On Securing RTP-Based Streaming Content with Firewalls*

Liang Lu, Rei Safavi-Naini, Jeffrey Horton, and Willy Susilo

Centre for Communication Security Research,
School of Information Technology and Computer Science,
University of Wollongong, Australia
{ll97, rei, jeffh, wsusilo}@uow.edu.au

Abstract. Delivery of real-time streaming content is an increasingly important Internet application. Applications involved in processing streaming content may have exploitable vulnerabilities, as many other applications have been discovered to have, and using a firewall to filter out malicious traffic may provide some benefit. However, as these applications largely rely on traffic carried by RTP/UDP, firewalls that are unaware of the behaviour of RTP data streams have difficulties in filtering out malicious traffic injected into a stream by an attacker. In this paper, we observe a vulnerability in the current RTP protocol which allows an attacker to inject malicious traffic into a data stream, and present a scheme that allows a stateful firewall that keeps state from RTP packets to detect such malicious traffic. Our technique uses non-static fields such as RTP sequence numbers to improve the inspection scheme by modelling streaming traffic and detecting malicious streams based on deviation for this model. We show effectiveness of our approach by giving the results of our experiments.

Keywords: Network security, firewall, streaming content.

1 Introduction

Recent years have seen increasing applications of streaming content online, such as live video or audio broadcast, IP telephony and teleconferencing. This growth is driven by technology advancement such as PC performance, residential broadband access and virtual-reality technologies and is expected to continue. Like other networking applications, applications involved in processing streaming content may have exploitable vulnerabilities. It is highly desirable to have a firewall that can filter out malicious traffic in streaming content.

However, streaming applications have different behavior to conventional networking applications. Unlike conventional networking applications which are mostly based on TCP, streaming applications based on open protocols (e.g. [16],

* This work is partially supported by Cooperative Research Center - Smart Internet Technology (CRC-SIT), Australia.

Y.G. Desmedt et al. (Eds.): CANS 2005, LNCS 3810, pp. 304–319, 2005.

[3], [21]) largely rely on RTP [17] for data delivery. RTP typically uses UDP as the underlying transport protocol. Although TCP may also be used, it is mainly used for tunneling through firewalls at the cost of timely delivery of stream data when there is packet loss. Because conventional firewalls have difficulties in detecting malicious UDP injection due to its connectionless nature, they cannot effectively filter out malicious traffic injected in streaming content.

In this paper, we first discuss the difficulties that various types of conventional firewalls have when filtering UDP-based streaming applications. Then we give an overview of the protocol stack of streaming applications, from where we can obtain more reliable information that distinguishes legal traffic from injected traffic. This information is not utilized effectively in current systems and results in vulnerabilities which an attacker can exploit to "hijack" a streaming session. That is, an innocent stream is replaced by a malicious one without the receiver detecting this replacement. We provide a novel and effective inspection scheme that can be used in stateful firewalls to filter out malicious traffic injected into streaming content, and hence prevent the injection. Finally, experimental results are given to show the effectiveness of our scheme.

The rest of this paper is organized as follows. Section 2 provides a brief introduction to streaming content and firewalls, and explains the reasons why conventional firewall techniques have difficulties in handling streaming content. Section 3 explains the vulnerability that allows an attacker to inject malicious traffic into streaming content. Section 4 presents a formal modelling of streaming content behavior, based on which our filtering scheme is then introduced. Section 5 gives experimental results showing the effectiveness of our scheme. Section 6 summarizes related work, and section 7 concludes the paper.

2 Preliminaries

2.1 Streaming Content Overview

Prior to the development of effective content streaming techniques, users needed to fully download media files to local storage devices before they could begin playing them. This could take from seconds to hours depending on the file size.

With streaming technology, users can start playing media files immediately or after a short buffering time. A chunk of media file content is packetized into a large number of small portions, which stream like little drops of water through network pipes to local devices. Streaming technology also enables users to select a particular section of a media file such as the second half of a soccer game, or communicate with other peers in realtime. Major applications of streaming content include IP telephony, live video/audio broadcast and stock monitoring. The main disadvantage of streaming technology from a user perspective is that received content cannot be archived or redistributed easily.

Streaming application protocols such as *Real Time Steaming Protocol (RTSP)* [16], *Session Initiation Protocol (SIP)* [21], and *H.323* [3] are defined at the application layer. Typically, these protocols use TCP to reliably deliver session control messages such as setup, manipulate, and tear down commands. The data