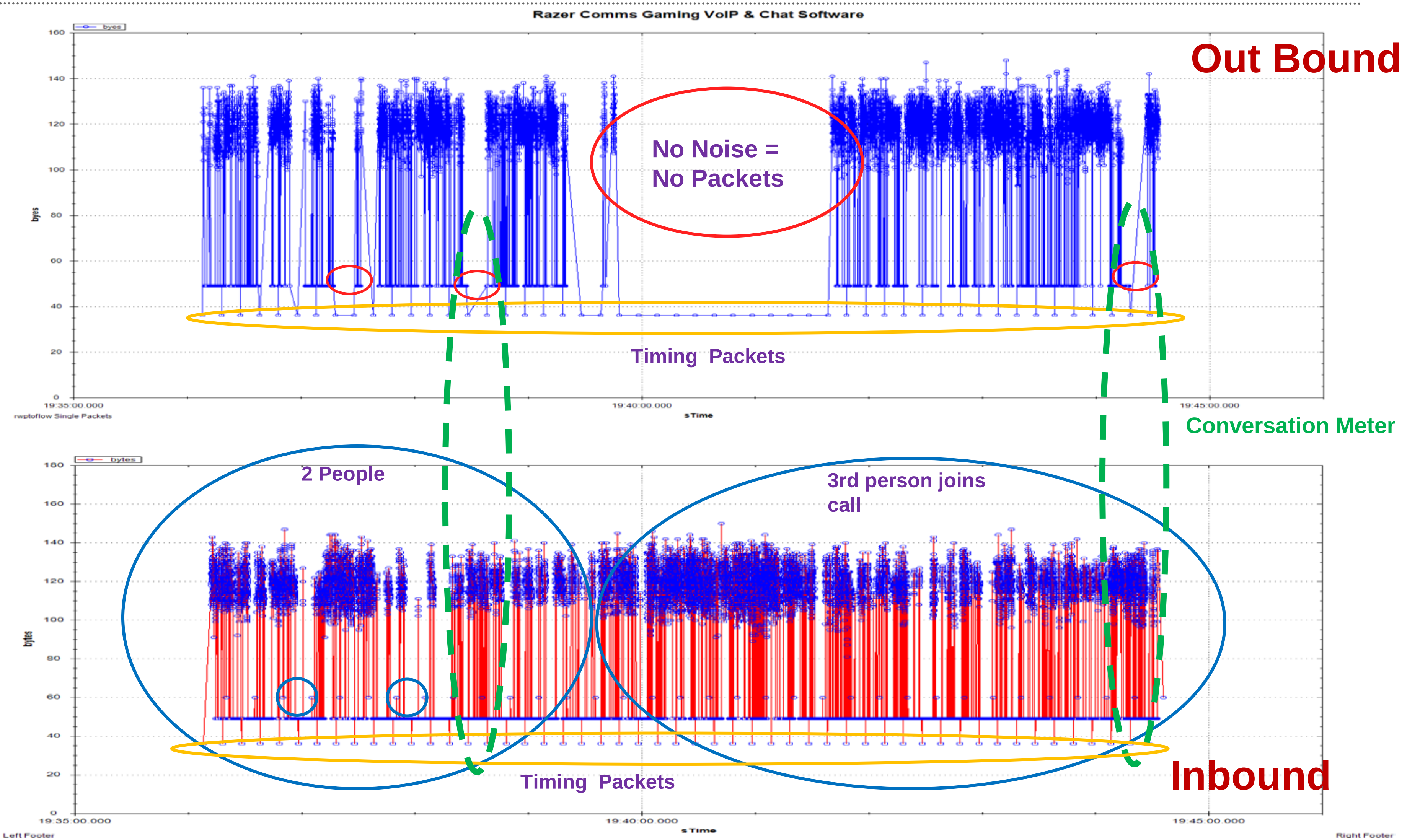
- Two use cases for packet generation:
  - On demand - noise is registered on the input devices (i.e microphone) then packaged and sent
  - Software continuously generate packets regardless of input, providing flow obfuscation
- Typically session generated port pair is used
  - Some software allows you specify port ranges
  - RTP should be even, RTCP next open odd port
  - Some software use dedicated ports for communication

Noise Generated Packets

- With a low idle time out on the flow sensor you can see:
  - General meter of the conversation
  - Who talks when and for how long

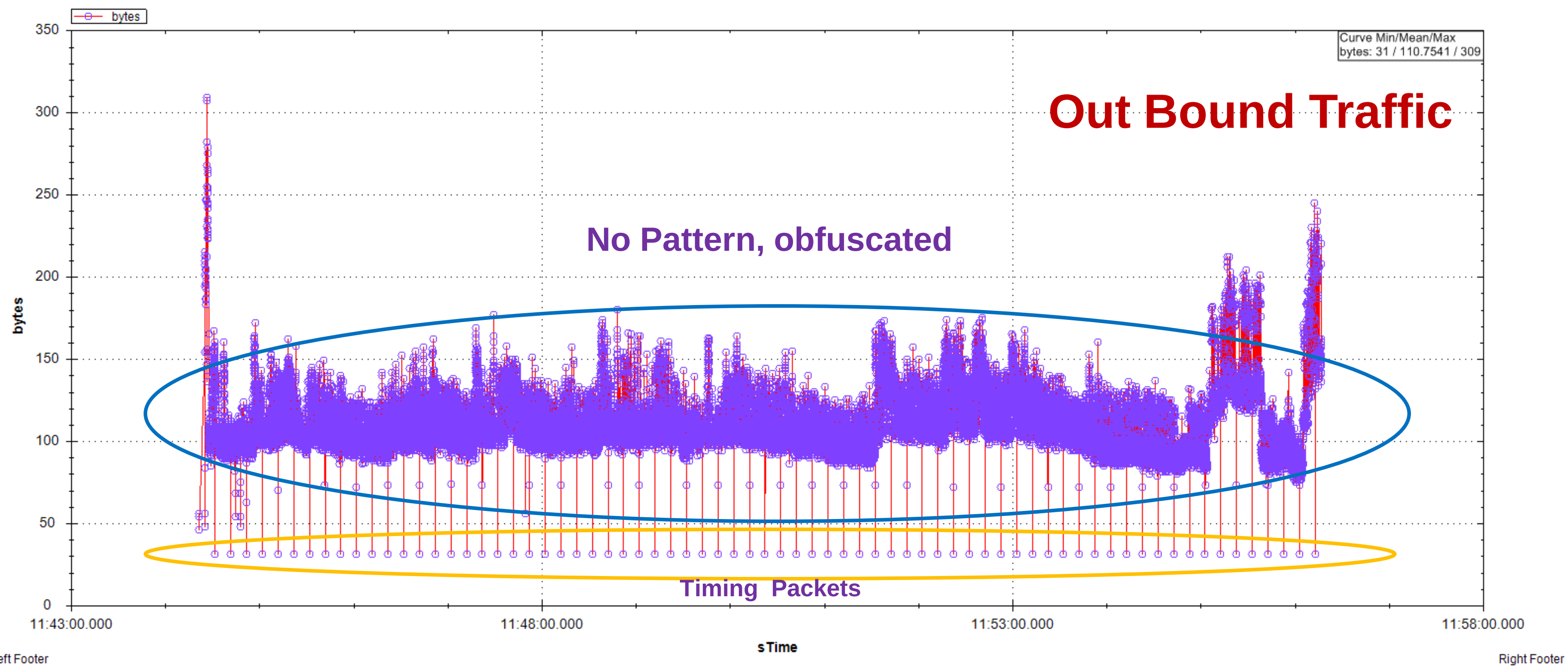


Noise Generated Packets

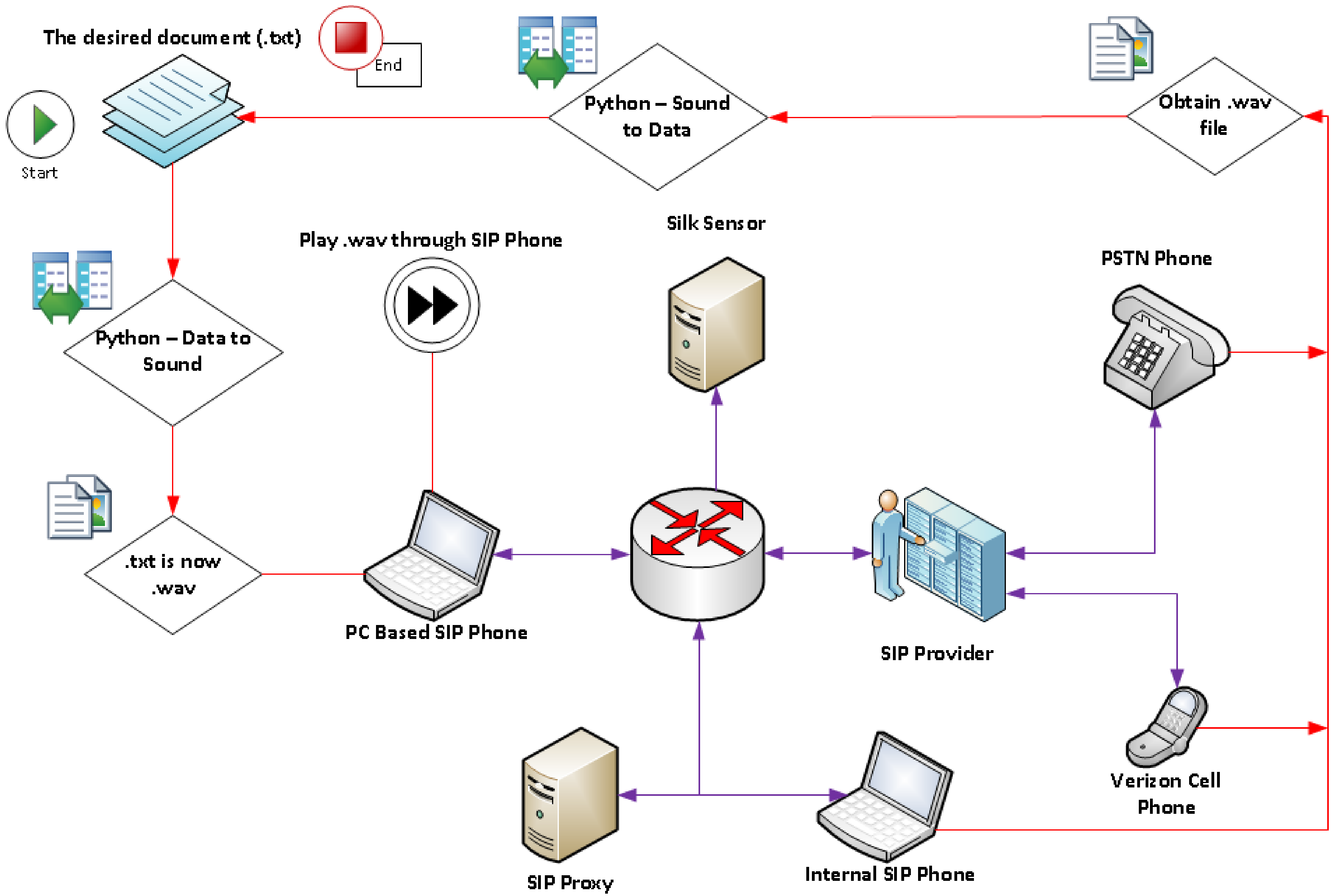


Software Generated Packets

- No break in the conversation regardless of human interaction
- Software packets appear to be uniformed



Data Exfiltration Lab – Work Flow



Lab: Data Exfiltration - Data to Sound

```
def to sound(infile, outfile):
        sampleFreq = 8000
        denominator = 8
        frequencies = (262, 294, 330, 350, 392, 440, 494, 523,
587, 659, 698, 784, 880, 988, 1047, 1175)

        size = os.path.getsize(infile)

        fmt = FormatChunk()
        fmt.size = 0x10
        fmt.formatTag = 1
        fmt.channels = 1
        fmt.bitsPerSample = 16
        fmt.samplesPerSec = sampleFreq
        fmt.avgBytesPerSec = fmt.samplesPerSec *
fmt.bitsPerSample / 8
        fmt.blockAlign = 4

        amplitude = 3000
        length = (3*denominator + size) * fmt.samplesPerSec  /
denominator * fmt.channels * fmt.bitsPerSample / 8
```

https://code.google.com/p/data-sound-poc/

- A file is already in a digital format but it must be passed through a VoIP network
- Convert the digital format (.txt) into audio format (.wav)
- Think Fax/Modem

Compression is Important

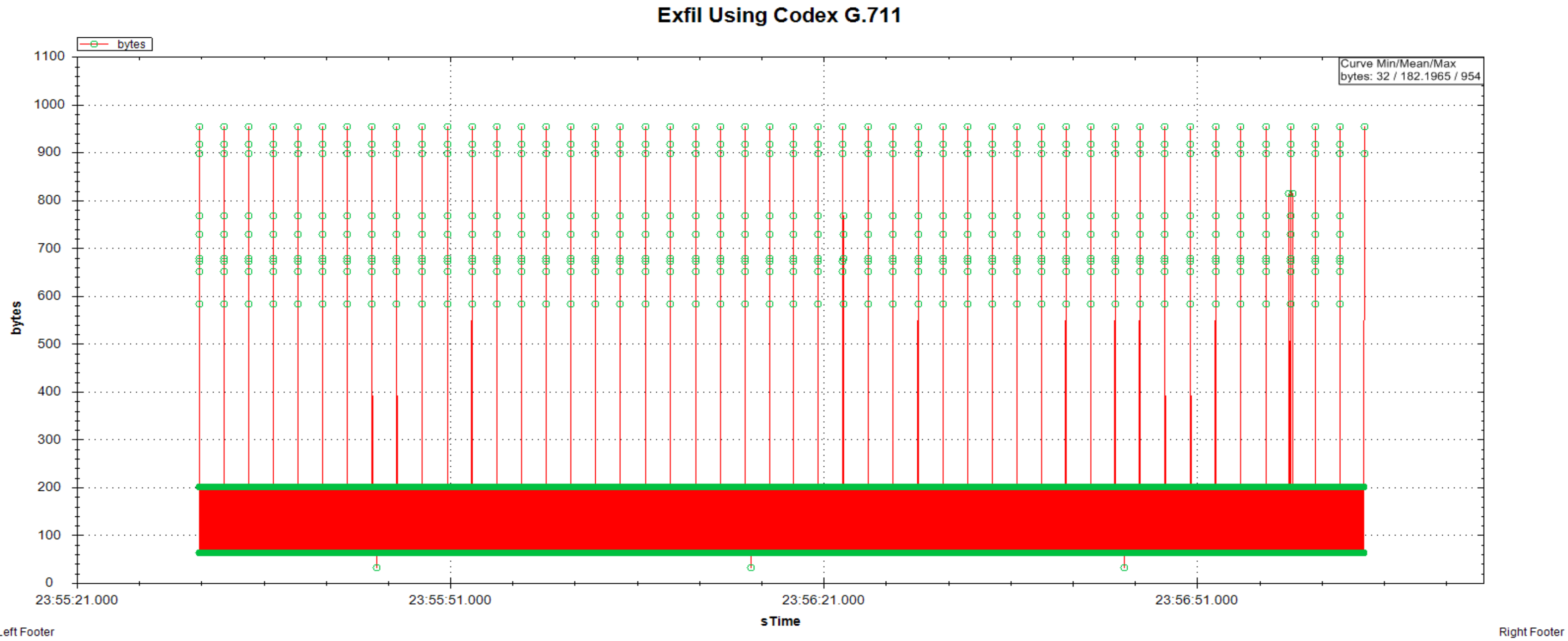
```
halfByteSampleLength = fmt.samplesPerSec / denominator

        j = 0
        while True:
                data = infh.read(1)
                if data == "":
                        break

                (m,) = struct.unpack("B", data)

                a = m & 0xF
                b = m >> 4

                for halfbyte in (a,b):
                        frequency = frequencies[halfbyte]
```

- To extract 23 Bytes it takes
  - 5.87 second sound bite
  - 136.8 KB .wav file
- To extract “CERT Resilience Management Module” or 52,691 Bytes
  - 1H 49M sound bite
  - 201MB .wav file

https://code.google.com/p/data-sound-poc/

Data Exfiltration - Flow

- Without good compression flow long
- The flow very uniformed, set frequency and timing
- The flow is unidirectional
  - Software generated packets obfuscated this flow, instead you will find bidirectional flow



Summary

- Understanding the network function of a object can help you understand observer flow data
  - Many comment elements between VoIP deployments (Proxy, Redirect, Voice Mail, Transcoder, etc.)
- Their two use cases for real time communication
  - On demand generation – noise = packets
  - Software generated packets - endless packets
- Data Exfiltration
  - Unidirectional flow, unless software generated packets
  - Weak Compression will create long flows
  - On demand VoIP will look like SW VoIP

Future Work

- Prove that there are only two use cases for real time communications
- Software Generated
  - Discover observable features
- On demand Generation
  - Developing a comprehensive list of observable features
  - Cross section those features with flow idle time out
    - Example: If you want the ability to observer feature X you will need an idle time of Y

Questions/comments?