WebRTC UDP Firewall Traversal

Authors: Tirumaleswar Reddy (tireddy@cisco.com), Prashanth Patil (praspati@cisco.com), Dan wing (dwing@cisco.com), Bill Ver Steeg (versteb@cisco.com)

Introduction

The Web Real-Time communication (WebRTC) framework [I-D.ietf-rtcweb-overview] provides protocol building blocks to support direct, interactive, real-time communication using audio, video, collaboration, games, etc., between peer web browsers. For peer to peer communication, WebRTC uses Interactive Connectivity Establishment (ICE) protocol [RFC5245] for gathering address candidates and doing connectivity checks.

Security policies of enterprise networks typically require filtering of incoming unsolicited traffic, blocking certain protocols, and doing application-level filtering and scanning for spam, malware, and intellectual property. Most of these protocols run over TCP and are well supported by firewalls and application-specific proxy functions .

Because most protocols use TCP, many enterprise firewalls are configured to simply block UDP. This paper discusses what makes UDP more difficult to enable on a firewall, and how existing firewall policies can be retained with WebRTC.

Why is TCP traversal not problematic?

TCP clearly indicates the beginning of a flow (SYN) and end of a flow (FIN or RST). This is used by firewalls to open and close pinholes. Exceptionally, a TCP connection that has received no traffic for a long time also has its pinhole closed (to accommodate network topology changes or failure of both TCP peers). Firewalls also perform protocol validation to clean up problems with out-of-window TCP segments and overlapping TCP segments. This allows the firewall to protect the network and protect hosts from several attack vectors (replay attacks, host IP address probing, DDOS attack etc.).

Why is UDP traversal problematic?

For UDP flows, the first outgoing packet on a 5-tuple will be used by the firewall as a start-of-session indicator. But UDP does not have an end-of-session indicator, so the firewall has only two ways to close a pinhole: timing out the pinhole after the interior host does not send traffic for several seconds or the interior host generates a fatal ICMP error. Because there is no reliable way to determine that a session is being stopped, the firewall has a much harder job. It could implement an ALG and be aware of whatever semantics are imposed by the higher-level code on top of UDP. It could also rely on a set of well know application servers to inform it of sessions as they start and end, but that suffers from many challenges like application servers hosted independently of the network on which they are used.

Using an ALG, a firewall can determine when the call is terminated and close any dynamic mappings created for the media session. But the problem is session signaling between the WebRTC application running in the browser and the Web server could be using TLS, in which case the ALG no longer has access to the signaling. Moreover, WebRTC does not enforce a particular session signaling protocol to be used, so firewalls using ALGs would fail to inspect the signaling to identify the 5-tuple used for each media stream. Furthermore, the session signaling and the peer-to-peer media may traverse different Firewalls.

In the absence of an ALG or help from application servers, there is no way to deterministically tell that this session has been idle for a while or has ended. The net effect is that a firewall that does a good job with general UDP traffic is much more resource intensive than the same functionality for a set of TCP sessions. “Resource intensive” strongly correlates to “expensive” and “brittle”. This tends to cause network operators to block all UDP traffic except for the protocols that they absolutely have to have.

Solution Overview

The following sections outline and provide some analysis of various solutions to the issues raised regarding WebRTC media traversing firewalls.

TURN

Traversal Using Relays around NAT (TURN) [RFC5766] is a protocol that is often used to improve the connectivity of Peer-to-Peer (P2P) applications (as defined in Section 2.7 of [RFC5128]). TURN allows a connection to be established when one or both sides are incapable of a direct P2P connection. The TURN server is also a building block to support interactive, real-time communication using audio, video, collaboration, games, etc., between two peer web browsers using the Web Real-Time Communication (WebRTC) framework.

Firewall traversal using TURN

Enterprise Firewall policy typically has a white-list of permitted outside applications/sites and can blacklist HTTP(S) connections via various forms of detections (DNS lookup, ALPN, HTTP URL Filtering, DPI proxy that at least performs HTTPS inspection of SNI in TLS exchange and validates SSL records, HTTP(S) proxy etc.). Firewall in this configuration would block TCP/UDP connection external peers in the Internet and arbitrary TURN servers. For example firewall would block usage of STUN with external peers and force the clients to use enterprise provided TURN server for all external WebRTC media communications.

Enterprise firewalls are configured to permit UDP, TCP traffic to well-known TURN servers to allow WebRTC media streams and data channels. Enterprise TURN server can use third party authorization http://tools.ietf.org/html/draft-ietf-tram-turn-third-party-authz-04 to distinguish between business and social calls. Enterprise provided TURN server will have to support both first and third party authorization. This way client can use third party authorization for WebRTC servers that have business relationship with the Enterprise network and use first party authentication for non-business related calls.

Advantages

- Browsers already support TURN client and TURN servers are already available.
- TURN server provides location privacy from external peer.
- A TURN server could be deployed for RTP Mobility.
- A TURN server may be used for IPv4-to-IPv6, IPv6-to-IPv6, and IPv6-to-IPv4 relaying.
- TURN client can indicate start and end of UDP flows.

Disadvantages

- TURN server could increase media latency as explained in section 4.1.2.2 of [RFC5245].
- High-end TURN server would be needed (For example when TLS-over-TCP transport or DTLS-over-UDP transport is used between the clients and the TURN server).
- TURN server could either be located in the DMZ of the enterprise network or located in the public Internet. If the TURN server is located in the public Internet it comes at a high cost to the provider of the TURN server, since the server typically needs a high-bandwidth connection to the Internet as explained in the introduction of [RFC5766]. As a

consequence, it is best to use a TURN server only when a direct communication path cannot be found. When the client and a peer use ICE to determine the communication path, ICE will use hole punching techniques to search for a direct path first and only use a TURN server when a direct path cannot be found.

- Some of the other limitations of TURN explained in section 2.6 of [RFC5766] are, the value of the Diffserv field may not be preserved, the Explicit Congestion Notification (ECN) field may be reset etc.
- Double encryption of media and data channels when (D)TLS is used between the client and the TURN server.
- If branch offices have local Internet breakout then TURN server will have to be deployed and maintained in each branch office.

PCP

Port Control Protocol (PCP, [RFC6887]) provides a mechanism to describe a flow to the network. The primary driver for PCP has been creating port mappings on NAT and firewall devices. When doing this, PCP pushes flow information from the host into the network (specifically to the network's NAT or firewall device), and receives information back from the network (from the NAT or firewall device).

Firewall traversal using PCP

PCP resolves the WebRTC traversal problems by restricting firewall traversal to authorized clients and communicating mapping lifetimes and call termination between the PCP client and the PCP-controlled firewall. A PCP Server can also enforce per-host quotas for mappings.

Advantages

- PCP client can signal start and end of UDP flows.
- PCP can optimize NAT and firewall keepalives.
- PCP avoids TURN-in-TURN tunnel (https://tools.ietf.org/html/draft-schwartz-rtcweb-return-03) for privacy. Client can pick application provided TURN server for location privacy.
- Other advantages of PCP are discussed in https://tools.ietf.org/html/draft-penno-rtcweb-pcp-00.

Disadvantages

- PCP-aware firewalls are not deployed and browsers do not yet support PCP.

Conclusion

WebRTC UDP firewall traversal needs both TURN first party authentication and third party authorization. The long-term solution could be to deploy PCP-aware firewalls to address WebRTC firewall traversal problem and continue using TURN server for location privacy, Mobility, IPvx-to-IPvy relaying etc.