27C3 Day 4, 17:15

Having fun with RTP
”Who is speaking???”

kapejod@googlemail.com

Having fun with RTP ”Who is speaking???”
Overview

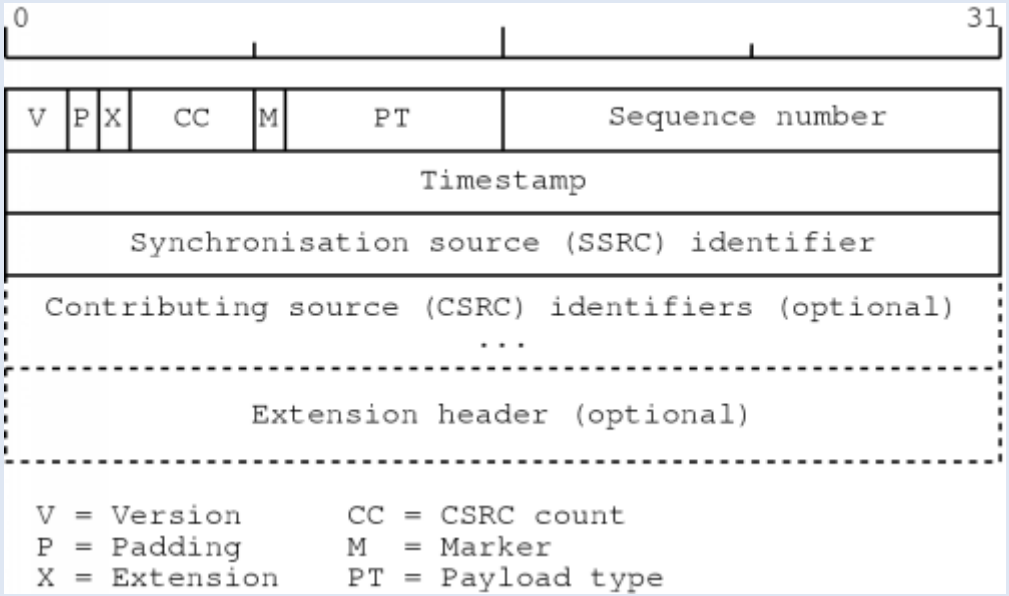
- Short introduction to RTP
- RTP packet structure
- SIP and RTP with NAT
- Shortcomings of the RTP specification
- Finding RTP sessions
- Demonstration: finding all RTP session on an Asterisk server
- Possible exploitations
- Possible solutions to protect against exploitation
- System under test

Short introduction to RTP

- Realtime Transport Protocol
- RFC 1889 (1996), replaced by RFC 3550 (2003)
- Transport of multimedia streams over IP/UDP
- Reordering of packets
- Convey the timing of the sender to the receiver
- Used together with Voice over IP signalling protocols (H.323, SIP, MGCP)
- RTP and RTCP (Realtime Transport Conrol Protocol) usually referred to just by ”RTP”
- RTP (usually) runs on an even numbered port, RTCP for that session on the higher odd port

RTP packet structure

- RTP header:
  - Minimum length of 12 octets, (16kbit/s overhead with 20ms framesize)
  - Sequence numer used for packet reordering
  - Timestamp used for playback
  - SSRC identifies the sender (32bit random value)



SIP and RTP with NAT

- A SIP user agent behind NAT will announce the private IP address in the SDP body, which cannot be reached from outside the local network.
- Typical reason for one-way audio.
- Can be worked around either with a SIPaware router that is proxying or redirecting the RTP to the local network OR by the remote server replying to the IP / port combination from where the received packets originated.
- The RTP specification does not enforce that the received packet will be coming from the same IP / port as the transmitted packets are sent to. (This special case is known as ”symetrical rtp”)
- But most RTP NAT-traversal solutions rely on this and assume that a random RTP port is good enough to protect the session against misdirected data.

Shortcomings of the RTP specification

- It is very difficult to validate that incoming packets are actually RTP or belong to the session (RTP header version, timestamp continuity, sequence number continuity).
- The only complete validation would be to wait for the first RTCP receiver report (which might take 5-10+ seconds) and many RTP devices do not implement RTCP at all.
- Basically anybody can send you RTP packets and you cannot find out which of the RTP streams is the correct one. (IF THEY KNOW THE PORT).

Finding RTP sessions

- Example: Asterisk (also works with other software, like RTPproxy)
- NAT-traversal support for RTP is implemented quite simple to assure that there will be no one-way-audio issues if possible.
- Every received packet on a RTP port overwrites the return IP / port combination, so the next packet will be transmitted to this peer.
- Sending several packets to the same port with a varying delay (between 2 and 10 ms) will make sure that the probability of overwriting the return address increases (the correct RTP peer will send one packet every 20 ms usually).
- A simple loop accross the RTP port range of the server (or the complete port range) does not take very long.
- For every port involved in an RTP conversation the attacker will receive a reply (with actual RTP data from the other side).
- By also ”stealing” RTCP packets the attacker can find out which RTP session is connected to which other RTP session

Demonstration: finding all RTP session on an Asterisk server

- Demo system: Asterisk 1.6.2.16-rc1 running on an embedded system.
- 2 SNOM phones connected to a local network
- Attacker connected via the internet
- NO local network attack (arp cache poisoning is for children)

...

Possible exploitations

- Denial of service
- …
- Call redirection (by sending DTMFs), toll fraud
- ....
- Man in the middle attacks (requires a very good timing), your homework!

Possible solutions to protect against exploitation

- Asterisk 1.6+ has the global option ”strictrtp=yes” in rtp.conf (disabled by default, why?)
- Only accept a new packet source at the beginning of a session (first few packets), can cause problems with music on hold servers and other media sources.
- SIPaware firewall in front of the Asterisk server which opens the RTP port only to the IP/port combinations advertised in the SIP SDP body. (some firewalls , e.g. C*SC*, do not recognize the termination of a session reliably and leave the port open).
- Do not use NAT. Use a VPN instead (like the openVPN firmware for the SNOM phones)
- Do not accept a new media source while there are still packets being received from the original source (requires a bit of effort).
- SRTP? (Asterisk 1.8.0 dropped the call after receiving one single packet)

Systems under test

- Asterisk 1.2, 1.4, 1.6, 1.8
- RTPproxy 1.2.1
- SNOM phones (300, 360, 820) various firmwares
- Linksys 2FXS ATA
-
-
-
- Got anything to test???

Questions ???

- Thanks for your patience...