

A decorative graphic consisting of numerous thin, overlapping orange lines that fan out from the left side of the page towards the right, creating a sense of motion and depth.

The XMPP Cloud:

Building a Presence-Enabled Infrastructure for Real-Time Communication

Why Federate? What is the sound of one hand clapping? While you ponder that ancient Buddhist koan, consider this: what is the effectiveness of a single communications silo that prevents you from exchanging presence, messages, voice, video, files, notifications, and other data with people, devices, and applications at other domains?

All of these services and applications gain more power when they can be shared across organizations and service providers. This is the basic tenet of Metcalfe's Law: the value of a communications network is proportional to the square of the number of users of the system. Traditionally, both telephony services and more modern email services have recognized the power of federated communication. You need only one phone, which enables you to call people at any telephony service in the world. Similarly, you need only one email client, which enables you to exchange email messages with people in your own organization but also people at companies, government agencies, and Internet Service Providers (ISPs) all over the world.

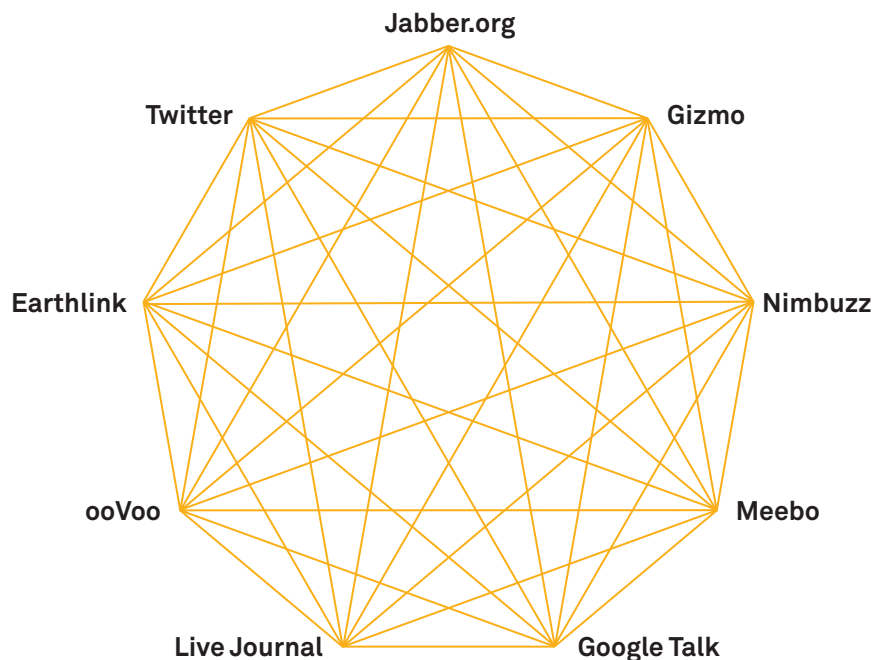
Unfortunately, in the realm of real-time communications, too many enterprises and service providers are satisfied with being a "single hand." It's true that the productivity benefits of deploying instant messaging (IM), Voice over Internet Protocol (VoIP), collaboration, and other presence-enabled technologies within an organization can be significant. But in today's networked world it's usually not enough to communicate only with other employees in your organization or other users of a given service.

Consider the following scenarios:

- » You are a trader at a Wall Street firm and you need to communicate with traders at hedge funds throughout the world.
- » You are a user of an online video chat service that incorporates IM, but you'd like to interact in real time with popular social networking sites that offer IM interfaces to hot new features like microblogging.
- » You are an employee in the Department of Homeland Security and want to receive real-time emergency weather feeds for your area from the National Weather Service.

In all these instances, you may be using a real-time communication service hosted by your employer or a service provider on the Internet, but you also want to tap into the wealth of other people and applications that are hosted at other domains. To do so, you need your “home” service to communicate with such “foreign” services over a dynamically-negotiated link between the services. When services are linked in this way, we say they are “federated.”

This whitepaper describes how inter-domain federation happens using the Jabber Extensible Communications Platform™ (Jabber XCP™). We focus primarily on federation using the Internet Engineering Task Force (IETF) standard Extensible Messaging and Presence Protocol (XMPP), since this protocol is used by a wide range of existing services offered by providers as diverse as Google Talk, Live Journal, Earthlink, Facebook, ooVoo, Meebo, Twitter, the U.S. Marines Corps, National Weather Service, and many others. However, we also look at federation with non-XMPP technologies such as the Session Initiation Protocol (SIP), which is the foundation of popular enterprise messaging systems such as IBM’s Lotus Sametime and Microsoft’s Live Communications Server (LCS) and Office Communications Server (OCS). If you are a service operator, line-of-business manager, security professional, or application developer, this whitepaper will help you understand the benefits and challenges involved in building communication “clouds” using real-time technologies such as XMPP.



» *The open XMPP network*

What Is Federation?

In XMPP usage, federation is the ability of two deployed servers to communicate over a dynamically-established link between the servers.

Federation differs from peering, which requires a prior business-level agreement between two parties before a server-to-server (S2S) link will be established. Peering is more common among traditional telecommunication providers (e.g., because of the relatively high cost of transferring voice traffic), whereas federation is more common on the open Internet (e.g., most email systems are federated dynamically through proper Domain Name System (DNS) settings and email server software configuration).

There are four levels of federation:

Permissive Federation—a server accepts a connection from any other peer on the network, even without verifying the identity of the peer based on DNS lookups or certificate checking. The lack of peer verification or authentication means that domains can be spoofed, opening the door to widespread spam and other abuses. Permissive federation was effectively outlawed on the XMPP network in October 2000 with the release of the open-source jabberd 1.2 server, which included support for the then-new server dialback protocol (fully supported in Jabber XCP).

Verified Federation—a server accepts a connection from a peer only after the identity of the peer has been verified based on information obtained via the DNS and verification of domain-specific keys exchanged in-band. However, the connection is not encrypted. The use of identity verification effectively prevents domain spoofing, but federation requires proper DNS setup and is still subject to DNS poisoning attacks. Verified federation has been the default service policy followed by servers on the open XMPP network from October 2000 to the present.

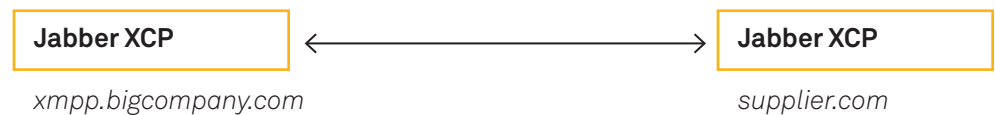
Encrypted Federation—a server accepts a connection from a peer only if the peer supports Transport Layer Security (TLS) as defined for XMPP in Request for Comments (RFC) 3920, and the peer presents a digital certificate. However, the certificate may be self-signed, in which case mutual authentication is typically not possible. Therefore, after TLS negotiation the parties proceed to weakly verify identity using server dialback. This combination (TLS + dialback) results in an encrypted connection with weak identity verification.

Trusted Federation—a server accepts a connection from a peer only if the peer supports TLS and presents a digital certificate issued by a root certification authority (CA) that is trusted by the server. The list of trusted root CAs may be determined by the operating system, XMPP server software, or local service policy. The use of trusted domain certificates effectively prevents DNS poisoning attacks but makes federation more difficult since traditionally such certificates have not been easy to obtain.

Which federation level is acceptable to a particular operator depends on the operator's security policies. However, over time the open XMPP network has been working to upgrade S2S communication so that Encrypted Federation will become the default, with ubiquitous Trusted Federation as the ultimate goal. These efforts are described later in this white paper.

Federation Basics

XMPP servers such as Jabber XCP usually are deployed at fully-qualified domain names (FQDNs) such as `supplier.com` or `xmpp.bigcompany.com` rather than at "raw" machine Internet Protocol (IP) addresses. Such domain names rely on the core Internet infrastructure of name registration and translation known as the DNS. While the DNS can be poisoned and corrupted, successful attacks against the DNS are relatively rare. Therefore most service operators place a fair measure of trust (though not complete trust) in DNS-based verification. The XMPP method for such verification is called server dialback and was originally defined in RFC 3920 (now in XMPP Extension Protocol 0220 (XEP-0220)). The server dialback technology is similar in several ways to the "domain keys" and "sender policy framework" technologies that have been defined for the email network.



» *A direct XMPP connection*

However, XMPP federation differs from email federation in several important ways. First, XMPP does not use multi-hop routing. In email, if `supplier.com` needs to deliver a message to `bigcompany.com` so that a customer service representative (CSR) at the supplier can interact with a buyer at the big company, the message might be routed via an intermediate server at `isp.net` (or any number of such intermediate servers). In XMPP, `supplier.com` performs an appropriate DNS lookup and then opens a direct connection to the XMPP server it has discovered at `bigcompany.com`. The use of direct connections helps to prevent modifications to messages in route, although it also means that XMPP servers need to maintain "always-on" connections to the network and achieve more reliable uptime than is required of email servers.

Second, an XMPP server checks all messages from a peer server to ensure that the "from" address matches the peer domain. If the peer attempts to send a message whose domain does not match, a server must terminate the S2S link with the peer. This helps to prevent message relaying and address spoofing.

For these reasons, it is important to correctly set up DNS to handle XMPP communications. At its most basic, this involves defining a DNS "A" record for the domain (e.g., mapping `supplier.com` to an IP address of `208.68.163.218`) and allowing communication at that IP address on the default S2S port for XMPP (i.e., `5269`). A more advanced configuration would involve defining one or more DNS service location ("SRV") records pointing to particular machines that handle S2S traffic for a domain. (Such an SRV record can be thought of as similar to a Mail exchange ("MX") record as used for email, except using the generic SRV record

format instead of the email-specific MX record format.) For example, a large XMPP service like the fictional isp.net domain might offer two different S2S connection managers (talk1.isp.net and talk2.isp.net), the first one at port 5269 and the second one at port 443.

Therefore the DNS configuration might be as follows:

```
_xmpp-server._tcp.isp.net. 86400 IN SRV 5 0 5269 talk1.isp.net
_xmpp-server._tcp.isp.net. 86400 IN SRV 20 0 443 talk2.isp.net
```

Translated into plain English, this means that isp.net provides S2S XMPP connectivity over Transmission Control Protocol (TCP) via two machines: talk1.isp.net on port 5269 and talk2.isp.net on port 443.

A peer server desiring to negotiate a S2S connection with isp.net would then determine its preferred connection manager (e.g., talk2.isp.net), perform a DNS lookup to map that domain to an IP address (e.g., 208.245.212.98), and connect to the advertised port at that IP address (e.g., 443).

As described below, all of the various federation levels build on these basics to provide verified, encrypted, or trusted communication between two server deployments.

Verified Federation

Verified federation relies most directly on the DNS infrastructure because it actively uses DNS information within the XMPP server dialback protocol.

A helpful analogy for server dialback might be the following telephone scenario:

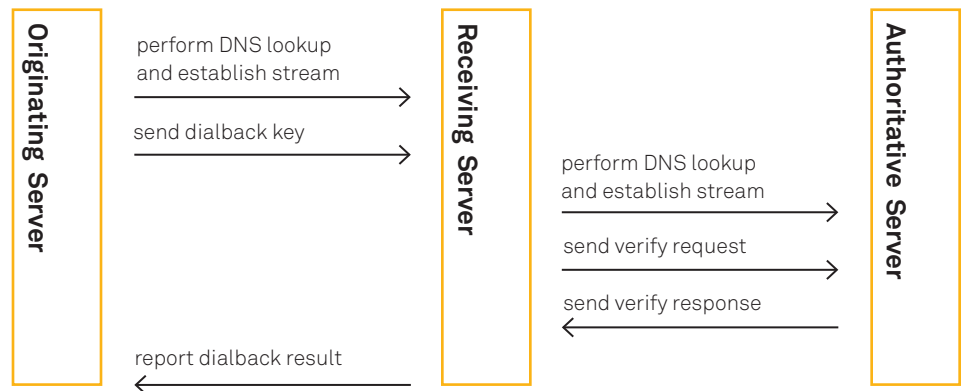
1. CSR from your bank calls you on the phone.
2. Rather than immediately accepting the phone call, you ask for the CSR's employee ID number and put the CSR on hold.
3. You open the phone book, find the authoritative phone number for the bank's headquarters, and give them a call.
4. After being transferred to the customer service department, you ask if a CSR with that particular ID number is authorized to be calling your number.
5. The bank tells you that the CSR is authorized, so you thank them and hang up.
6. You then take the CSR off hold and continue the conversation.

In server dialback, the equivalent of the CSR is the originating server, i.e., the machine that wants to send a message to an entity at a destination domain. The equivalent of the person being called is the receiving server, i.e., the machine to which the originating server has opened a connection for the purpose of sending the message. And the equivalent of the bank is the authoritative server, i.e., the machine that answers to a DNS lookup for the domain asserted by the originating server (which is not necessarily the machine associated with the originating server).

The basic flow of events in server dialback is as follows:

1. The originating server determines the IP and port for the receiving server and establishes an Extensible Markup Language (XML) stream with the receiving server.
2. The originating server sends a key over the connection to the receiving server.
3. The receiving server does not immediately accept the connection but instead determines the IP and port for the authoritative server and establishes an XML stream with the authoritative server.
4. The receiving server sends the same key to the authoritative server for verification.
5. The authoritative server replies that key is valid or invalid.
6. The receiving server informs the originating server whether it is verified or not.

» We can represent this flow of events graphically as follows.



Aside from the “callback” feature (contacting the authoritative server), this technology results in domain verification because the dialback key is based on a shared secret known to the authoritative server’s network—which includes the originating server (at least if it is legitimate).

Encrypted Federation

The concept of encrypted federation builds on verified federation by requiring the use of TLS for channel encryption. TLS is the IETF's formalization of the Secure Sockets Layer (SSL) technology originally developed by Netscape for secure communication over HTTP. Instead of requiring a separate port for secure communication (e.g., port 443 instead of port 80), XMPP uses a TLS profile that enables two entities to upgrade a connection from unencrypted to encrypted. Since XMPP S2S communication uses two connections (one in each direction), encrypted federation requires each server to present a digital certificate to the other party.

In trusted federation, the use of digital certificates results not only in a channel encryption but also in strong authentication. So how is encrypted federation different? The short answer is that not all certificates are created equal—or, to use another metaphor, trust is in the eye of the beholder. For example, I might not trust your digital certificates in the following situations:

» *Your certificate is “self-signed”, i.e., issued by you instead of by a recognized CA*

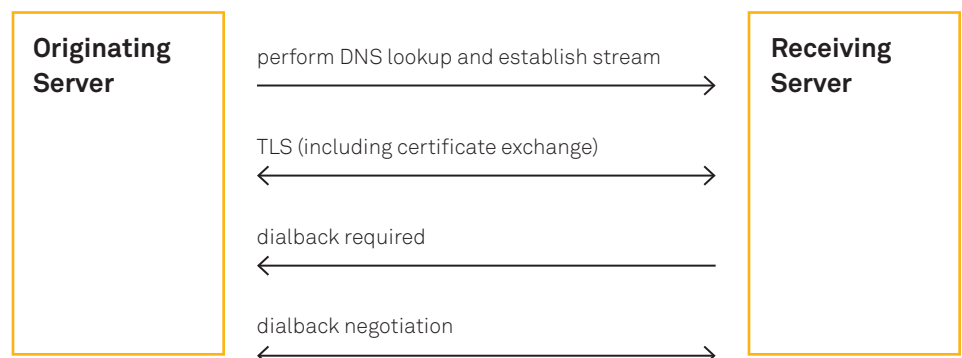
» *Your certificate is issued by a CA, but I don't know or trust the CA*

In these cases, if your server connects to my server, my server would accept your untrusted certificate for the purpose of bootstrapping channel encryption, but not for domain verification (since I have no way of following the certificate chain back to a trusted root). Therefore our servers would complete the TLS negotiation but my server would then require your server to complete server dialback.

The order of events would be as follows:

1. The originating server determines the IP and port for the receiving server and establishes an XML stream with the receiving server.
2. The servers upgrade the stream to an encrypted channel via TLS negotiation, which includes a mutual exchange of certificates.
3. The receiving server determines that the originating server's certificate is not strong enough to use in Simple Authentication and Security Layer (SASL) authentication and therefore indicates that server dialback is required.
4. The servers then complete server dialback negotiation as described above.

» *We can represent this flow of events graphically as follows.*



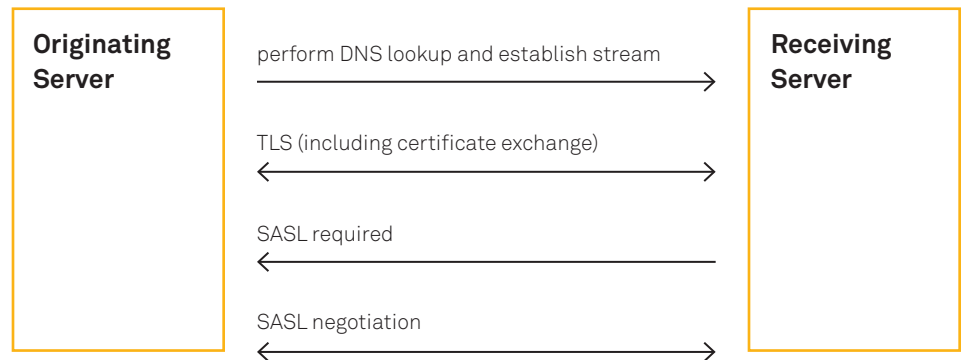
Trusted Federation

Going beyond verified federation and encrypted federation, trusted federation uses TLS for channel encryption and then proceeds to use the SASL protocol for authentication based on the credentials presented in the certificates. In this case, the servers dispense with server dialback entirely because SASL (in particular the EXTERNAL mechanism) provides strong authentication.

The order of events would be as follows:

1. The originating server determines the IP and port for the receiving server and establishes an XML stream with the receiving server.
2. The servers upgrade the stream to an encrypted channel via TLS negotiation, which includes a mutual exchange of certificates.
3. The receiving server determines that the originating server's certificate is strong enough to use in SASL authentication and therefore indicates that SASL negotiation is required.
4. The servers then complete SASL negotiation using the EXTERNAL mechanism.

» We can represent this flow of events graphically as follows.



Because trusted federation does not directly rely on the DNS and because it makes use of certificates issued by trusted authorities, it lays the foundation for a highly secure infrastructure that can host the full range of services offered by XMPP technologies, as well as the various applications that can be created using those services.

Discovering Federated Services and Applications

Server-to-server federation is only the first step toward building a real-time communications cloud. Such a cloud consists of all the users, devices, services, and applications that are connected to the network.

To take full advantage of this network, any given participant needs to ability to find other entities of interest. Such entities might be end users, multi-user chat rooms, real-time content feeds, user directories, data relays, messaging gateways, etc.

XMPP contains a built-in method for finding such entities: service discovery as defined in XEP-0030. This protocol enables any network participant to query another entity regarding its identity, capabilities, and associated entities. Typically, when a participant connects to the network it queries the authoritative server for its domain regarding the entities associated with that server. For example, let us imagine that a user `rep@supplier.com` connects to the `supplier.com` server in order to gain access to the XMPP lit. In response to a service discovery query, the `supplier.com` server would inform the buyer about services hosted at `supplier.com` (say, a file transfer proxy for use by employees of the supplier) and optionally also pointing to services hosted elsewhere (say, a user directory at `industry.org` for all members of an industry association).

Admittedly, it is inefficient to query the entire network each time you connect. That's why XMPP also includes a method for maintaining personal lists of other entities. This "roster" technology enables end users to keep track of people, chat rooms, bots, and other entities they are interested in or interact with regularly.

Furthermore, XMPP deployments within a particular organization or "sub-network" (such as an industry consortium or supply chain) typically include their own directories so that users of those services can quickly and easily find people and applications of interest. Such directories are typically built by connecting to existing databases or regularly polling the network for new entities of interest, much as search engines like Google and Yahoo regularly "spider" the World Wide Web for new content. To date there are no such directories for the Internet-wide XMPP network, but it is probably only a matter of time before such directories emerge.

Protecting and Controlling Federated Communication

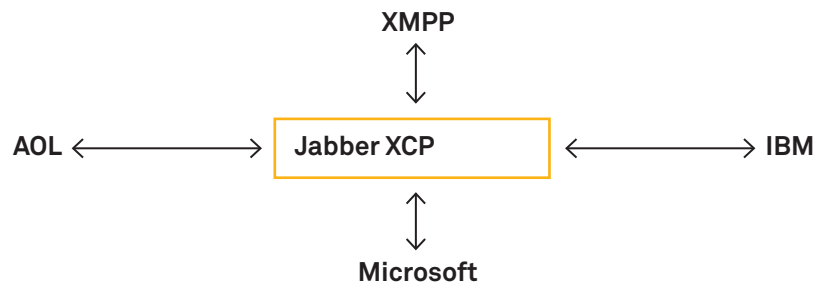
Some organizations are wary of federation because they are concerned that real-time communication networks will introduce problems that have become endemic on the email network, including spam, viruses, abuse, and malware. While these concerns are not unfounded (no one wants to experience a repeat of today's spam problems!), they tend to be overblown, for several reasons:

- 1.** The designers of modern communication technologies such as XMPP learned important lessons from email systems, and incorporated methods for preventing address spoofing, unlimited binary attachments, inline scripts, and other attack vectors.
- 2.** Point-to-point federation avoids several problems with multi-hop federation, including injection attacks, information loss, and unencrypted intermediate links.
- 3.** The use of certificates issued by trusted root CAs results not only in encrypted connections but also strong authentication, which is not currently feasible on the email network.
- 4.** Intelligent servers such as Jabber XCP include the ability to blacklist (explicitly block) and whitelist (explicitly permit) foreign services, either at the hostname level or the IP address level, improving the security profile even further. Such connections can be accepted or rejected at the application layer (by Jabber XCP) or the hardware/network level itself (by an organizational firewall). Typically these connections are easy to manage since XMPP uses a single port (5269) for S2S connectivity.
- 5.** Jabber XCP also provides the ability to control whether local or foreign entities can take advantage of particular features such as chat rooms and file transfer, and even limit such availability to particular roles within an organization. This level of control improves manageability for the purpose of regulatory compliance and confidentiality.

Beyond XMPP

This whitepaper has focused on XMPP technologies because they provide the most robust, stable, secure platform for federation of real-time communication. Many providers already support XMPP federation including popular services such as Google Talk. However, Jabber XCP also supports federation with non-XMPP technologies.

One example of such federation is Jabber, Inc.'s support for communication with the AOL® Instant Messenger™ (AIM®) service, through the AIM® Gateway from Jabber, Inc. component that works hand-in-glove with Jabber XCP. The AIM Gateway from Jabber, Inc. is an AOL-authorized server component that eliminates the legal and technical barriers between XMPP clients and AIM® clients, yet it is completely transparent to Jabber XCP users. It unites Jabber XCP servers directly with the AIM® network, so Jabber XCP users no longer need an AIM® account to communicate with AIM® users. They only need one account, which means that users only distribute one identifying address, remember one password, and maintain a single contact list.



» Jabber XCP Federation

Another compelling technology for cross-technology federation is the SIP/SIMPLE Gateway, which also functions as an add-on component for Jabber XCP. This gateway enables any Jabber XCP deployment to communicate with systems that implement the Session Initiation Protocol (SIP) for Instant Messaging and Presence Leveraging Extensions (SIMPLE). Because SIMPLE is an evolving standard, products from both IBM and Microsoft have filled in the gaps using temporary extensions for commonly-requested features such as contact lists. However, Jabber Inc.'s SIP/SIMPLE Gateway overcomes these obstacles by communicating the particular “flavors” of SIMPLE deployed on the Internet today. As a result, the gateway enables communication with IBM Lotus Sametime and Microsoft LCS or OCS.

The Federated Future

The nuts and bolts of federated communication are just a precursor to building a seamless network or “cloud” of people, devices, information feeds, documents, application interfaces, and other entities. Once you are able to find those entities, determine their identities and roles, request their presence, and discover their capabilities, you can interact with them in real time.

Those interactions can take many forms, including standard one-to-one and multi-party messaging, on-demand alerts and notifications, real-time eventing, media sessions such as voice and video, dynamic workflows, and more.

But these “services” in turn merely provide the building blocks for a wide range of applications, including situation rooms, trading systems, presence-enabled directories, content syndication, microblogging, gaming, polling and voting, voice and video chat, collaborative editing, whiteboarding, telemetry and control of remote devices, network management, real-time interfaces to enterprise resource planning systems, lightweight middleware and web services, and many others.

The power of a federated, presence-enabled communications infrastructure is that it enables software developers and service providers to build and deploy such applications without asking permission from a large, centralized communications operator. As a result, it brings the decentralized energy of the Internet, where radical innovation at the edges is the happy norm, to the more traditional world of voice, messaging, and collaboration. Just as the early developers of the World Wide Web could not have foreseen that the same underlying technologies would eventually power applications such as worldwide fleamarkets, social networking, and virtual worlds, so it is impossible to predict what applications will appear in the emerging real-time communications cloud. However, if the history of the World Wide Web is any guide, we are in for an exciting period of innovation as enterprises, service providers, government agencies, and open-source projects begin to build out the real-time Internet.

Conclusion

Much of the power of any communications technology derives from the power of the network effect. Real-time communications using XMPP are no exception.

From the earliest development of Jabber/XMPP technologies in 1999 to the large and growing XMPP network running on the Internet today, S2S federation for the purpose of inter-domain communication has played a large role in the success of XMPP. Instead of requiring formal business-level agreements between organizations and service providers, XMPP federation relies on a small set of simple but powerful mechanisms for domain checking and security to generate verified, encrypted, and trusted connections between any two deployed servers. These mechanisms have provided a stable, secure foundation for organic growth of the XMPP network and similar real-time technologies.

References

The following specifications provide further information regarding XMPP federation and may be of interest to technical readers.

IETF RFC 3920: Extensible Messaging and Presence Protocol (XMPP): Core. URL: <http://www.ietf.org/rfc/rfc3920.txt>

IETF RFC 3921: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence. URL: <http://www.ietf.org/rfc/rfc3921.txt>

XSF XEP-0030: Service Discovery. URL: <http://www.xmpp.org/extensions/xep-0030.html>

XSF XEP-0220: Server Dialback. URL: <http://www.xmpp.org/extensions/xep-0220.html>

XSF XEP-0238: XMPP Protocol Flows for Inter-Domain Federation. URL: <http://www.xmpp.org/extensions/xep-0238.html>