Thoughts on Threat Models for Real-Time Communications on the Web

Richard Barnes, BBN Technologies

Traditionally, real-time communications and the web have lived separate lives, with separate software stacks, separate operator communities, and separate operational practices. In particular, each community has developed its own security mechanisms and practices, based on understandings of adversary capabilities and threats specific to each domain. As more capabilities for inter-working between these two technologies are created, adversaries capabilities will change as well.

In this paper, we will discuss two general areas where different features of web communications and real-time communications can combine to create new threats. We'll first compare the intermediary models of the two applications, and consider how the corresponding classifications of on-path adversaries might map to a combined system. Second, we'll look at the some of the assumptions underlying the linking or session-initiation patterns for the web and for real-time applications, and how combining these models might create some new possibilities for off-path attackers.

In this brief paper, we will not consider countermeasures against these attacks. Some of them can be prevented with judicious use of current security technologies, but others may require new techniques. Our goal here is to lay out some of the new threats that arise from coupling real-time applications with the web.

On-Path: Intermediaries

Real-time applications have traditionally had a very different model from the web for how intermediaries participate in protocol interactions. In protocols such as SIP, XMPP, and H.323, intermediaries are a critical part of the process of real-time communications. Because protocol endpoints are frequently highly mobile, entities like registrars and proxies enable these entities by maintaining databases of current contact information and routing messages to recipients on behalf of their originators.

This communication pattern puts some constraints on security mechanisms. Messages between the endpoints cannot be entirely encrypted, since some information in protocol messages needs to be visible to proxies in order to route the messages. Such protocols thus typically have a notion of a separation between a "signaling path" comprising information that needs to be available to proxies and a "media path" that belongs only to the endpoints.

These "paths" can be physically or cryptographically separated. In SIP, the two paths are commonly independent IP routes, with signaling messages going through proxies and media messages going directly between the two endpoints. In XMPP, all messages pass through servers, but some parts of the message can be encrypted end-to-end. In either case, there are two separate logical paths for secure data.

HTTP, by contrast, has a much simpler intermediary model. In most cases, the protocol is executed without any intermediaries at all: Since an HTTP URI specifies directly where an HTTP request should be sent, no separate rendezvous service is necessary. Proxies are thus used to provide additional services, such as caching, access control, or anonymization. In secure HTTP transactions, proxies are entirely cut out of the loop, since they simply relay TLS messages (based on an HTTP CONNECT transaction). Figure 1 illustrates the distinction between this pattern of secure channels

and the separated one discussed above.

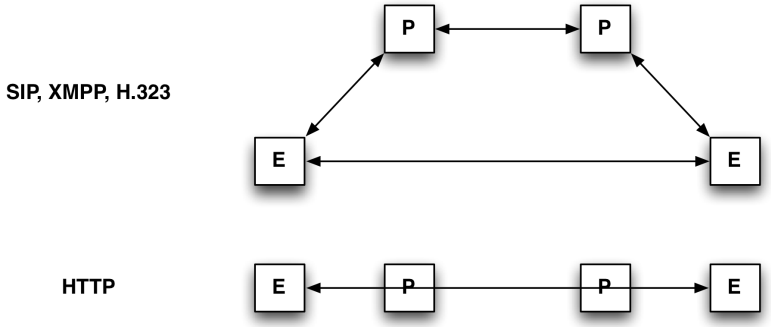

Figure 1: Logical secure paths

Of course, there is an equivalence between intermediaries and attackers, in the sense that each path or intermediary represents a point of compromise. Studies of SIP security, for example, typically consider four classes of attackers: Passive and active attackers, with access to the media or signaling path [RFC5479]. There is some security benefit to this separation of paths, since an adversary with access to only has limited influence over the session (although influence at the signaling layer is clearly more effective). In HTTP, there are only two classes of on-path attackers (active vs. passive), but both have access to the full session. In both cases, there is a well-established tradecraft for inserting intermediaries without users' knowledge; in HTTP, these entities are known as transparent proxies, and in SIP, session border controllers.

In considering real-time communications as an adjunct to the web, then, the question is how to establish a comparable classification of attackers. The answer will depend on how the two systems are integrated. Figure 2 illustrates three examples.

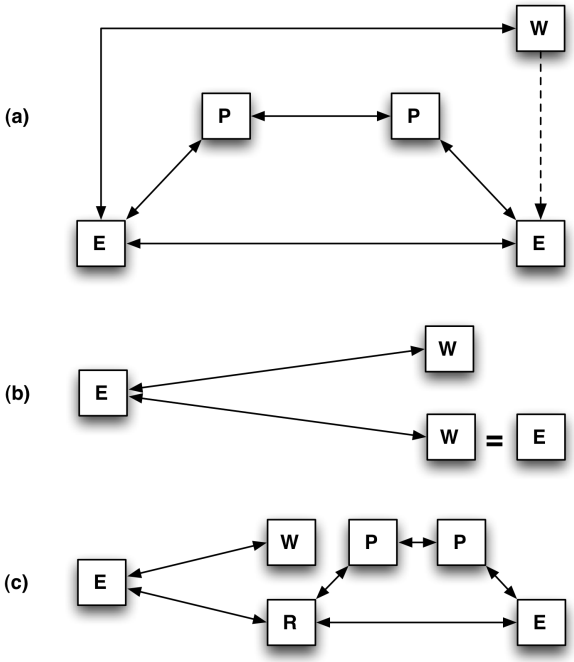

Figure 2: Three models for real-time communications from a browser

In the basic scenario where HTTP simply carries pointers to multimedia resources (accessed using

their own protocols), there can be limited attackers of all six types described above. Just as with SIP, however, all are not equal: On the one hand, an intermediary that only targets or only understands HTTP won't be able to gain access to the multimedia session. The presence of HTTP as a "super-signaling" protocol enables an active attacker on the HTTP session to insert himself at the signaling layer and the media layer of the real-time session.

In contrast, Figure 1(b) illustrates a more pure HTTP model, where an endpoint discovers a remote multimedia endpoint from a link in an HTTP resource and conducts real-time communications with that endpoint over an HTTP streaming mechanism [ref]. In this model, an attacker that only has access to HTTP has visibility into the whole session, but conversely, HTTP security will be much more effective.

Finally, one could also consider an extension to the pure HTTP model, in which real-time signaling media are tunneled over HTTP (e.g., in a WebSocket [ref]) to a relay that forwards them to their destinations. In this model, applying HTTP security would protect the user from redirection attacks by intermediaries in the first HTTP transaction, and would protect the multimedia session from "local attackers" between the user and the relay. Beyond the relay, however, the user's real-time communications would face the normal set of risks, i.e., the possibility of untrusted intermediaries in the signaling and media paths.

Off-Path: Links and Content Indirection

A core aspect of the web is that resources can link and embed other resources, either embedding those resources as part of the original resource (e.g., as images within an HTML page or as XMLHttpRequests) or as links to a separate resource. By allowing pages to be assembled in this way, users and browsers grant a degree of control to authors of web pages. This pattern is largely useable because individual HTTP transactions are fairly low-cost: HTTP uses a normal client-server transaction model over a single TCP connection, typically to transfer less than a megabyte of data. Browsers do assume some risk by giving up control, however, as has become apparent with the increased prevalence of cross-site request forgery attacks.

These risks could become greater as real-time communications become more integrated into the inter-linked fabric of the web. By contrast with HTTP, the real-time communication session are often much more complex, commonly involving multiple peer-to-peer TCP and UDP packet flows, and negotiations to enable these flows to transit firewalls and NATs. The average multimedia session transfers much more information that the average HTTP transaction (many kilobits per second, often over extended time periods). At the same time, real-time communications systems tend to rely on an access control model more akin to email than to the web, which is to say a model based on keeping contact information private rather than applying explicit access controls. In many systems, the default action when the system receives a request to initiate a session is to forward that request to the recipient, who then makes a decision about whether to accept the session.

Given these differences, a few new attacks become feasible. Most obviously, in an environment where web pages direct users to multimedia endpoints, malicious web pages can direct users to malicious endpoints. For example, a high-traffic web page could be used to collect remote SIP UAs to act as amplifiers in the "voice hammer" attack [RFC5245]. There are actually even more straightforward denial of service attacks. If code on a malicious page can direct a browser to initiate

a multimedia session to a victim's contact URI, then the victim could be overwhelmed by requests for sessions, even without an amplification attack. This technique could be especially powerful against systems with user interfaces that produce intrusive alerts for every request they receive, or against automated systems, such as corporate support lines, which are designed to automatically begin a session in response to any request.

The peer-to-peer nature of many real-time systems can also create vulnerabilities. These endpoints could, for example, send mal-formed media packets to the victim host, or act as a media relay to intercept or modify communications with the intended target of the communication. (There is already user-friendly software for the latter class of attack against the Omegle anonymous chat service [http://omegle-spy.googlecode.com/], although Omegle was patched fairly quickly after the tool's release.) Indeed, if an attacker even knows that a session has been opened by a given host – for example, by presenting a link on a malicious web page and observing a click on that link – it knows that that host will be listening on one or more ports, giving the attacker a chance to inject traffic on those ports.