LISTENING BY SPEAKING

(AN UNDER-ESTIMATED SECURITY ATTACK ON MEDIA GATEWAYS AND RTP RELAYS)

ECHO $USER

About Sandro Gauci:

- Behind Enable Security GmbH
- We do Pentests!
- VoIP / RTC / Network Infrastructure / Web Application / Software security testing
- Amongst this audience, I'm known for SIPVicious

ON RTCFUZZ

- Collaboration between Alfred Farrugia and myself, an Enable Security project
- In our spare/research time: we have started an internal project called RTC fuzz
- Making use of fuzzing techniques, i.e. mutated input fed to target code and observing behaviour
- Using well-known tools like AFL and Radamsa together with our custom tools
- Reporting back to the community

ON RTCFUZZ

So far:

- PJSIP (2 findings)
- Asterisk (1 finding separate from PJSIP)
- FreeSWITCH (1 finding)
- Kamailio (nothing yet)

ON SIPVICIOUS PRO

https://sipvicious.pro

Allows you to test I'm about to describe

AGENDA

- We will explain what this is NOT about
- Explain the attack itself and how it came about
- Show what can be done with this attack

AGENDA

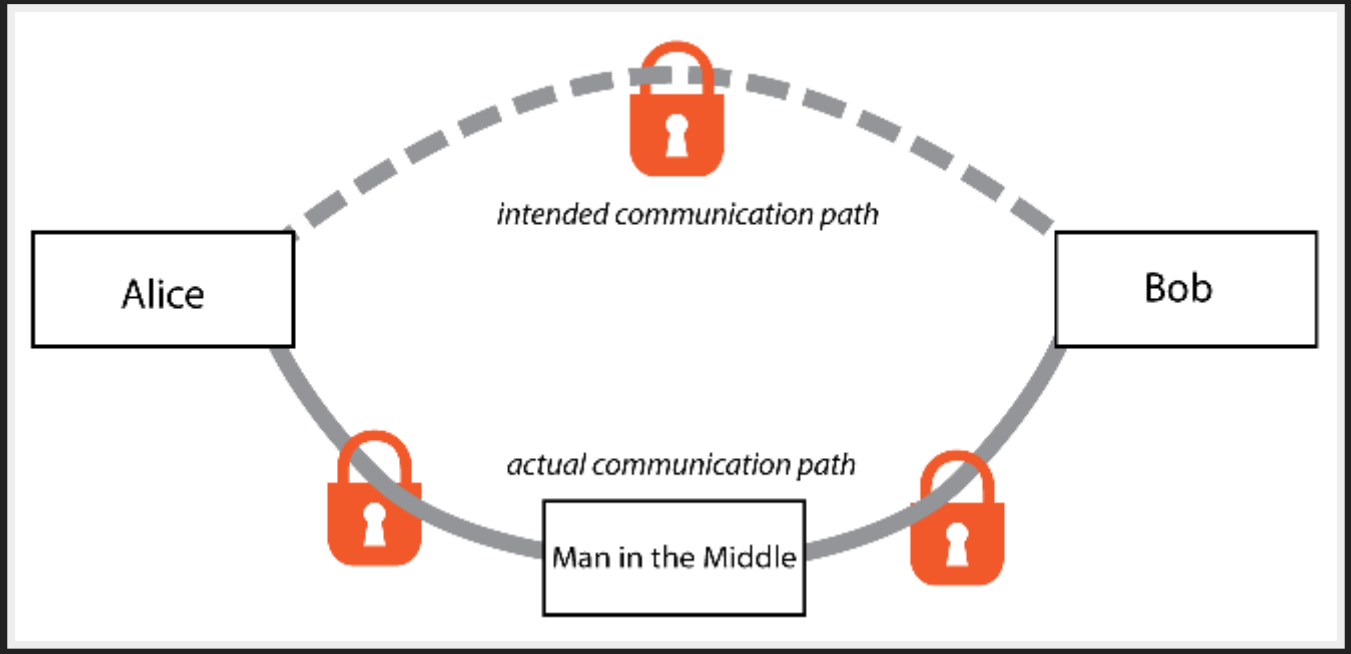
- Explain how it affects traditional media gateways and RTP proxies
- Touch upon how it applies to WebRTC and SRTP
- Discuss solutions that we have looked at
- Give an brief summary our findings in OSS
- Q&A
- Weissbeer

PURPOSE

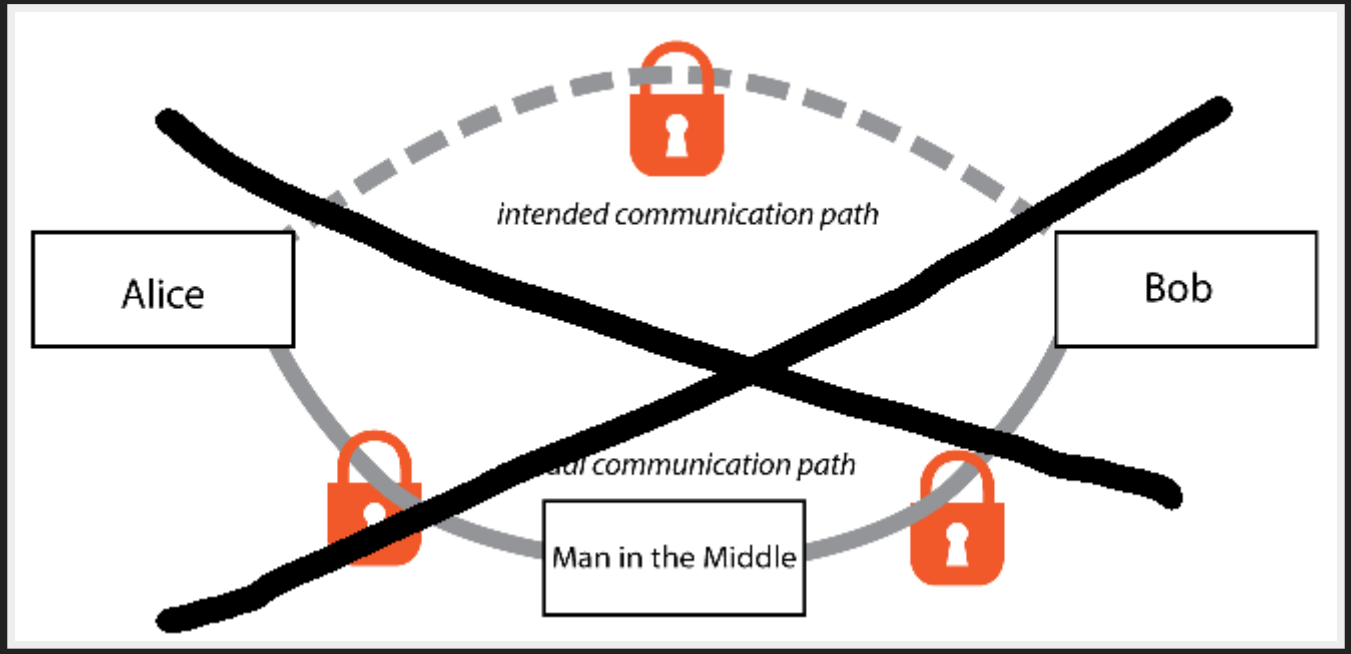
- This room is full of experts
- Will not be explaining how RTP works
- Start a conversation, get feedback and discuss
- and naturally ...

BEFORE WE DESCRIBE THE ATTACK ...

NO NEED FOR MAN-IN-THE-MIDDLE



(image taken from benthamsgaze blog)



MITM not required

NO MITM REQUIRED

- Our RTP attack does not require the attacker to be MITM
- But it can be used to create a man-in-the-middle situation
- Additionally, it has other security implications for RTC systems

EXPLAINING RTP HIJACK/INJECT/BLEED

ABOUT RTP PROXIES

- Have the job of defying the limitations of NAT
- Sometimes also have the *benefit* of allowing calls to be recorded
- ... or intercepted
- In various cases, the both of the above are a requirement

HOW DOES IT ACTUALLY WORK?

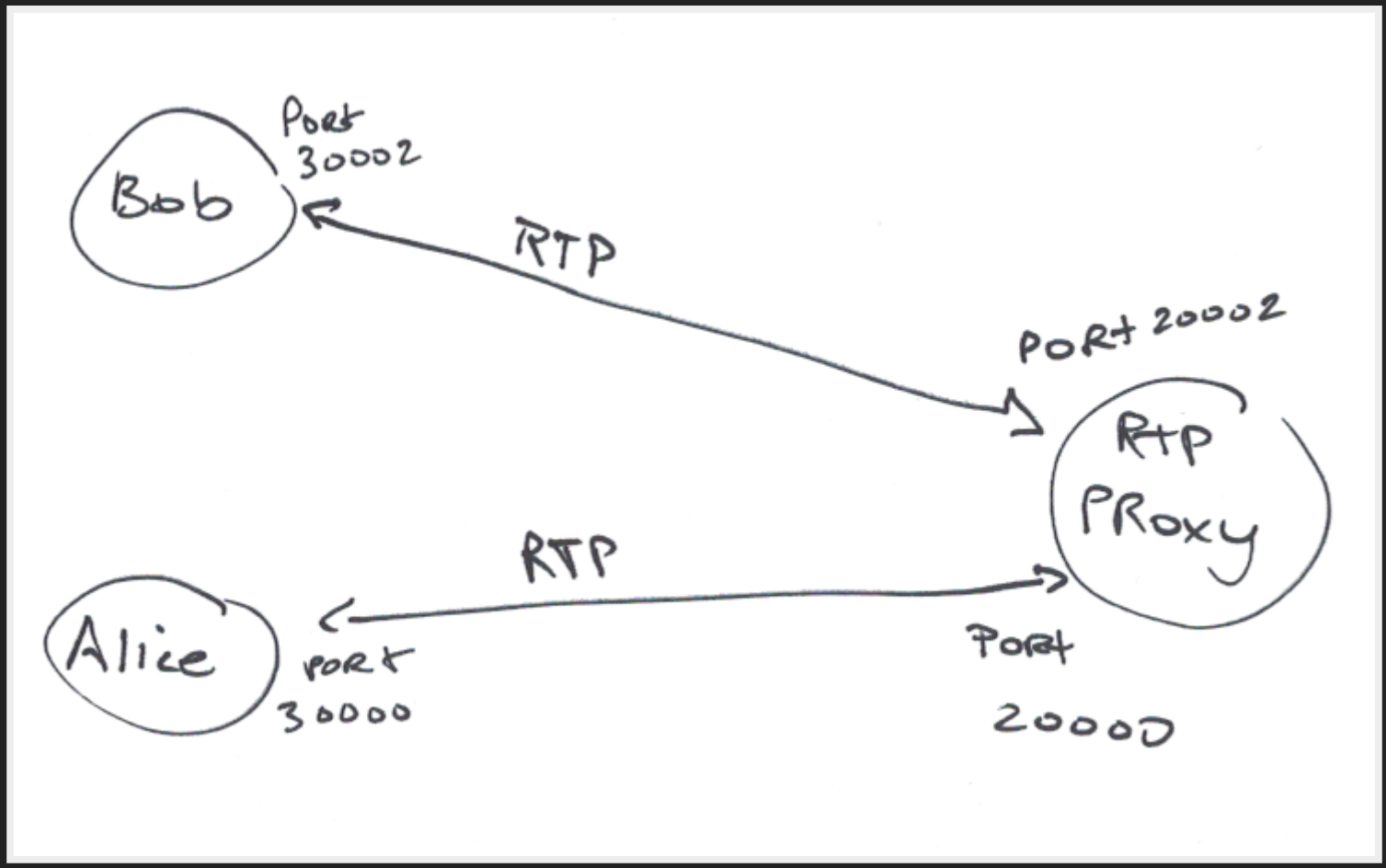
Affected RTP proxies do the following:

1. Listen on ports set in the signaling protocol (e.g. SDP within an INVITE message)
2. Learn about where to send responses by inspecting the incoming RTP traffic
3. Respond to that IP and port, relaying the proxied RTP stream

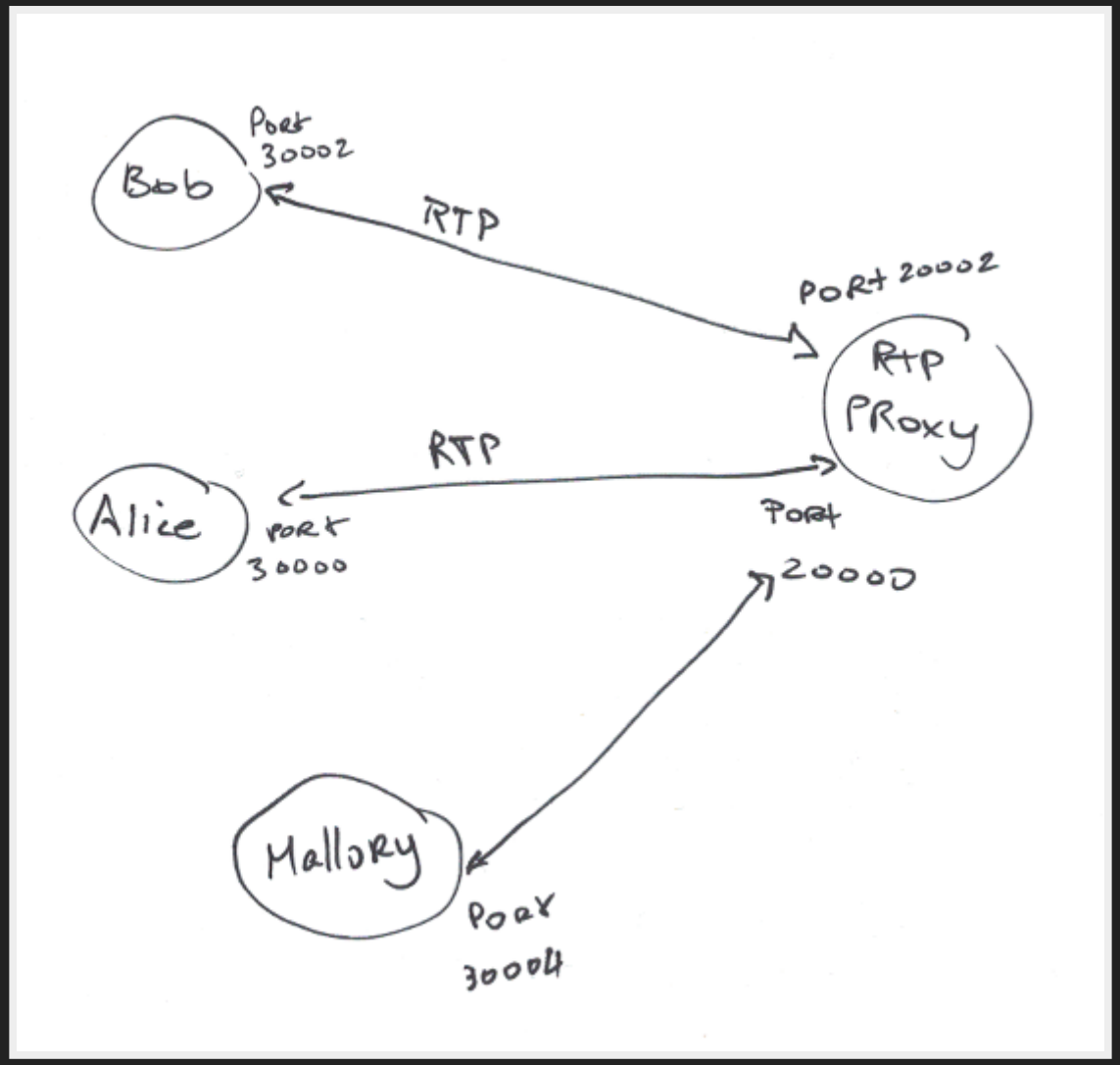
HOW DOES IT ACTUALLY WORK?

From a security perspective, this means:

- **no** authentication takes place
- trust is based on .. not knowing the port? timing?



Normal RTP being proxied from Bob to Alice and back



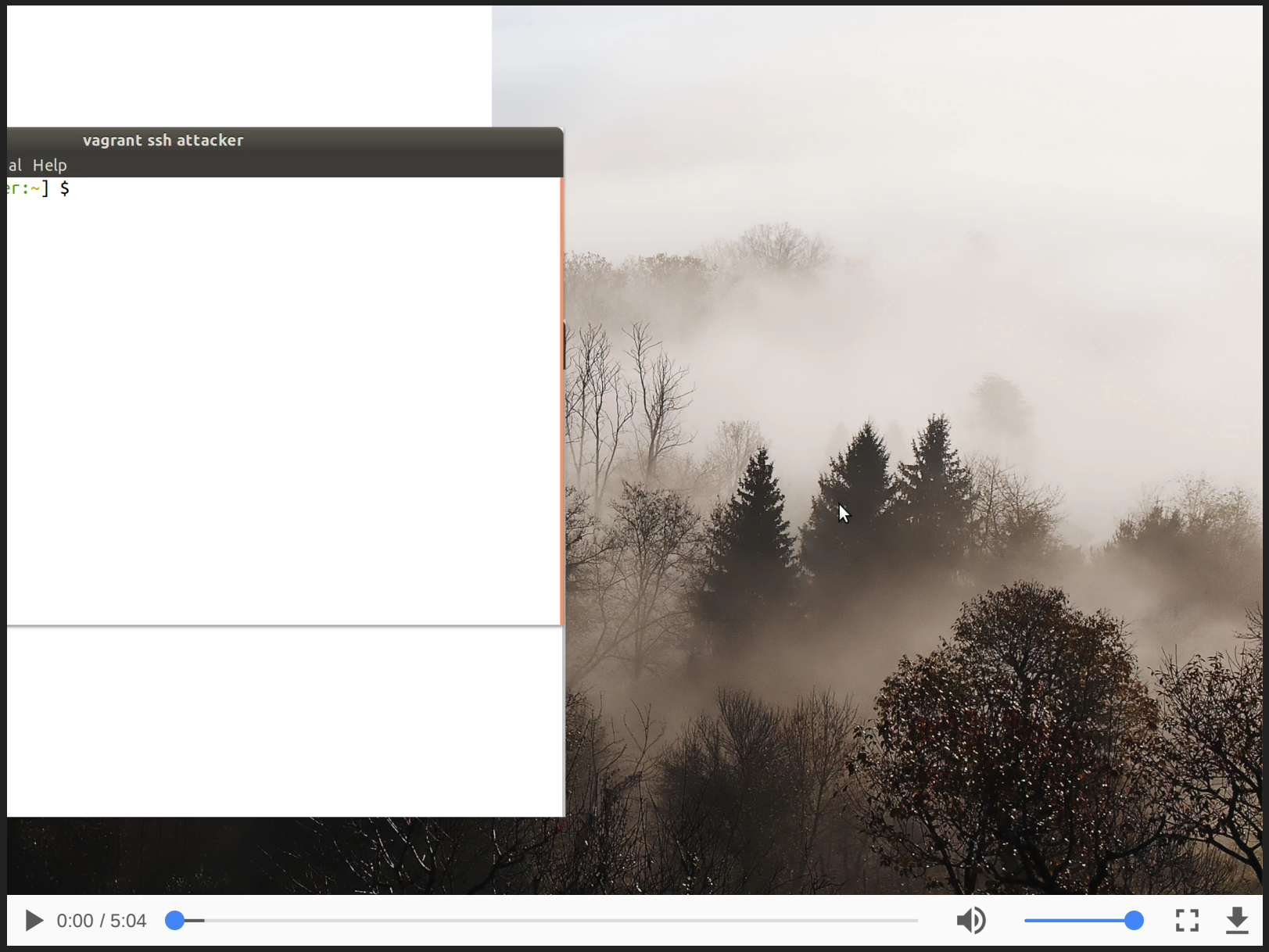
Simple RTP hijack taking place on Alice's RTP session

WHAT HAPPENS IN THIS CASE?

1. Attacker RTP is injected in the RTP stream
2. Attacker receives the proxied RTP stream

The attack has two different security implications

DEMO


vagrant ssh attacker
al Help
r:~] $
0:00 / 5:04

NOTE - THIS IS A WORST CASE SCENARIO

In the demo, one RTP packet caused all RTP traffic to be sent to the attacker. In many other cases, the attacker needs to constantly send RTP traffic.

GIVE CREDIT WHERE CREDIT IS DUE

Klaus-Peter Junghanns presenting in 2010 27th Chaos Communication Congress

PREVIOUS WORK

- Presentation in 2010-12-30: Having fun with RTP by Klaus-Peter Junghanns in 2010 @ CCC
- Olle filed an issue for Asterisk in 2011-09-11: Enable `strictrtp` by default
- Last year at Kamailio World I spoke to Mikko Lehto and he mentioned his concern about RTP proxies not checking source IP/port

PREVIOUS WORK

- Sometime in July last year I started exploring this and started understanding the impact,
- Found one of our clients who had implemented their own RTP proxy, to be vulnerable
- Also found the default Debian packaged RTPproxy to be vulnerable
- Our work, although researched and developed independently from Klaus-Peter Junghanns, is very similar

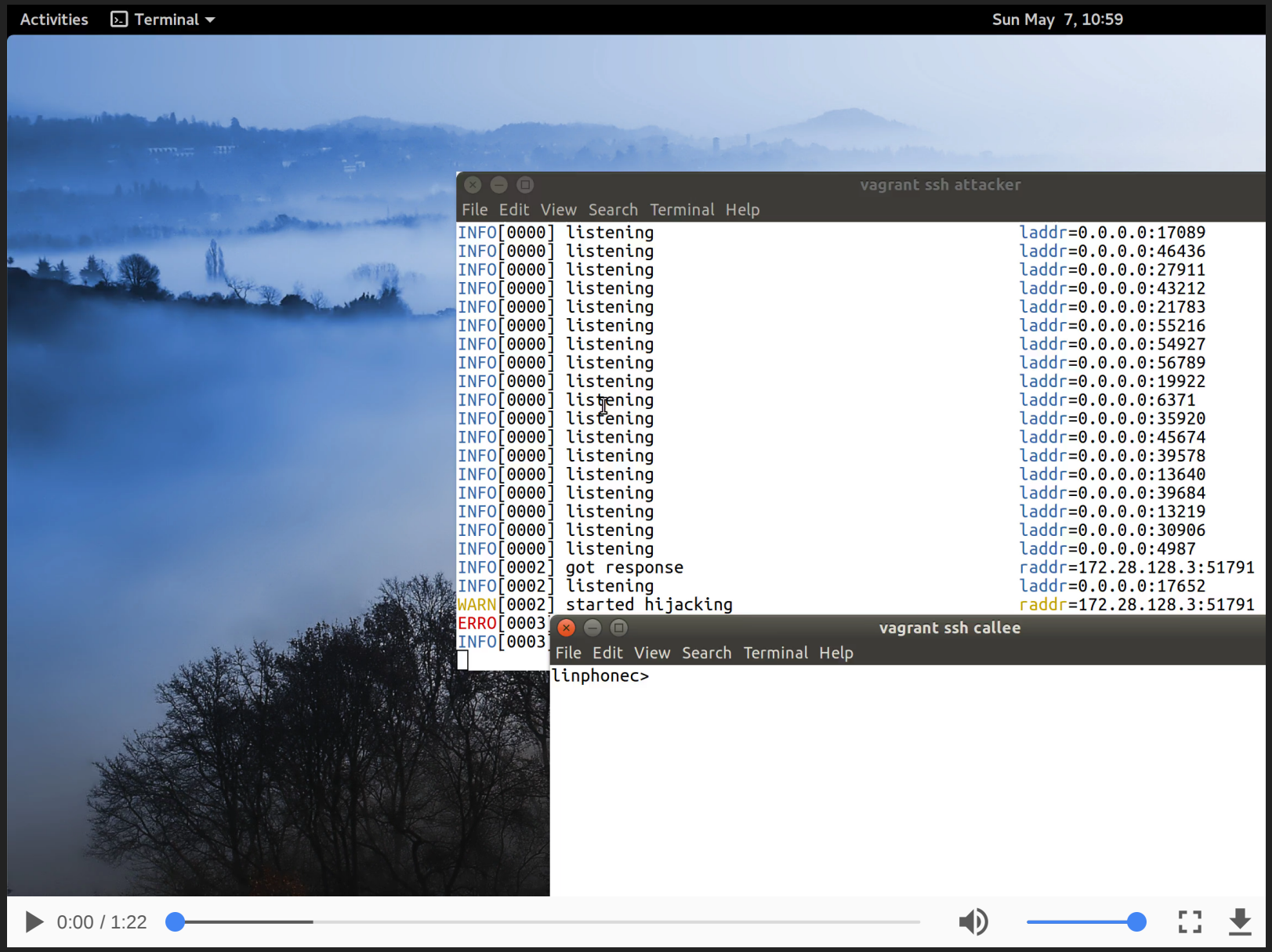
DOES THIS ATTACK SCALE?

i.e. can we attack service providers on a large scale?

SURE

- attackers can send RTP packets to all ports on the media gateways (like a UDP port scan)
- whenever the attacker receives RTP responses, starts sending RTP to that particular port
- SIPVicious PRO supports this by default

DEMO


Activities
Terminal
Sun May 7, 10:59
vagrant ssh attacker
File Edit View Search Terminal Help
INFO[0000] listening laddr=0.0.0.0:17089
INFO[0000] listening laddr=0.0.0.0:46436
INFO[0000] listening laddr=0.0.0.0:27911
INFO[0000] listening laddr=0.0.0.0:43212
INFO[0000] listening laddr=0.0.0.0:21783
INFO[0000] listening laddr=0.0.0.0:55216
INFO[0000] listening laddr=0.0.0.0:54927
INFO[0000] listening laddr=0.0.0.0:56789
INFO[0000] listening laddr=0.0.0.0:19922
INFO[0000] listening laddr=0.0.0.0:6371
INFO[0000] listening laddr=0.0.0.0:35920
INFO[0000] listening laddr=0.0.0.0:45674
INFO[0000] listening laddr=0.0.0.0:39578
INFO[0000] listening laddr=0.0.0.0:13640
INFO[0000] listening laddr=0.0.0.0:39684
INFO[0000] listening laddr=0.0.0.0:13219
INFO[0000] listening laddr=0.0.0.0:30906
INFO[0000] listening laddr=0.0.0.0:4987
INFO[0002] got response raddr=172.28.128.3:51791
INFO[0002] listening laddr=0.0.0.0:17652
WARN[0002] started hijacking raddr=172.28.128.3:51791
ERRO[0003
INFO[0003
vagrant ssh callee
File Edit View Search Terminal Help
linphonec>
0:00 / 1:22

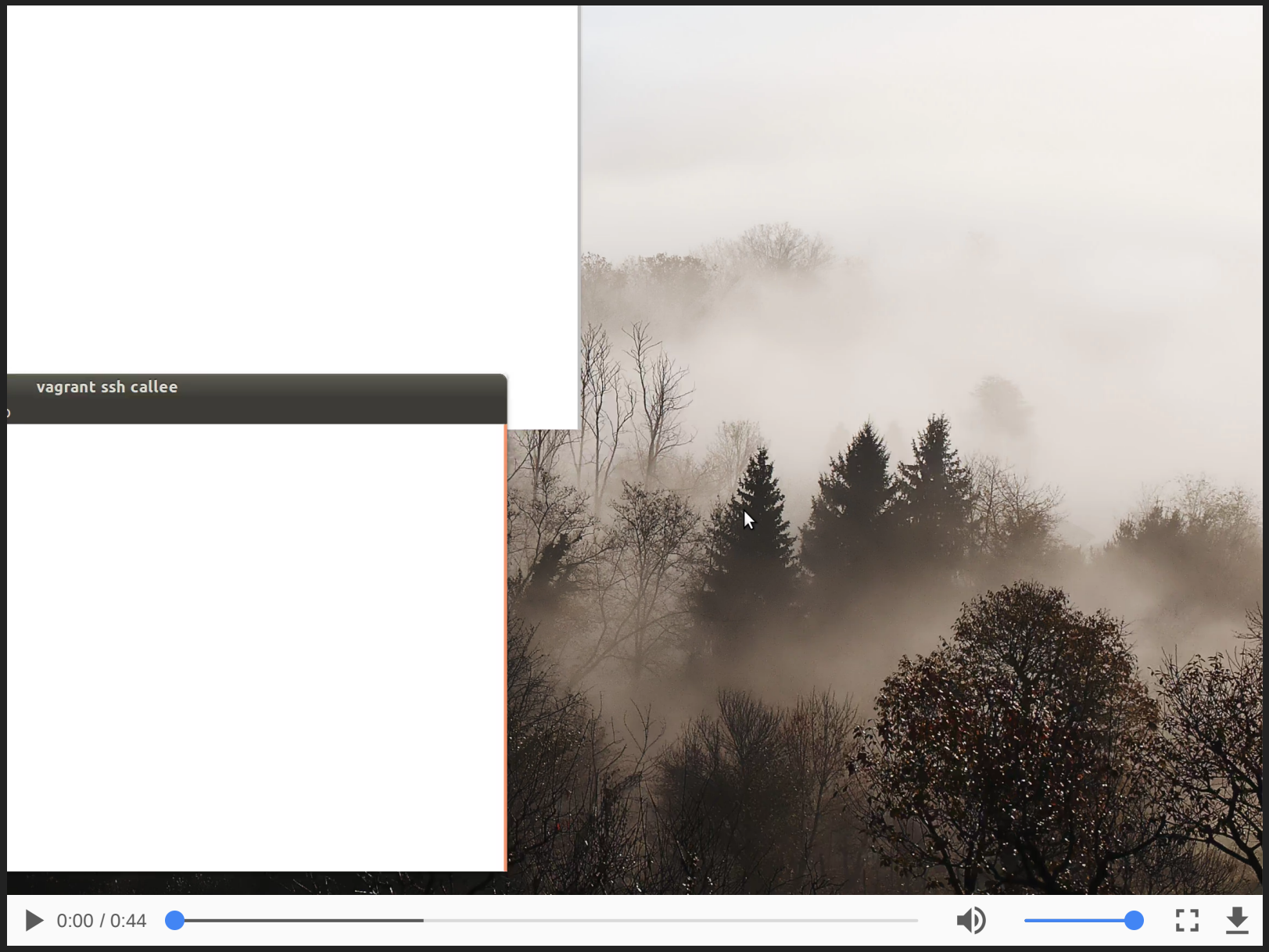
WORD OF CAUTION

Doing this on **live systems** will lead to Denial of Service,

i.e. all calls will go mute

WHAT ELSE CAN WE DO?

INJECT OUR OWN RTP MEDIA


vagrant ssh callee
0:00 / 0:44

WHAT ABOUT WEBRTC AND SRTP?

- WebRTC does **not** need traditional RTP proxies thanks to ICE, STUN and **TURN**
- but we still found proxying of SRTP in WebRTC anyway
- if SRTP is in use, the security implications on confidentiality of the voice data (should) become a non-concern
- seems to be mainly a DoS issue - which in RTC is a major problem

WHAT SOLUTIONS HAVE WE SEEN?

STICK TO WHAT IS ADVERTISED IN SDP

Major problem with NAT

LATCHING, TEMPORARY TRUST, PROBATION PERIOD

- Latching is usually vulnerable to a race condition
- i.e if the attacker is scanning all the time, he/she will get some of the RTP streams before the victim does

LATCHING AND HANDLING CHANGING IPS

- When no RTP has been received for a while by trusted IP, allow for change
- Race condition issue (again), upon each change of IP
- Possibility to DoS the endpoint (unlikely)

SRTP AS A SOLUTION TO THIS ISSUE

- It addresses confidentiality
- Should also address integrity, i.e. injection of RTP traffic introduced by the attacker
- Does not address the denial-of-service aspect

AUTHENTICATED STUN

- Is an effective way to authenticate and whitelist IP addresses
- Seems like IP spoofing might be an interesting vector here
- If successful, would allow RTP injection

HOW DO OPEN SOURCE RTC SOLUTIONS FARE?

- RTPproxy 1.2.1-2.2
- RTPproxy 2.2.alpha.20160822 (github)

RTPPROXY 1.2.1-2.2

main.c

```
573
574         /*
575          * Update recorded address if it's necessary. Set "untrusted address"
576          * flag in the session state, so that possible future address updates
577          * from that client won't get address changed immediately to some
578          * bogus one.
579          */
580         if (i != 0) {
581             sp->untrusted_addr[ridx] = 1;
582             memcpy(sp->addr[ridx], &packet->raddr, packet->rlen);
583             if (sp->prev_addr[ridx] == NULL || memcmp(sp->prev_addr[ridx],
584               &packet->raddr, packet->rlen) != 0) {
585                 sp->canupdate[ridx] = 0;
586             }
587
588             port = ntohs(satosin(&packet->raddr)->sin_port);
589
590             rtpp_log_write(RTPP_LOG_INFO, sp->log,
591               "%s's address filled in: %s:%d (%s)",
592               (ridx == 0) ? "callee" : "caller",
593               addr2char(sstosa(&packet->raddr)), port,
594               (sp->rtp == NULL) ? "RTP" : "RTCP");
```

RTPPROXY 2.2.ALPHA.20160822 (GITHUB)

Very similar to 1.2.1 in that:

- race condition exists within the first few seconds

RECAP AND CONCLUDE

- Vulnerable systems allow:
  - receiving of RTP media
  - injection of RTP media
- This seems to be a widespread security issue
- Some of the solutions implemented have their limiations
- Let us get this security issue sorted

THANKS, APPRECIATION & REFERENCES

- @kapejod for 27c3: Having fun with RTP and his feedback
- Alfred for his feedback
- Daniel for the invitation
- RFC 7362: Latching: Hosted NAT Traversal (HNT) for Media in Real-Time Communication
- RFC 3550 RTP: A Transport Protocol for Real-Time Applications

Q&A

TIME TO ASK ME THOSE TRICKY QUESTIONS ;-)

Test RTP Hijack by getting SIPVicious PRO Beta

https://sipvicious.pro

sandro@enablesecurity.com