8-2010

A Covert Channel in RTP Protocol

Chrisopher Forbes
Rochester Institute of Technology

Bo Yuan
Rochester Institute of Technology

Daryl Johnson
Rochester Institute of Technology

Peter Lutz
Rochester Institute of Technology

Recommended Citation

CHRISTOPHER FORBES, BO YUAN, DARYL JOHNSON, and PETER LUTZ (2010) A COVERT CHANNEL IN RTP PROTOCOL. Computational Intelligence: pp. 813-819. https://doi.org/10.1142/9789814324700_0123

A Covert Channel in RTP Protocol

Christopher Forbes, Bo Yuan, Daryl Johnson, and Peter Lutz

Department of Networking, Security, and Systems Administration
B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York, 14623, USA
{*crf6296, bo.yuan, daryl.johnson, peter.lutz*}*@rit.edu*

A new covert channel over the RTP protocol is designed and implemented by modifying the timestamp value in the RTP header. Due to the high frequency of RTP packets, the covert channel has a high bit-rate, theoretically up to 350 bps. The broad use of RTP for multimedia applications such as VoIP, provides abundant opportunities to such a covert channel to exist. By using the RTP header, many of the challenges present for covert channels using the RTP payload are avoided. A reference implementation of this covert channel is presented. Bit-rates of up to 325 bps were observed. The channel is very difficult to detect due to expected variations in the timestamp field and the flexible nature of RTP.

Keywords: Covert channels; RTP protocol, bit-rate.

1. Introduction

Several VoIP based covert channels have been proposed. Druid[2] designed and implemented "SteganRTP" which uses steganography techniques to embed secret messages in the payload of RTP packets. In order to increase the reliability of the covert channel, Druid[2] designed a minimalistic protocol within the covert channel that contains fields such as checksum, type, length, and sequence fields. The design was effective to mitigate the reliability issue, though potentially costly in a low bandwidth channel.

Tian et al[4] seek to provide a real time steganography design that resists detection. They make use of the G.729 codec as a cover medium and established an m-sequence technique to hide the data. A RSA-like key exchange provides for synchronization between the two endpoints. The authors found that their techniques provided good security for transmitting covert data while maintaining the real time requirements of VoIP.

Mazurczyk and Kotulski[5] seek to exchange information over VoIP using digital watermarking and steganography techniques to provide a covert channel. Watermarks and steganographic data are embedded to provide for authentication and integrity of the VoIP stream. Control fields were embedded into the existing protocol headers, while the data were embedded within the voice stream.

Mazurczyk et al[6] compares steganographic techniques that can be used to introduce covert channels within VoIP. The primary method introduced makes use of both timing delays and modifying packet contents for a hybrid channel. The idea is to use excessively delayed packets that are discarded to carry a load of covert data.

Except Mazurczyk and Kotulski,[5] who embedded control field in protocol headers, all RTP based on covert channels reviewed so far are based on embedding covert data in payload of protocols. While the payload provides a potentially large bandwidth for covert channel, it also has limitations, including changes in codec rendering the channel unusable.

In this paper, a covert channel is devised based on fields in the RTP header only. It utilizes the least significant bits of the timestamp in the protocol header to deliver covert message rather than delivering it in the payload. By using the protocol header, the channel provides for broad applicability by ignoring many of the codec issues encountered in the payload. A sample implementation of the covert channel is also presented. Experimental results have shown that the covert channel is reliable.

2. Covert communications in RTP timestamp

The proposed new covert channel is to modify the timestamp field in the RTP header to transmit data. For a regular voice stream sampled at 8000 Hz using G.711, for example, the timestamp is incremented by a value of 160 in each packet rather than incrementing by the actual time passed. The numbering of the timestamps needs only to be in proper sequence to function properly at the receiving end.

A timestamp was first employed to carry covert data in TCP protocol.[7] This channel carried 1 bit in the least significant bit of the TCP timestamp by delaying the packet creation. In TCP, the timestamp is only an option, making it easier to detect if a system does not usually use the option. In contrast, RTP makes use of the timestamp in every packet.

In this research, the G.711 codec was chosen as the reference point due to its standardized nature, high quality performance, and common usage in VoIP communications. G.711 provides a high quality communications

channel, using pulse code modulation at 8,000 Hz with 8 bits per sample for a 64 Kbit/s bit rate.[8]

Data can easily be embedded into the last seven bits based on standard VoIP sampling rate of 8 kHz without disturbing transmission. Seven bits cover a range of 128 (0-127) in decimal, which is still below the value of 160 used for incrementing the timestamp. With 50 packets transmitted per second, this provides a gross data rate of 350 bits per second full duplex, with less available dependent on network conditions, reliability protocols implemented, and needed level of covertness.

3. Characteristics of the Covert Channel

3.1. *Covertness*

This implementation is difficult to detect provided the channel is not already known. The timestamp field is expected by most to contain actual timestamp values, which may contain some variability. In addition, it is possible for RTP streams to be reset during a call, resulting in the timestamp to be reset to a new start time. Current detection of covert channels based on RTP focuses on the payload manipulation, which is left untouched for this channel.

The varying timestamps by codec makes it very difficult to detect manipulation of the timestamp. Since even a single application (VoIP) makes use of a wide range of codecs, simply detecting that timestamps are not incrementing at a certain rate is not easily accomplished. Additionally, newer codecs, such as speex, do not even have an assigned number for RTP payloads, further complicating detection.

Another difficulty in detecting the RTP timestamp is the underlying use of UDP, as UDP is not reliable and may lose packets in transit. It can also be difficult to detect that RTP is in use, as UDP does not specify what kind of data is in its payload and RTP does not need to use a standard port.

The ease of detecting the channel also depends upon how much data is being transferred. The more data that is transferred, the more irregular the timestamp may appear. Slowing the amount of data sent per packet, such as two bits instead of seven, also notably increases covertness of the channel. Use of a man-in-the-middle technique can also increase covertness, as the RTP timestamps can be returned to their original values before reaching the end user.

3.2. *Reliability*

The channel is not particularly robust without further mechanisms being used to make it a reliable carrier. This is because it is carried over a UDP carrier, as opposed to a reliable carrier such as TCP. As such, part of the available bandwidth can be dedicated to a minimalistic protocol to improve the reliability and integrity of the messages. However, given the limited bandwidth available, the channel may still experience reliability issues when faced with a highly disruptive channel.

3.3. *Bandwidth*

The bandwidth of the channel is up to 7 bits per packet by 50 packets a second, for a total of 350 bits per second. This max rate can be closely realized using the full 7 bits of covert data on a well-functioning network. The speed from the channel is sufficient for text based communications and small file transfer, among other applications. Distance will not decrease the speed of the transmission, as it is essentially two unidirectional streams.

4. Reference implementation

The reference implementation of this covert channel was built upon the sample implementation provided by pjsip.org.[9] The application, pjsua, provides a basic SIP client based upon libraries from the PJSIP open source project.

While many modules are used in placing calls, the only parts involved in the covert channel are rtp.c and rtp.h. Accordingly, the vast majority of the application is not even aware of the timestamp modification. To further conceal the presence of the covert channel, the covert channel data is removed from the timestamp by rtp.c before heading to the rest of the application. By doing so, the modifications made by the covert channel are seen by as little of the application as possible, and do not interfere with the jitter buffer or other components.

5. Experimental results

The experimental results were collected running two instances of the client, one running on Windows Vista SP2 and the second on a Windows XP SP3. Data was captured using Wireshark for analysis. Unless otherwise noted, the G.711 codec was used for the RTP payload with a sampling rate of 8000 Hz.

For NY to IL tests, both machines were behind home routers using NAT, and connected to the Internet using cable modems. Both computers were connected to the LAN via 802.11G wireless networking. This wireless networking provided another source of potential interference for these tests, as multiple computers were using the shared wireless medium at both ends.

For the tests, the standard file used was a 1024 character block of ASCII text using Windows line endings. This file was transmitted in full duplex. In addition to the standard 1024 character file, a 300 KB file was also transmitted on some local tests to check for the ability to send large files. Data was transmitted without an additional reliability protocol at 7 bits per packet (7-bit) and 2 bits per packet (2-bit).

5.1. *Bandwidth*

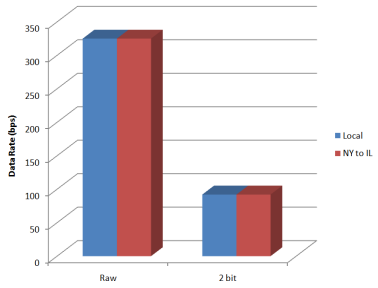

Fig. 1. Speed of the data transfer with various modes of the reference implementation.

The speed of the application when running in the 7-bit mode was close to the expected theoretical maximum. In tests, both local and long distance from NY to IL, the speed transmission speed was 325 bps, compared to the theoretical maximum of 350 bps as illustrated in Fig. 1. The 2-bit mode similarly came close to its theoretical maximum, averaging 92 bps out of a theoretical 100. It is not entirely clear what caused this slight discrepancy though it not surprising that it does not exactly match the theoretical rate. A minor difference in the timing of the packets could easily cause such a drop in rate.

5.2. *Reliability*

Reliable transfers with no errors were repeatedly obtained over both the local network and across the public Internet. The client transmitted flaw-

lessly on all local and remote tests, even from NY to IL. The client also performed flawlessly on a local 300 KB test file. This persistence of the data indicates a quality Internet infrastructure and that most calls could use the client without additional reliability provisions.

5.3. *Covertness*

Overall the channel exhibited a high degree of covertness, making it difficult to detect. This was mainly due to the flexible nature of the RTP protocol and operation of the RTP timestamp. Even with knowledge of the covert channel, one must first find a data bearing packet, then determine the expected timestamp value, before being able to decode the covert data. Firewalls inspecting only layers one through four are unable to block the channel; only application layer firewalls with specific knowledge of the covert channel can detect it, and even then face significant difficulties.

The jitter experienced with the rapid rate of packets also leads to detection problems. Data on the wire may not arrive in the order it was sent due to the use of a UDP transport. To effectively process this data while looking for the channel, one needs a jitter buffer and to arrange packets based on sequence numbers.

Mathematically the channel is hard to detect since the RTP timestamp field need not follow any particular pattern. The RTP timestamp increment changes depending on protocol used for the RTP payload at that point in time, and may use system clock time. Additionally, the fact that the timestamp may not increment between packets further complicates analysis, and could make it easy to get false positives.

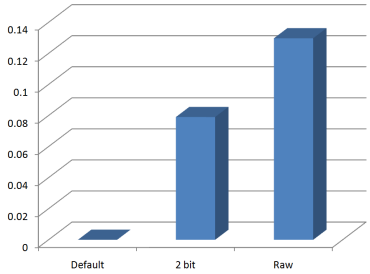

Fig. 2. Entropy of the timestamp field.

Fig. 2 shows entropies of the timestamp increment for G.711, for each

of the different modes. 2-bit mode with no reliability protocol is the most covert. 1024 characters were sent on the covert channel out of 10,000 RTP packets for the calculations in Fig. 2.

6. Conclusion

We have shown the ability to transmit and receive data using a new covert channel over RTP, without interrupting the reception of the voice stream. The reference implementation shows that this covert channel can be practically implemented and used. Speed was shown to be sufficient for two-way text based communications and small file transfer. Reliability was also shown to be good, if not always perfect. The ability to detect this channel was also seen to be difficult, given the many packets involved and the way RTP operates. Future work will further develop this reference implementation with increased reliability, flexibility, and usability.

References

1. X. Wang, S. Chen and S. Jajodia, Tracking anonymous peer-to-peer voip calls on the internet, in *CCS '05: Proceedings of the 12th ACM conference on Computer and communications security*, (ACM, New York, NY, USA, 2005) pp. 81–91.
2. Druid, *Real-time steganography with RTP* Online: http://www.uninformed.org/ (September 2007).
3. T. Takahashi and W. Lee, An assessment of VoIP covert channel threats, in *Proceedings of Third Intern. Conference on Security and Privacy in Communications Networks*, (Nice, France, 2007) pp. 371–380.
4. H. Tian, K. Zhou, H. Jiang, J. Liu, Y. Huang and D. Feng, An M-sequence based steganography model for voice over IP, in *Proceedings of IEEE Intern. Conference on Communications*, (Dresden, Germany, 2009) pp. 1–5.
5. W. Mazurczyk and Z. Kotulski, Covert channel for improving VoIP security, in *Advances in Information Processing and Protection*, (Springer-Verlag New York Inc, 2007) pp. 271–280.
6. W. Mazurczyk, J. Lubacz and K. Szczypiorski, Hiding data in VoIP, in *Proceedings of The 26th Army Science Conference*, (Orlando, Florida, 2008).
7. J. Giffin, R. Greenstadt, P. Litwack and R. Tibbetts, Covert messaging through TCP timestamps, in *Privacy Enhancing Technologies*, eds. G. Goos, J. Hartmanis and J. van Leeuwen, Lecture Notes in Computer Science, Vol. 2482 (Springer Berlin/Heidelberg, 2003) pp. 189–193.
8. ITU-T, Pulse Code Modulation(PCM) of voice frequencies, Online: http://www.itu.int/rec/T-REC-G.711-198811-I/en (1993).
9. B. Prijono, pjproject-1.3 Online: http://www.pjsip.org/download.htm (2009).
10. Asterisknow 1.5.0 Online: http://asterisknow.org/(2009).