2009

San José State UNIVERSITY

RTSP Protocol

Focus on SECURITY

Samuel MONY – Philippe SAWADOGO
CMPE 209
03/03/2009

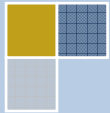

Table of Contents

Introduction

The RTSP protocol is based on the HTTP protocol, and is commonly used to manage media content streaming.
This protocol doesn't directly deals with the streaming content, but uses the RTP protocol to handle content transmissions.

The RTSP protocol provides some basics security mechanisms coming from the HTTP protocol, to perform communication between entities.
Because of its HTTP background, the RTSP protocol is subject to common security attacks.

This document will describe how the protocol is secured and what kind of attacks can be attempted.

1. RTSP overview

1.1 Description of the protocol

RTSP means Real Time Streaming Protocol. This protocol is specified in an IETF's rfc: rfc2326.
This specification describes a HTTP based protocol, using a lot of references to the HTTP1.1 protocol specification: rfc2616. Clients and servers can then exchange ASCII-based messages over TCP, using messages' headers to perform requests or send information.
This protocol is used

1.2 Application

RTSP has been designed to fulfil a media streaming diffusion manager role.
That's why this protocol can be used to perform video and audio real time diffusion.
The protocol has been specified to handle high as well as low bandwidth, and to adjust the stream compression as soon as the bandwidth changes to optimize the transmission.
A lot of web sites use its HTTP protocol background's particularity to perform video and audio diffusion.
Its real time bandwidth optimization particularity is also very useful in video conference systems.

1.3 Functioning

RTSP is a streaming manager protocol. This means that media contents are not directly handled by the protocol, but are handled by the paired protocol RTP.
RTSP is mainly used to launch, stop and replay a stream using specific messages called "commands".

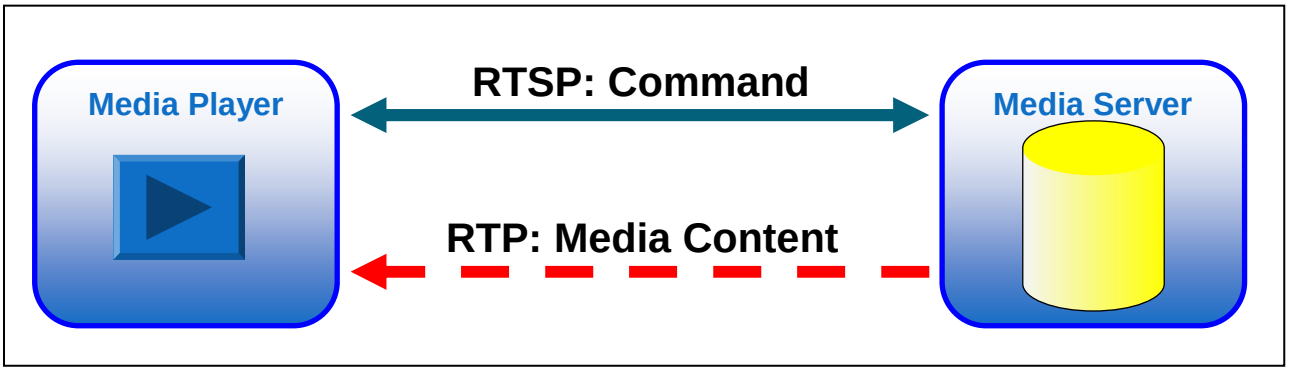

Figure 1: RTSP mechanism

2. Security mechanisms

2.1 Sessions

To ensure the security, the RTSP protocol defines an embedded session management mechanism.
The protocol then supply two different types of session identifiers: the session ID and the conference ID, to ensure a part of the authentication.

The session ID is randomly generated when two entities established a connection. Generally this ID is generated on RTSP server side. To be valid the ID must be composed of at least eight digits and/or letters.

The conference ID is generated by a third party protocol, external to the RTSP server. This generated ID must be globally unique, and should be composed of multiple characters types to be valid.
Theses ID ensure that the client can access the requested resources on the server.

2.2 HTTP Authentication

Using only the session's security mechanism wouldn't be enough in several application fields.
That's why the RTSP protocol uses its HTTP background to supply the well known HTTP Authentication service.
The HTTP Authentication service is specified through the rfc2617. It allows to perform client authentication using the challenge mechanism.
Two different types of HTTP Authentication service are available: the Basic Access Authentication and the Digest Access Authentication.

Using the Basic Access Authentication service, a server would simply send a "Basic" challenge token to the client, claiming for a login and a password.
The client would concatenate the login and the password with a ":" character between the two values, and encode the result using the Base64 cipher before sending it back to the server.

Text content	M							a							n							
ASCII	77							97							110							
Bit pattern	0	1	0	0	1	1	0	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
Index	19							22					5					46					
Base64-Encoded	T							W					F					u					

Figure 2: Base64 algorythm

Using the Digest Access Authentication, a server would perform the same behaviour by sending a "Digest" challenge token to the client, claiming for the login and the password.
The client would also perform the same actions except that instead of using the Base64 cypher, it would use the MD5 hasher on the concatenation result to make the content impossible to decrypt.

3. Security issues

As we saw previously, RTSP Protocol has the same design as HTTP protocol. Thus, some exploitable vulnerabilities related to bad (or incautious) RTSP implementations can occur during streaming media.
These vulnerabilities are the following:

3.1 Abuse of Server Log Information

RTSP Server can retrieve personal information from the client user, and generates Log files. Protecting these files is particularly important to respect users' privacy.

3.2 Man-in-the-middle Attack

It is a kind of active eavesdropping attack. The hacker intercepts the stream between the client and the server and redirects it. In that case, it can read, insert and modify messages between the two parts. The victims are not aware that the stream channel is compromised.

3.3 Attack on file and path name

It is really important that RTSP servers take care to restrict the documents that they want to make available. While translating RTSP URIs, RTSP servers must pay attention to not send files that clients shouldn't collect.
For instance, in many operating system like Linux or Microsoft Windows, the double point ".." means to locate on the above (parent) directory.
If the parent directory contains sensitive files and is not protected, some experiences have shown that it could lead to a security risk. This security risk is particularly present with HTTP protocol and can occur in the RTSP protocol.
Even if RTSP URLs are often opaque and don't necessarily contains file system semantics, it is more cautious to take care of this risks, and prevent it by defining restricted area on the server.

3.4 DNS Spoofing

HTTP Clients use the Domain Name Service (DNS) to resolve the IP address/DNS Name association.
The DNS Spoofing attack is based on this deliberate mis-association.
RTSP protocol is subject to the same type of attack. If clients don't rely on their name resolver in order to get a confirmation for an IP address/DNS name association, or otherwise don't take care of information reported by DNS while the servers make optimization by caching, DNS Spoofing attack can occur.
Generally, RTSP sessions' connection times are typically longer than HTTP sessions; therefore DNS optimization for RTSP client should be less frequent.
Nonetheless it is more cautious to take care of the IP/DNS mapping.

3.5 Denial of service

The RTSP protocol allows the opportunity for an attack called remote-controlled denial-of-service.
In such a case, an attacker can initiate a traffic flowing to one or several IP addresses by modifying the destination in Setup requests. The goal of this kind of attack is to bring the server or the network down by overloading the bandwidth.
The solution may be to check client's identities while initiating a traffic session, by using a client's database or the RTSP authentication mechanism.

3.6 Session hijacking

RTSP protocol uses a random identifier while starting a streaming session, instead of a special relation between a transport layer connection and a RTSP session. If an attacker catches the random identifier, he can intercept the stream.
A solution against this kind of attack is to use a large random number as identifier, and avoid sequential series of identifiers to reduce the risk.

3.7 Authentication mechanism (www-Authentication mechanism)

RTSP protocol contains an authentication mechanism (called www-Authentication mechanism), that should enable to initiate safe connections between clients and servers.
This mechanism allows using a login and a password in order to start a new session.
In practice, it appears that is not really safe due to the fact the client's login and the client's password are sent in clear text. Below a sample of trace intercepted with Wireshark showing our login (MyLogin), and our password (MyPassword):

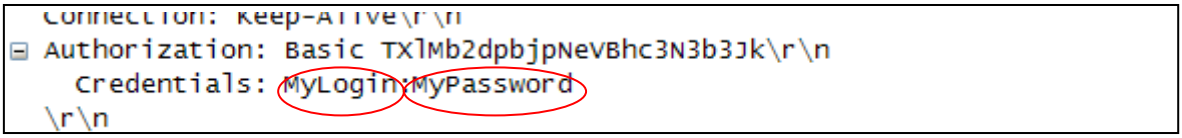

Figure 3: HTTP Authentication trace

Thus, RTSP protocol is subject to the risk of stealing sensitive information.

Conclusion

The RTSP protocol is built with the same design as the HTTP protocol, and contains the same secure mechanisms. These mechanisms are not really efficient, and can be broken easily in case of poor implementation of the protocol. Some familiar applications use the RTSP protocol to enable streaming functionality. Among the list, we can specially think about *Quicktime* (developed by Apple). The later is subject to several vulnerabilities which are currently massively used by some hackers.

But RTSP protocol can also be secure with different technologies, like TLS, that allow being really secured in case of needs.

Bibliography

IETF:

- RFC 2326: RTSP
- RFC 2616: HTTP

Web Sites:

- http://en.wikipedia.org/wiki/Base64
- http://frontier.userland.com/stories/storyReader$2159
- http://www.httr.ups-tlse.fr/pedagogie/cours/tcp-ip/rtsp/index.html#S4
- http://www.cs.columbia.edu/~hgs/rtsp/