OpenID

OpenID is an open standard and decentralized authentication protocol promoted by the non-profit OpenID Foundation. It allows users to be authenticated by cooperating sites (known as relying parties, or RP) using a third-party identity provider (IDP) service, eliminating the need for webmasters to provide their own *ad hoc* login systems, and allowing users to log in to multiple unrelated websites without having to have a separate identity and password for each.[1] Users create accounts by selecting an OpenID identity provider,[1] and then use those accounts to sign on to any website that accepts OpenID authentication. Several large organizations either issue or accept OpenIDs on their websites.[2]

The OpenID logo

The OpenID standard provides a framework for the communication that must take place between the identity provider and the OpenID acceptor (the "relying party").[3] An extension to the standard (the OpenID Attribute Exchange) facilitates the transfer of user attributes, such as name and gender, from the OpenID identity provider to the relying party (each relying party may request a different set of attributes, depending on its requirements).[4] The OpenID protocol does not rely on a central authority to authenticate a user's identity. Moreover, neither services nor the OpenID standard may mandate a specific means by which to authenticate users, allowing for approaches ranging from the common (such as passwords) to the novel (such as smart cards or biometrics).

The final version of OpenID is OpenID 2.0, finalized and published in December 2007.[5] The term *OpenID* may also refer to an identifier as specified in the OpenID standard; these identifiers take the form of a unique Uniform Resource Identifier (URI), and are managed by some "OpenID provider" that handles authentication.[1]

Contents

Adoption

As of March 2016, there are over 1 billion OpenID-enabled accounts on the Internet (see below) and approximately 1,100,934 sites have integrated OpenID consumer support:[6] AOL, Flickr, Google, Amazon.com, Canonical (provider name Ubuntu One), LiveJournal, Microsoft (provider name Microsoft account), Mixi, Myspace, Novell, OpenStreetMap, Orange, Sears, Sun, Telecom Italia, Universal Music Group, VeriSign, WordPress, Yahoo!, the BBC,[7] IBM,[8] PayPal,[9] and Steam,[10] although some of those organizations also have their own authentication management.

Many if not all of the larger organizations require users to provide authentication in the form of an existing email account or mobile phone number in order to sign up for an account (which then can be used as an OpenID identity). There are several smaller entities that accept sign-ups with no extra identity details required.

Facebook did use OpenID in the past, but moved to Facebook Connect.[11] Blogger also used OpenID, but since May 2018 no longer supports it.[12]

Technical overview

An *end user* is the entity that wants to assert a particular identity. A *relying party* (RP) is a web site or application that wants to verify the end user's identifier. Other terms for this party include "service provider" or the now obsolete "consumer". An identity provider, or *OpenID provider* (OP) is a service that specializes in registering OpenID URLs or XRIs. OpenID enables an end user to communicate with a relying party. This communication is done through the exchange of an identifier or *OpenID*, which is the URL or XRI chosen by the end user to name the end user's identity. An identity provider provides the OpenID authentication (and possibly other identity services). The exchange is enabled by a *user-agent*, which is the program (such as a browser) used by the end user to communicate with the relying party and OpenID provider.

Logging in

The end user interacts with a relying party (such as a website) that provides an option to specify an OpenID for the purposes of authentication; an end user typically has previously registered an OpenID (e.g. `alice.openid.example.org`) with an OpenID provider (e.g. `openid.example.org`).[1]

The relying party typically transforms the OpenID into a canonical URL form (e.g. `http://alice.openid.example.org/`).

- With OpenID 1.0, the relying party then requests the HTML resource identified by the URL and reads an HTML link tag to discover the OpenID provider's URL (e.g.

`http://openid.example.org/openid-auth.php`). The relying party also discovers whether to use a *delegated identity* (see below).
- With OpenID 2.0, the relying party discovers the OpenID provider URL by requesting the *XRDS document* (also called the *Yadis document*) with the content type `application/xrds+xml`; this document may be available at the target URL and is always available for a target XRI.

There are two modes in which the relying party may communicate with the OpenID provider:

- `checkid_immediate`, in which the relying party requests that the OpenID provider not interact with the end user. All communication is relayed through the end user's user-agent without explicitly notifying the end user.
- `checkid_setup`, in which the end user communicates with the OpenID provider via the same user-agent used to access the relying party.

The `checkid_immediate` mode can fall back to the `checkid_setup` mode if the operation cannot be automated.

First, the relying party and the OpenID provider (optionally) establish a shared secret, referenced by an *associate handle*, which the relying party then stores. If using the `checkid_setup` mode, the relying party redirects the end user's user-agent to the OpenID provider so the end user can authenticate directly with the OpenID provider.

The method of authentication may vary, but typically, an OpenID provider prompts the end user for a password or some cryptographic token, and then asks whether the end user trusts the relying party to receive the necessary identity details.

If the end user declines the OpenID provider's request to trust the relying party, then the user-agent is redirected back to the relying party with a message indicating that authentication was rejected; the relying party in turn refuses to authenticate the end user.

If the end user accepts the OpenID provider's request to trust the relying party, then the user-agent is redirected back to the relying party along with the end user's credentials. That relying party must then confirm that the credentials really came from the OpenID provider. If the relying party and OpenID provider had previously established a shared secret, then the relying party can validate the identity of the OpenID provider by comparing its copy of the shared secret against the one received along with the end user's credentials; such a relying party is called *stateful* because it stores the shared secret between sessions. In contrast, a *stateless* or *dumb* relying party must make one more background request (`check_authentication`) to ensure that the data indeed came from the OpenID provider.

After the OpenID has been verified, authentication is considered successful and the end user is considered logged into the relying party under the identity specified by the given OpenID (e.g. `alice.openid.example.org`). The relying party typically then stores the end user's OpenID along with the end user's other session information.

Identifiers

To obtain an OpenID-enabled URL that can be used to log into OpenID-enabled websites, a user registers an OpenID identifier with an identity provider. Identity providers offer the ability to register a URL (typically a third-level domain, e.g. username.example.com) that will automatically be configured with OpenID authentication service.

Once they have registered an OpenID, a user can also use an existing URL under their own control (such as a blog or home page) as an alias or "delegated identity". They simply insert the appropriate OpenID tags in the HTML[13] or serve a Yadis document.[14]

Starting with OpenID Authentication 2.0 (and some 1.1 implementations), there are two types of identifiers that can be used with OpenID: URLs and XRIs.

XRIs are a new form of Internet identifier designed specifically for cross-domain digital identity. For example, XRIs come in two forms—i-names and i-numbers—that are usually registered simultaneously as synonyms. I-names are reassignable (like domain names), while i-numbers are never reassigned. When an XRI i-name is used as an OpenID identifier, it is immediately resolved to the synonymous i-number (the CanonicalID element of the XRDS document). This i-number is the OpenID identifier stored by the relying party. In this way, both the user and the relying party are protected from the end user's OpenID identity ever being taken over by another party as can happen with a URL based on a reassignable DNS name.

OpenID Foundation

The OpenID Foundation (OIDF) promotes and enhances the OpenID community and technologies. The OIDF is a non-profit international standards development organization of individual developers, government agencies and companies who wish to promote and protect OpenID. The OpenID Foundation was formed in June 2007 and serves as a public trust organization representing an open community of developers, vendors and users. OIDF assists the community by providing needed infrastructure and help in promoting and supporting adoption of OpenID. This includes managing intellectual property and trade marks as well a fostering viral growth and global participation in OpenID.

People

The OpenID Foundation's board of directors has four community members and eight corporate members:[15]

Community board members

- Chairman: Nat Sakimura (Nomura Research Institute)
- Treasurer: John Bradley (Ping Identity)
- Secretary: Mike Jones (Microsoft)
- Community Representative: George Fletcher (AOL)

Corporate board members

- Google – Adam Dawes
- Janrain – Jim Kaskade
- Microsoft – Anthony Nadalin
- Oracle – Prateek Mishra
- Ping Identity – Pam Dingle
- Symantec – Brian Berliner
- US Department of Health & Human Services, Office of the National Coordinator – Debbie Bucci
- Verizon – Bjorn Hjelm
- VMware – Ashish Jain

Chapters

OIDF is a global organization to promote digital identity and to encourage the further adoption of OpenID, the OIDF has encouraged the creation of member chapters. Member chapters are officially part of the Foundation and work within their own constituency to support the development and adoption of OpenID as a framework for user-centric identity on the internet.

Intellectual property and contribution agreements

The OIDF ensures that OpenID specifications are freely implementable therefore the OIDF requires all contributors to sign a contribution agreement. This agreement both grants a copyright license to the Foundation to publish the collective specifications and includes a patent non-assertion agreement. The non-assertion agreement states that the contributor will not sue someone for implementing OpenID specifications.

Legal issues

The OpenID trademark in the United States was assigned to the OpenID Foundation in March 2008.[16] It had been registered by NetMesh Inc. before the OpenID Foundation was operational.[17][18] In Europe, as of August 31, 2007, the OpenID trademark is registered to the OpenID Europe Foundation.[19]

The OpenID logo was designed by Randy "ydnar" Reddig, who in 2005 had expressed plans to transfer the rights to an OpenID organization.[20]

Since the original announcement of OpenID, the official site has stated:[21]

> Nobody should own this. Nobody's planning on making any money from this. The goal is to release every part of this under the most liberal licenses possible, so there's no money or licensing or registering required to play. It benefits the community as a whole if something like this exists, and we're all a part of the community.

Sun Microsystems, VeriSign and a number of smaller companies involved in OpenID have issued patent non-assertion covenants covering OpenID 1.1 specifications. The covenants state that the companies will not assert any of their patents against OpenID implementations and will revoke their promises from anyone who threatens, or asserts, patents against OpenID implementors.[22][23]

Security

Authentication bugs

In March, 2012, a research paper[24] reported two generic security issues in OpenID. Both issues allow an attacker to sign in to a victim's relying party accounts. For the first issue, OpenID and Google (an Identity Provider of OpenID) both published security advisories to address it.[25][26] Google's advisory says "An attacker could forge an OpenID request that doesn't ask for the user's email address, and then insert an unsigned email address into the IDPs response. If the attacker relays this response to a website that doesn't notice that this attribute is unsigned, the website may be tricked into logging the attacker in to any local account." The research paper claims that many

popular websites have been confirmed vulnerable, including Yahoo! Mail, smartsheet.com, Zoho, manymoon.com, diigo.com. The researchers have notified the affected parties, who have then fixed their vulnerable code.

For the second issue, the paper called it "Data Type Confusion Logic Flaw", which also allows attackers to sign in to victims' RP accounts. Google and PayPal were initially confirmed vulnerable. OpenID published a vulnerability report[27] on the flaw. The report says Google and PayPal have applied fixes, and suggest other OpenID vendors to check their implementations.

Phishing

Some observers have suggested that OpenID has security weaknesses and may prove vulnerable to phishing attacks.[28][29][30] For example, a malicious relaying party may forward the end user to a bogus identity provider authentication page asking that end user to input their credentials. On completion of this, the malicious party (who in this case also controls the bogus authentication page) could then have access to the end user's account with the identity provider, and then use that end user's OpenID to log into other services.

In an attempt to combat possible phishing attacks, some OpenID providers mandate that the end user needs to be authenticated with them prior to an attempt to authenticate with the relying party.[31] This relies on the end user knowing the policy of the identity provider. In December 2008, the OpenID Foundation approved version 1.0 of the Provider Authentication Policy Extension (PAPE), which "enables Relying Parties to request that OpenID Providers employ specified authentication policies when authenticating users and for OpenID Providers to inform the Relying Parties which policies were actually used."[32]

Privacy and trust issues

Other security issues identified with OpenID involve lack of privacy and failure to address the trust problem.[33] However, this problem is not unique to OpenID and is simply the state of the Internet as commonly used.

The Identity Provider does, however, get a log of your OpenID logins; they know when you logged into what website, making cross-site tracking much easier. A compromised OpenID account is also likely to be a more serious breach of privacy than a compromised account on a single site.

Authentication hijacking in unsecured connection

Another important vulnerability is present in the last step in the authentication scheme when TLS/SSL are not used: the redirect-URL from the identity provider to the relying party. The problem with this redirect is the fact that anyone who can obtain this URL (e.g. by sniffing the wire) can replay it and get logged into the site as the victim user. Some of the identity providers use nonces (number used once) to allow a user to log into the site once and fail all the consecutive attempts. The nonce solution works if the user is the first one to use the URL. However, a fast attacker who is sniffing the wire can obtain the URL and immediately reset a user's TCP connection (as an attacker is sniffing the wire and knows the required TCP sequence numbers) and then execute the replay attack as described above. Thus nonces only protect against passive attackers, but cannot prevent active attackers from executing the replay attack.[34] Use of TLS/SSL in the authentication process can significantly reduce this risk.

This can be restated as:

```
IF (Both RP1 and RP2 have Bob as a client) AND       // a common case
   (Bob uses the same IDP with both RP1 and RP2) AND // a common case
   (RP1 does not use VPN/SSL/TLS to secure their connection with the client) // preventable!
THEN
  RP2 could obtain credentials sufficient to impersonate Bob with RP1
END-IF
```

Covert Redirect

On May 1, 2014, a bug dubbed "Covert Redirect related to OAuth 2.0 and OpenID" was disclosed.[35][36] It was discovered by mathematics doctoral student Wang Jing at the School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore.[37][38][39]

The announcement of OpenID is: "'Covert Redirect', publicized in May 2014, is an instance of attackers using open redirectors – a well-known threat, with well-known means of prevention. The OpenID Connect protocol mandates strict measures that preclude open redirectors to prevent this vulnerability."[40]

"The general consensus, so far, is that Covert Redirect is not as bad, but still a threat. Understanding what makes it dangerous requires a basic understanding of Open Redirect, and how it can be exploited."[41]

A patch was not immediately made available. Ori Eisen, founder, chairman and chief innovation officer at 41st Parameter told Sue Marquette Poremba, "In any distributed system, we are counting of the good nature of the participants to do the right thing. In cases like OAuth and OpenID, the distribution is so vast that it is unreasonable to expect each and every website to patch up in the near future".[42]

History

The original OpenID authentication protocol was developed in May 2005[43] by Brad Fitzpatrick, creator of popular community website LiveJournal, while working at Six Apart.[44] Initially referred to as Yadis (an acronym for "Yet another distributed identity system"),[45] it was named OpenID after the openid.net domain name was given to Six Apart to use for the project.[46] OpenID support was soon implemented on LiveJournal and fellow LiveJournal engine community DeadJournal for blog post comments and quickly gained attention in the digital identity community.[47][48] Web developer JanRain was an early supporter of OpenID, providing OpenID software libraries and expanding its business around OpenID-based services.

In late June, discussions started between OpenID users and developers from enterprise software company NetMesh, leading to collaboration on interoperability between OpenID and NetMesh's similar Light-weight Identity (LID) protocol. The direct result of the collaboration was the Yadis discovery protocol, adopting the name originally used for OpenID. The new Yadis was announced on October 24, 2005.[49] After a discussion at the 2005 Internet Identity Workshop (http://iiw.idcommons.net) a few days later, XRI/i-names developers joined the Yadis project,[50] contributing their Extensible Resource Descriptor Sequence (XRDS) format for utilization in the protocol.[51]

In December, developers at Sxip Identity began discussions with the OpenID/Yadis community[52] after announcing a shift in the development of version 2.0 of its Simple Extensible Identity Protocol (SXIP) to URL-based identities like LID and OpenID.[53] In March 2006, JanRain developed a Simple Registration (SREG) extension for OpenID enabling primitive profile-exchange[54] and in April submitted a proposal to formalize extensions to OpenID. The same month, work had also begun on incorporating full XRI support into OpenID.[55] Around early

May, key OpenID developer David Recordon left Six Apart, joining VeriSign to focus more on digital identity and guidance for the OpenID spec.[48][56] By early June, the major differences between the SXIP 2.0 and OpenID projects were resolved with the agreement to support multiple personas in OpenID by submission of an identity provider URL rather than a full identity URL. With this, as well as the addition of extensions and XRI support underway, OpenID was evolving into a full-fledged digital identity framework, with Recordon proclaiming "We see OpenID as being an umbrella for the framework that encompasses the layers for identifiers, discovery, authentication and a messaging services layer that sits atop and this entire thing has sort of been dubbed 'OpenID 2.0'.[57] " In late July, Sxip began to merge its Digital Identity Exchange (DIX) protocol into OpenID, submitting initial drafts of the OpenID Attribute Exchange (AX) extension in August. Late in 2006, a ZDNet opinion piece made the case for OpenID to users, web site operators and entrepreneurs.[58]

On January 31, 2007, Symantec announced support for OpenID in its Identity Initiative products and services.[59] A week later, on February 6 Microsoft made a joint announcement with JanRain, Sxip, and VeriSign to collaborate on interoperability between OpenID and Microsoft's Windows CardSpace digital identity platform, with particular focus on developing a phishing-resistant authentication solution for OpenID. As part of the collaboration, Microsoft pledged to support OpenID in its future identity server products and JanRain, Sxip, and VeriSign pledged to add support for Microsoft's Information Card profile to their future identity solutions.[60] In mid-February, AOL announced that an experimental OpenID provider service was functional for all AOL and AOL Instant Messenger (AIM) accounts.[61]

In May, Sun Microsystems began working with the OpenID community, announcing an OpenID program,[62] as well as entering a non-assertion covenant with the OpenID community, pledging not to assert any of its patents against implementations of OpenID.[22] In June, OpenID leadership formed the OpenID Foundation, an Oregon-based public benefit corporation for managing the OpenID brand and property.[63] The same month, an independent OpenID Europe Foundation was formed in Belgium[64] by Snorri Giorgetti. By early December, non-assertion agreements were collected by the major contributors to the protocol and the final OpenID Authentication 2.0 and OpenID Attribute Exchange 1.0 specifications were ratified on December 5.[65]

In mid-January 2008, Yahoo! announced initial OpenID 2.0 support, both as a provider and as a relying party, releasing the provider service by the end of the month.[66] In early February, Google, IBM, Microsoft, VeriSign and Yahoo! joined the OpenID Foundation as corporate board members.[67] Around early May, SourceForge, Inc. introduced OpenID provider and relying party support to leading open source software development website SourceForge.net.[68] In late July, popular social network service MySpace announced support for OpenID as a provider.[69] In late October, Google launched support as an OpenID provider and Microsoft announced that Windows Live ID would support OpenID.[70] In November, JanRain announced a free hosted service, RPX Basic, that allows websites to begin accepting OpenIDs for registration and login without having to install, integrate and configure the OpenID open source libraries.[71]

In January 2009, PayPal joined the OpenID Foundation as a corporate member, followed shortly by Facebook in February. The OpenID Foundation formed an executive committee and appointed Don Thibeau as executive director. In March, MySpace launched their previously announced OpenID provider service, enabling all MySpace users to use their MySpace URL as an OpenID. In May, Facebook launched their relying party functionality,[72][73] letting users use an automatic login-enabled OpenID account (e.g. Google) to log into Facebook.[74]

In September 2013, Janrain announced that MyOpenID.com would be shut down on February 1, 2014; a pie chart showed Facebook and Google dominate the social login space as of Q2 2013.[75] Facebook has since left OpenID; it is no longer a sponsor, represented on the board, or permitting OpenID logins.[15][76]

In May 2016, Symantec announced that they would be discontinuing their pip.verisignlabs.com OpenID personal identity portal service.[77][78]

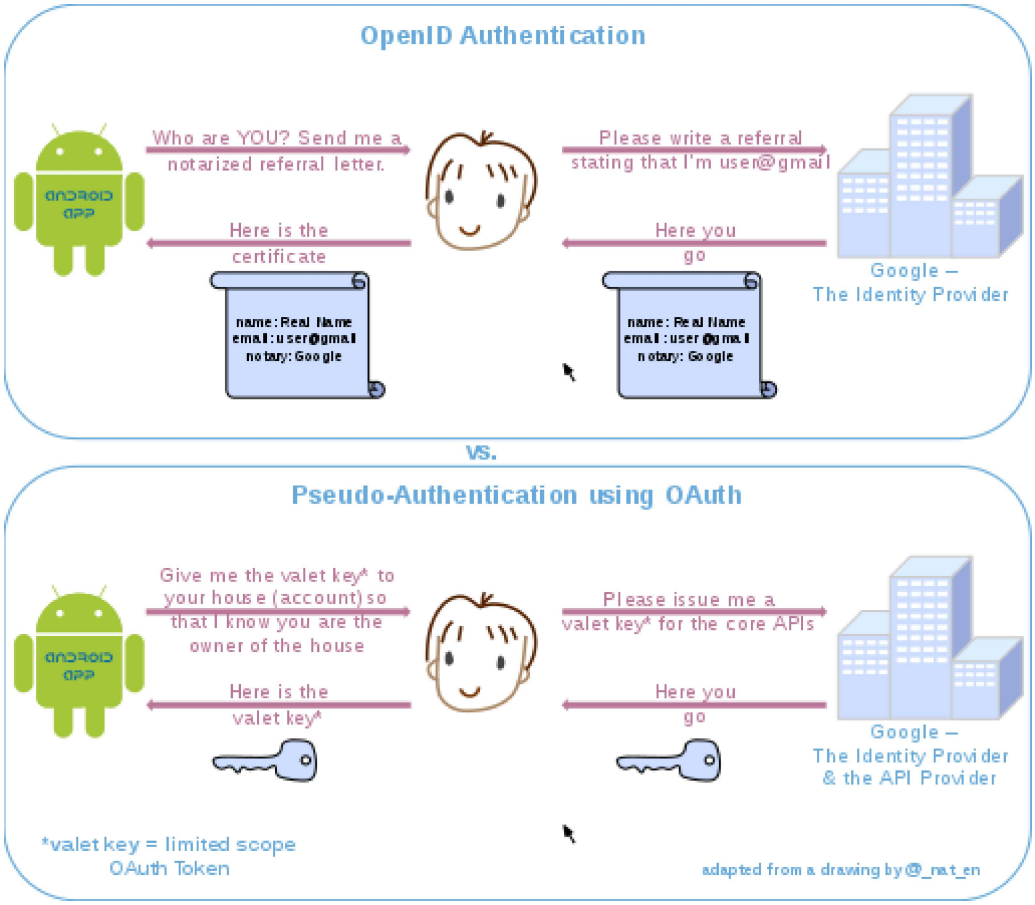
In March 2018, Stack Overflow announced an end to OpenID support, citing insufficient usage to justify the cost. In the announcement, it was stated that based on activity, users strongly preferred Facebook, Google, and e-mail/password based account authentication.[79]

OpenID versus pseudo-authentication using OAuth

OpenID is a way to use a single set of user credentials to access multiple sites, while OAuth facilitates the authorization of one site to access and use information related to the user's account on another site. Although OAuth is not an authentication protocol, it can be used as part of one.

> Authentication in the context of a user accessing an application tells an application who the current user is and whether or not they're present. [...] Authentication is all about the user and their presence with the application, and an internet-scale authentication protocol needs to be able to do this across network and security boundaries.
>
> However, OAuth tells the application none of that. OAuth says absolutely nothing about the user, nor does it say how the user proved their presence or even if they're still there. As far as an OAuth client is concerned, it asked for a token, got a token, and eventually used that token to access some API. It doesn't know anything about who authorized the application or if there was even a user there at all. In fact, much of the point of OAuth is about giving this delegated access for use in situations where the user is not present on the connection between the client and the resource being accessed. This is great for client authorization, but it's really bad for authentication where the whole point is figuring out if the user is there or not (and who they are).[80]

The following drawing highlights the differences between using OpenID versus OAuth for authentication. Note that with OpenID, the process starts with the application asking the user for their identity (typically an OpenID URI), whereas in the case of OAuth, the application directly requests a limited access OAuth Token (valet key) to access the APIs (enter the house) on user's behalf. If the user can grant that access, the application can retrieve the unique identifier for establishing the profile (identity) using the APIs.



Attack against pseudo-authentication

OpenID provides a cryptographic verification mechanism that prevents the attack below against users who misuse OAuth for authentication.

Note that the valet key does not describe the user in any way, it only provides limited access rights, to some house (which is not even necessarily the user's, they just had a key). Therefore if the key becomes compromised (the user is malicious and managed to steal the key to someone else's house), then the user can impersonate the house owner to the application who requested their authenticity. If the key is compromised by any point in the chain of trust, a malicious user may intercept it and use it to impersonate user X for any application relying on OAuth2 for pseudo authentication against the same OAuth authorization server. Conversely, the notarized letter contains the user's signature, which can be checked by the requesting application against the user, so this attack is not viable. [81]

Verifying the letter

The letter can use public-key cryptography to be authenticated.

- The requesting application provides its encryption public key to the user, which provides it to the authentication server.
- The authentication server encrypts a document containing an encryption key which corresponds to a one-way hash of a secret the user knows (e.g. passphrase) for challenge–response using the application's public key.
- The user passes the encrypted document back to the application, which decrypts it.
- The application encrypts a random phrase using the received encryption key, and asks that the user do the same, then compares the results, if they match, the user is authentic.

OpenID Connect (OIDC)

Published in February 2014 by the OpenID Foundation, OpenID Connect is the third generation of OpenID technology. It is an authentication layer on top of the OAuth 2.0 authorization framework.[82] It allows computing clients to verify the identity of an end user based on the authentication performed by an authorization server, as well as to obtain the basic profile information about the end user in an interoperable and REST-like manner. In technical terms, OpenID Connect specifies a RESTful HTTP API, using JSON as a data format.

OpenID Connect allows a range of parties, including web-based, mobile and JavaScript clients, to request and receive information about authenticated sessions and end users. The OpenID Connect specification is extensible, supporting optional features such as encryption of identity data, discovery of OpenID providers, and session management.

See also

- Authorization
- Athens access and identity management
- BrowserID
- Central Authentication Service
- IndieAuth
- Information Card
- Liberty Alliance
- Light-weight Identity
- SAML
- Shibboleth (Shibboleth Consortium)
- Single sign-on
- SQRL
- WebFinger
- WebID
- WS-Federation

References

1. Eldon, Eric (2009-04-14). "Single sign-on service OpenID getting more usage" (https://venturebeat.com/2009/04/14/single-sign-on-service-openid-getting-more-usage/). venturebeat.com. Retrieved 2009-04-25.
2. "What is an OpenID?" (http://openid.net/get-an-openid/what-is-openid/). Retrieved 19 June 2014.
3. "OpenID Authentication 2.0 specification – Final" (http://openid.net/specs/openid-authentication-2_0.html). Retrieved 2011-10-24.
4. "OpenID Attribute Exchange 1.0 – Final" (http://openid.net/specs/openid-attribute-exchange-1_0.html). Retrieved 2011-10-24.
5. "OpenID Authentication 2.0 - Final" (http://openid.net/specs/openid-authentication-2_0.html). 2007-12-05. Retrieved 2014-05-18.
6. "OpenID Usage Statistics" (https://trends.builtwith.com/docinfo/OpenID).
7. bashburn, bill (2008-04-22). "BBC Joins OpenID Foundation" (http://openid.net/2008/04/22/british-broadcasting-corp-bbc-joins-openid-foundation/).
8. "Technology Leaders Join OpenID Foundation to Promote Open Identity Management on the Web" (http://www-03.ibm.com/press/us/en/pressrelease/23461.wss). 2008-02-07.
9. "PayPal Access Uses OpenID 2.0" (http://openid.net/2011/10/19/paypal-access-uses-openid-2-0/). OpenID ·. Retrieved 19 June 2014.
10. "Steam Community :: Steam Web API Documentation" (http://steamcommunity.com/dev). Retrieved 2012-02-10.
11. Perez, Juan Carlos. "Facebook, Google launch data portability programs to all" (http://www.networkworld.com/article/2270803/data-center/facebook--google-launch-data-portability-programs-to-all.html). Network World, Inc. Retrieved 19 June 2014.

12. "It's spring cleaning time for Blogger" (https://blogger.googleblog.com/2018/05/its-spring-cleaning-time-for-blogger.html). Blogger team. Retrieved 10 September 2019.
13. "OpenID Authentication 1.1#Delegation" (http://openid.net/specs/openid-authentication-1_1.html#delegating_authentication).
14. Paul Tarjan. "Easy OpenID Delegation with Yadis" (https://web.archive.org/web/20090704234010/http://blog.paulisageek.com/2009/06/easy-openid-delegation-with-yadis.html). Archived from the original (http://blog.paulisageek.com/2009/06/easy-openid-delegation-with-yadis.html) on 2009-07-04. Retrieved 2009-06-30.
15. "Leadership" (http://openid.net/foundation/leadership/). openID Foundation. Retrieved 19 June 2014.
16. "Trademark Assignment, Serial #: 78899244" (http://assignments.uspto.gov/assignments/q?db=tm&sno=78899244). United States Patent and Trademark Office. 2008-05-06. Retrieved 2008-05-19. "Exec Dt: 03/27/2008"
17. "Latest Status Info" (http://tarr.uspto.gov/servlet/tarr?regser=serial&entry=78899244). United States Patent and Trademark Office. 2006-03-27. Retrieved 2008-03-20.
18. "NetMesh: Company / Management" (https://web.archive.org/web/20070830095651/http://netmesh.us/company/management/). *NetMesh*. Archived from the original (http://netmesh.us/company/management/) on 2007-08-30. Retrieved 2008-03-20.
19. "OpenID Europe Trademark & Logo Policy" (https://web.archive.org/web/20080309221914/http://www.openideurope.eu/policies/openid-trademark-policy/). *OpenID Europe Foundation*. Archived from the original (http://www.openideurope.eu/policies/openid-trademark-policy/) on 2008-03-09. Retrieved 2008-03-20.
20. Reddig, Randy (2005-06-29). "OpenID Logo" (http://lists.danga.com/pipermail/yadis/2005-June/000990.html). *Danga Interactive*. Retrieved 2008-03-20.
21. Fitzpatrick, Brad. "Intellectual Property" (http://openid.net/intellectual-property/).
22. "Sun OpenID: Non-Assertion Covenant" (http://www.sun.com/software/standards/persistent/openid/nac.xml). *Sun Microsystems*. Retrieved 2008-03-20.
23. "VeriSign's OpenID Non-Assertion Patent Covenant" (https://web.archive.org/web/20080415211645/http://www.verisign.com/research/Consumer_Identity_and_Profile_Management/042160.html). *VeriSign*. Archived from the original (http://www.verisign.com/research/Consumer_Identity_and_Profile_Management/042160.html) on 2008-04-15. Retrieved 2008-03-20.
24. Rui Wang; Shuo Chen & XiaoFeng Wang. "Signing Me onto Your Accounts through Facebook and Google: a Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services" (http://research.microsoft.com/apps/pubs/default.aspx?id=160659).
25. "Attribute Exchange Security Alert" (http://openid.net/2011/05/05/attribute-exchange-security-alert/).
26. "Security advisory to websites using OpenID Attribute Exchange" (http://googlecode.blogspot.com/2011/05/security-advisory-to-websites-using.html).
27. "Vulnerability report: Data confusion" (http://openid.net/2012/03/14/vulnerability-report-data-confusion/).
28. Crowley, Paul (2005-06-01). "Phishing attacks on OpenID" (http://lists.danga.com/pipermail/yadis/2005-June/000470.html). *Danga Interactive*. Retrieved 2008-03-20.
29. Anderson, Tim (2007-03-05). "OpenID still open to abuse" (http://www.itweek.co.uk/2184695). IT Week. Retrieved 2007-03-13.
30. Slot, Marco. "Beginner's guide to OpenID phishing" (http://openid.marcoslot.net/). Retrieved 2007-07-31.
31. "Verisign PIP FAQ" (https://web.archive.org/web/20081113065317/https://pip.verisignlabs.com/faq.do#faq5). Archived from the original (https://pip.verisignlabs.com/faq.do#faq5) on 2008-11-13. Retrieved 2008-11-13.
32. Jones, Mike. "PAPE Approved as an OpenID Specification" (http://openid.net/2008/12/31/pape-approved-as-an-openid-specification/). OpenID Foundation.

33. Stefan Brands (2007-08-22). "The problem(s) with OpenID" (https://web.archive.org/web/20110516013258/http://www.untrusted.ca/cache/openid.html). Archived from the original (http://www.untrusted.ca/cache/openid.html) on 2011-05-16. Retrieved 2010-12-12. (originally published on The Identity Corner at www.idcorner.org/?p=161)
34. Tsyrklevich, Eugene. "Single Sign-On for the Internet: A Security Story" (https://www.blackhat.com/presentations/bh-usa-07/Tsyrklevich/Whitepaper/bh-usa-07-tsyrklevich-WP.pdf) (PDF). Blackhat USA. Retrieved 2012-04-19.
35. "Serious security flaw in OAuth, OpenID discovered" (https://www.cnet.com/news/serious-security-flaw-in-oauth-and-openid-discovered/). CNET. 2 May 2014. Retrieved 10 November 2014.
36. "Covert Redirect" (http://tetraph.com/covert_redirect/). Tetraph. 1 May 2014. Retrieved 10 November 2014.
37. "Facebook, Google Users Threatened by New Security Flaw" (https://news.yahoo.com/facebook-google-users-threatened-security-192547549.html). Yahoo. 2 May 2014. Retrieved 10 November 2014.
38. "Nasty Covert Redirect Vulnerability found in OAuth and OpenID" (http://thehackernews.com/2014/05/nasty-covert-redirect-vulnerability.html). The Hacker News. 3 May 2014. Retrieved 10 November 2014.
39. "Math student detects OAuth, OpenID security vulnerability" (http://techxplore.com/news/2014-05-math-student-oauth-openid-vulnerability.html). Tech Xplore. 3 May 2014. Retrieved 10 November 2014.
40. "Covert Redirect" (http://openid.net/2014/05/15/covert-redirect/). OpenID. 15 May 2014. Retrieved 10 November 2014.
41. " 'Covert Redirect' vulnerability impacts OAuth 2.0, OpenID" (http://www.scmagazine.com/covert-redirect-vulnerability-impacts-oauth-20-openid/article/345407/). SC Magazine. 2 May 2014. Retrieved 10 November 2014.
42. "Lessons to be Learned from Covert Redirect" (http://www.the41.com/buzz/press/lessons-be-learned-covert-redirect). 41st Parameter. 5 May 2014. Retrieved 10 November 2014.
43. Fitzpatrick, Brad (2005-05-16). "Distributed Identity: Yadis" (https://web.archive.org/web/20060504054201/http://community.livejournal.com/lj_dev/683939.html). *LiveJournal*. Archived from the original (http://community.livejournal.com/lj_dev/683939.html) on 2006-05-04. Retrieved 2008-03-20.
44. Waters, John K (2007-12-01). "OpenID Updates Identity Spec" (https://web.archive.org/web/20080208155322/http://reddevnews.com/news/devnews/article.aspx?editorialsid=913). *Redmond Developer News*. Archived from the original (http://reddevnews.com/news/devnews/article.aspx?editorialsid=913) on 2008-02-08. Retrieved 2008-03-20.
45. "Glossary" (http://www.livejournal.com/doc/server/appx.glossary.html). LiveJournal Server: Technical Info. Retrieved 13 October 2009.
46. Lehn, David I. (18 May 2005). "18 May 2005" (https://web.archive.org/web/20101221200243/http://advogato.org/person/dlehn/diary/5.html). *Advogato blog for dlehn*. Advogato. Archived from the original (http://www.advogato.org/person/dlehn/diary/5.html) on 21 December 2010. Retrieved 13 October 2009. "They were looking for a name and managed to email me about openid.net right before I was going to offer it to them. So I gave it to them for the new and improved OpenID project."
47. "OpenID: an actually distributed identity system" (https://web.archive.org/web/20050924033518/http://www.danga.com/openid/). 2005-09-24. Archived from the original (http://www.danga.com/openid/) on 2005-09-24. Retrieved 2008-03-20.
48. Fitzpatrick, Brad (2006-05-30). "brad's life – OpenID and SixApart" (https://web.archive.org/web/20070425033329/http://brad.livejournal.com/2226738.html). *LiveJournal*. Archived from the original (http://brad.livejournal.com/2226738.html) on 2007-04-25. Retrieved 2008-03-20.
49. Recordon, David (2005-12-24). "Announcing YADIS...again" (http://lists.danga.com/pipermail/yadis/2005-October/001511.html). *Danga Interactive*. Retrieved 2008-03-20.

50. Reed, Dummond (2005-12-31). "Implementing YADIS with no new software" (http://lists.danga.com/pipermail/yadis/2005-October/001544.html). *Danga Interactive*. Retrieved 2008-03-20.
51. Reed, Drummond (2008-11-30). "XRD Begins" (http://www.equalsdrummond.name/?p=172). *Equals Drummond*. Retrieved 5 January 2009.
52. Hardt, Dick (2005-12-18). "Sxip concerns with YADIS" (http://lists.danga.com/pipermail/yadis/2005-December/001873.html). *Danga Interactive*. Retrieved 2008-03-20.
53. Hardt, Dick (2005-12-10). "SXIP 2.0 Teaser" (https://web.archive.org/web/20070814212337/http://identity20.com/?p=44). *Identity 2.0*. Archived from the original (http://identity20.com/?p=44) on 2007-08-14. Retrieved 2008-03-20.
54. Hoyt, Josh (2006-03-15). "OpenID + Simple Registration Information Exchange" (http://lists.danga.com/pipermail/yadis/2006-March/002304.html). *Danga Interactive*. Retrieved 2008-03-20.
55. Grey, Victor (2006-04-02). "Proposal for an XRI (i-name) profile for OpenID" (http://lists.danga.com/pipermail/yadis/2006-April/002388.html). *Danga Interactive*. Retrieved 2008-03-20.
56. Recordon, David (2006-04-29). "Movin' On..." (https://web.archive.org/web/20061020010916/http://daveman692.livejournal.com/251286.html) *LiveJournal*. Archived from the original (http://daveman692.livejournal.com/251286.html) on 2006-10-20. Retrieved 2008-03-20.
57. Recordon, David (2006-06-16). "Moving OpenID Forward" (http://lists.danga.com/pipermail/yadis/2006-June/002631.html). *Danga Interactive*. Retrieved 2008-05-19.
58. Johannes Ernst and David Recordon. Editor:Phil Becker (2006-12-04). "The case for OpenID" (https://www.zdnet.com/blog/digitalid/the-case-for-openid/78). *ZDNet*. Retrieved 2010-12-12. {{cite news}}: |author= has generic name (help)
59. "Symantec Unveils Security 2.0 Identity Initiative at DEMO 07 Conference" (http://www.symantec.com/about/news/release/article.jsp?prid=20070131_01). *Symantec*. 2007-01-31. Retrieved 2008-03-20.
60. Graves, Michael (2007-02-06). "VeriSign, Microsoft & Partners to Work together on OpenID + Cardspace" (https://web.archive.org/web/20080503001116/http://blogs.verisign.com/infrablog/2007/02/verisign_microsoft_partners_to_1.php). *VeriSign*. Archived from the original (http://blogs.verisign.com/infrablog/2007/02/verisign_microsoft_partners_to_1.php) on 2008-05-03. Retrieved 2008-03-20.
61. Panzer, John (2007-02-16). "AOL and 63 Million OpenIDs" (https://web.archive.org/web/20080511162600/http://dev.aol.com/aol-and-63-million-openids). *AOL Developer Network*. Archived from the original (http://dev.aol.com/aol-and-63-million-openids) on 2008-05-11. Retrieved 2008-03-20.
62. "Sun Microsystems Announces OpenID Program" (http://www.prnewswire.com/cgi-bin/stories.pl?ACCT=104&STORY=/www/story/05-07-2007/0004582105&EDATE=). *PR Newswire*. 2007-05-07. Retrieved 2008-03-20.
63. OpenID Board of Directors (2007-06-01). "OpenID Foundation" (http://openid.net/foundation/). Retrieved 2008-03-20.
64. OpenID Europe Foundation (http://www.openideurope.eu/foundation/)
65. "OpenID 2.0...Final(ly)!" (http://openid.net/2007/12/05/openid-2_0-final-ly/). *OpenID Foundation*. 2007-12-05. Retrieved 2008-03-20.
66. "Yahoo! Announces Support for OpenID; Users Able to Access Multiple Internet Sites with Their Yahoo! ID" (https://web.archive.org/web/20080304014817/http://biz.yahoo.com/bw/080117/20080117005332.html). *Yahoo!*. 2008-01-17. Archived from the original (http://biz.yahoo.com/bw/080117/20080117005332.html) on 2008-03-04. Retrieved 2008-03-20.
67. "Technology Leaders Join OpenID Foundation to Promote Open Identity Management on the Web" (http://www.marketwire.com/mw/release.do?id=818650). *OpenID Foundation*. Marketwire. 2008-02-07. Retrieved 2008-03-20.

68. "SourceForge Implements OpenID Technology" (https://web.archive.org/web/20080513100231/http://www.primenewswire.com/newsroom/news.html?d=142213) (Press release). SourceForge, Inc. May 7, 2008. Archived from the original (http://www.primenewswire.com/newsroom/news.html?d=142213) on May 13, 2008. Retrieved 2008-05-21.
69. "MySpace Announces Support for "OpenID" and Introduces New Data Availability Implementations" (http://www.businesswire.com/news/home/20080722006024/en). *Business Wire*. MySpace. 2008-07-22. p. 2. Retrieved 2008-07-23.
70. "Microsoft and Google announce OpenID support" (http://openid.net/2008/10/30/microsoft-and-google-announce-openid-support/). OpenID Foundation. 2008-10-30.
71. "JanRain Releases Free Version of Industry Leading OpenID Solution" (https://web.archive.org/web/20081218054500/http://www.janrain.com/press/2008/rpxnow) (Press release). JanRain, Inc. November 14, 2008. Archived from the original (http://www.janrain.com/press/2008/rpxnow) on December 18, 2008. Retrieved 2008-11-14.
72. "Facebook Developers | Facebook Developers News" (https://web.archive.org/web/20091223072159/http://developers.facebook.com/news.php?blog=1&story=246). Developers.facebook.com. 2009-05-18. Archived from the original (https://developers.facebook.com/news.php?blog=1&story=246) on 2009-12-23. Retrieved 2009-07-28.
73. "Facebook now accepts Google account logins" (http://www.pocket-lint.com/news/news.phtml/24185/facebook-accepting-google-login-openid.phtml). Pocket-lint.com. 2009-05-19. Retrieved 2009-07-28.
74. "OpenID Requirements – Facebook Developer Wiki" (https://web.archive.org/web/20091223094413/http://wiki.developers.facebook.com/index.php/OpenID_Requirements). Wiki.developers.facebook.com. 2009-06-26. Archived from the original (http://wiki.developers.facebook.com/index.php/OpenID_Requirements) on 2009-12-23. Retrieved 2009-07-28.
75. Kane, Zee M (4 September 2013). "MyOpenID to shut down. Will be turned off on February 1, 2014" (https://thenextweb.com/insider/2013/09/04/myopenid-to-shut-down/). *The Next Web*. Retrieved 5 September 2013.
76. "OpenID Sponsoring Members" (http://openid.net/foundation/sponsoring-members/). Retrieved 17 April 2014.
77. "Symantec Personal Identification Portal banner indicates service will be discontinued on 12 September 2016" (https://web.archive.org/web/20160611151621/https://pip.verisignlabs.com/login.do). Archived from the original (https://pip.verisignlabs.com/login.do) on 11 June 2016. Retrieved 17 May 2016.
78. "Is Symantec failing hard at being Google?" (https://what.thedailywtf.com/topic/19880/is-symantec-failing-hard-at-being-google/12). 7 May 2016. Retrieved 17 May 2016.
79. "Support for OpenID ended on July 25, 2018" (https://meta.stackexchange.com/questions/307647/support-for-openid-ends-on-july-1-2018).
80. "User Authentication with OAuth 2.0" (http://oauth.net/articles/authentication/). *OAuth.net*. Retrieved 19 March 2015.
81. "Why is it a bad idea to use plain oauth2 for authentication?" (https://security.stackexchange.com/questions/133065/why-is-it-a-bad-idea-to-use-plain-oauth2-for-authentication/134280#134280). *Information Security Stack Exchange*. Retrieved 7 July 2018.
82. "OpenID Connect FAQ and Q&As" (http://openid.net/connect/faq/). Retrieved 25 August 2014.

External links

- Official website (https://openid.net/)
- OpenID (https://curlie.org/Computers/Security/Authentication/Single_Sign-On/OpenID) at Curlie

OAUTH WG
Internet-Draft
Intended status: Informational
Expires: January 17, 2013

G. Fletcher
AOL
T. Lodderstedt
Deutsche Telekom AG
Z. Zeltsan
Alcatel-Lucent

July 16, 2012

OAuth Use Cases
draft-ietf-oauth-use-cases-01

Abstract

This document lists the OAuth use cases. The provided list is based on the Internet Drafts of the OAUTH working group and discussions on the group's mailing list.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 17, 2013.

Table of Contents

1. Introduction

This document describes the use cases that have been discussed on the oauth WG mailing list and introduced by the Internet Drafts submitted to the group. The selected use cases illustrate the use of the OAuth flows by the clients of the various profiles and types. The document also includes those cases that are not directly supported by the OAuth 2.0 protocol, but were considered during its development. The document provides a list of the requirements derived from the use cases. The use cases supported by OAuth 2.0 are indicated.

The document's objective is to help with understanding of the OAuth 2.0 protocol design.

Note: The use of the string ".example.com" in the URLs of the example entities does not mean that the entities belong to the same organization.

The following section provides the abbreviated descriptions of the use cases.

2. OAuth use cases

This section lists the use cases that have been discussed by the oauth WG.

2.1. Web server

Description:

Alice accesses an application running on a web server at www.printphotos.example.com and instructs it to print her photographs that are stored on a server www.storephotos.example.com. The application at www.printphotos.example.com receives Alice's authorization for accessing her photographs without learning her authentication credentials with www.storephotos.example.com.

Pre-conditions:

- Alice has registered with www.storephotos.example.com to enable authentication

- The application at www.printphotos.example.com has established authentication credentials with the application at www.storephotos.example.com

Post-conditions:

A successful procedure results in the application www.printphotos.example.com receiving an authorization code from www.storephotos.example.com. The code is bound to the application at www.printphotos.example.com and to the callback URL supplied by the application. The application at www.printphotos.example.com uses the authorization code for obtaining an access token from www.storephotos.example.com. The application at www.storephotos.example.com issues an access token after authenticating the application at www.printphotos.example.com and validating the authorization code that it has submitted. The application at www.printphotos.example.com uses the access token for getting access to Alice's photographs at www.storephotos.example.com.

Note: When an access token expires, the service at www.printphotos.example.com needs to repeat the OAuth procedure for getting Alice's authorization to access her photographs at www.storephotos.example.com. Alternatively, if Alice wants to grant the application a long lasting access to her resources at www.storephotos.example.com, the authorization server associated with www.storephotos.example.com may issue the long-living tokens. Those tokens can be exchanged for short-living access tokens required to access www.storephotos.example.com.

Requirements:

- The server www.printphotos.example.com, which hosts an OAuth client, must be capable of issuing the HTTP redirect requests to Alice's user agent - a browser

- Application at www.storephotos.example.com must be able to authenticate Alice. The authentication method is not in the OAuth scope

- Application at www.storephotos.example.com must obtain Alice's authorization of the access to her photos by www.printphotos.example.com

- Application at www.storephotos.example.com may identify to Alice the scope of access that www.printphotos.example.com has requested while asking for Alice's authorization

- Application at www.storephotos.example.com must be able to authenticate the application at www.printphotos.example.com and validate the authorization code before issuing an access token. The OAuth 2.0 protocol [I-D.draft-ietf-oauth-v2] specifies one authentication method that MAY be used for such authentication - Client Password Authentication.

- Application at www.printphotos.example.com must provide a callback URL to the application at www.storephotos.example.com (note: the URL can be pre-registered with www.storephotos.example.com)

- Application at www.storephotos.example.com is required to maintain a record that associates the authorization code with the application at www.printphotos.example.com and the callback URL provided by the application

- Access tokens are bearer's tokens (they are not associated with a specific application, such as www.printphotos.example.com) and should have a short lifespan

- Application at www.storephotos.example.com must invalidate the authorization code after its first use

- Alice's manual involvement in the OAuth authorization procedure (e.g., entering an URL or a password) should not be required. (Alice's authentication to www.storephotos.example.com is not in the OAuth scope. Her registration with www.storephotos.example.com is required as a pre-condition)

2.2. User-agent

Description:

Alice has installed on her computer a gaming application. She keeps her scores in a database of a social site at www.fun.example.com. In order to upload Alice's scores, the application gets access to the database with her authorization.

Pre-conditions:

- Alice has installed a gaming application implemented in a scripting language (e.g., JavaScript) that runs in her browser and uses OAuth for accessing a social site at www.fun.example.com

- There is no a web site supporting this application and capable of handling the OAuth flow, so the gaming application needs to update the database itself

- The installed application is registered with the social site at www.fun.example.com and has an identifier

- Alice has registered with www.fun.example.com for identification and authentication

- An auxiliary web server at www.help.example.com is reachable by Alice's browser and capable of providing a script that extracts an access token from an URL's fragment

Post-conditions:

A successful procedure results in Alice's browser receiving an access token. The access token is received from www.fun.example.com as a fragment of a redirection URL of an auxiliary web server www.help.example.com. Alice's browser follows the redirection, but retains the fragment. From the auxiliary web server at www.help.example.com Alice's browser downloads a script that extracts access token from the fragment and makes it available to the gaming application. The application uses the access token to gain access to Alice's data at www.fun.example.com.

Requirements:

- Registration of the application running in the Alice's browser with the application running on www.fun.example.com is required for identification

- Alice's authentication with www.fun.example.com is required

- Application running at www.fun.example.com must be able to describe to Alice the request made by the gaming application running on her computer and obtain Alice's authorization for or denial of the requested access

- After obtaining Alice's authorization the application running at www.fun.example.com must respond with an access token and redirect Alice's browser to a web server (e.g., www.help.example.com) that is capable of retrieving an access token from an URL

2.3. In-App-Payment (based on Native Application)

Description:

Alice has installed on her computer a gaming application (e.g., running as native code or as a widget). At some point she wants to play the next level of the game and needs to purchase an access to the advanced version of the game from her service provider at www.sp.example.com. With Alice's authorization the application accesses her account at www.sp.example.com and enables her to make the payment.

Pre-conditions:

- Alice has registered and has an account with her service provider at www.sp.example.com

- The application is registered with the service provider at www.sp.example.com. This enables the server provider to provide Alice with all necessary information about the gaming application (including the information about the purchasing price)

- Alice has a Web user-agent (e.g., a browser or a widget runtime) installed on her computer

Post-conditions:

A successful procedure results in the gaming application invoking the user browser and directing it to the authorization server of the service provider. The HTTP message includes information about the gaming application's request to access Alice's account. The authorization server presents to Alice the authentication and authorization interfaces. The authorization interface shows Alice the information about the application's request including the requested charge to her account. After Alice successfully authenticates and authorizes the request, the authorization server enables Alice to save the transaction details including the authorization code issued for the gaming application. Then the authorization server redirects Alice's browser to a custom scheme URI (registered with the operating system). This redirection request contains a one-time authorization code and invokes a special application that is able to extract the authorization code and present it to the gaming application. The gaming application presents the authorization code to the authorization server and exchanges it for a one-time access token. The gaming application then uses the access token to get access to Alice's account and post the charges at www.sp.example.com.

Requirements:

- An authorization server associated with the server at www.sp.example.com must be able to authenticate Alice over a secure transport

- An authorization server associated with the server at www.sp.example.com must be able to provide Alice with information about the access request that the gaming application has made (including the amount that is to be charged to her account with the service provider, and the purpose for the charge) over a secure transport

- An authorization server associated with the server at www.sp.example.com must be able to obtain Alice's authorization decision on the request over a secure transport

- An authorization server associated with the server at www.sp.example.com must be able to generate on demand a one-time authorization code and a one-time access token according to the scope authorized by Alice

- An authorization server associated with the server at www.sp.example.com must be able to call back to the gaming application with the authorization result over a secure transport

- An authorization server associated with the server at www.sp.example.com must enable the gaming application to exchange an authorization code for an access token over a secure transport

- * An authorization server associated with the server at www.sp.example.com must verify the authorization code and invalidate it after its first use

- * An authorization server associated with the server at www.sp.example.com must enable Alice to save the details of the requested transaction, including the authorization code

- * An authorization server associated with the server at www.sp.example.com must keep a record linking the requested transaction with the authorization code and the respective access token

- * An authorization server associated with the server at www.sp.example.com must enable the resource server www.sp.example.com to obtain the transaction information that is linked to the issued access token

- * Resource server at www.sp.example.com must verify access token and invalidate it after its first use

- * A resource server at www.sp.example.com must enable the gaming application to post charges to Alice's account according to the access token presented over a secure transport

- The gaming application must provide a custom scheme URI to the authorization server associated with www.sp.example.com (note: it can be preregistered with the authorization server)

- Alice's manual involvement in the OAuth authorization procedure (e.g., entering an URL or a password) should not be required.

(Alice's authentication to www.sp.example.com is not in the OAuth scope)

* The requirements denoted by '*' are not common for the Native Application use cases, but are specific to the In-App-Payment use case

2.4. Native Application

Description:

Alice wants to upload (or download) her photographs to (or from) storephotos.example.com using her smartphone. She downloads and installs a photo app on her smartphone. In order to enable the app to access her photographs, Alice needs to authorize the app to access the web site on her behalf. The authorization shall be valid for a prolonged duration (e.g. several months), so that Alice does not need to authenticate and authorize access on every execution of the app. It shall be possible to withdraw the app's authorization both on the smartphone as well as on the site storephotos.example.com.

Pre-conditions:

- Alice has installed a (native) photo app application on her smartphone
- The installed application is registered with the social site at storephotos.example.com and has an identifier
- Alice holds an account with storephotos.example.com
- Authentication and authorization shall be performed in an interactive, browser-based process. The smartphone's browser is used for authenticating Alice and for enabling her to authorize the request by the Mobile App

Post-conditions:

A successful procedure results in Alice's app receiving an access and a refresh token. The app may obtain the tokens by utilizing either the web server or the user agent flow. The application uses the access token to gain access to Alice's data at storephotos.example.com. The refresh token is persistently stored on the device for use in sub-sequent app executions. If a refresh token exists on app startup, the app directly uses the refresh token to obtain a new access token.

Requirements:

- Alice's authentication with storephotos.example.com is required

- Registration of the application running on Alice's smartphone is required for identification and registration and may be carried out on a per installation base

- The application at storephotos.example.com provides a capability to view and delete the apps' authorizations. This implies that the different installations of the same app on the different devices can be distinguished (e.g., by a device name or a telephone number)

- The app must provide Alice an option to logout. The logout must result in the revocation of the refresh token on the authorization server

2.5. Device

Description:

Alice has a device, such as a game console, that does not support an easy data-entry method. She also has an access to a computer with a browser. The application running on the Alice's device gets authorized access to a protected resource (e.g., photographs) stored on a server at www.storephotos.example.com

Pre-conditions:

- Alice uses an Oauth-enabled game console, which does not have an easy data-entry method, for accessing her photographs at www.storephotos.example.com. The device starts the OAuth procedure by requesting a token

- Alice is able to connect to www.storephotos.example.com using a computer that provides an easy data-entry method, which is equipped with a browser. This computer is used to authorize access by the application running on the game console to Alice's photographs

- Application running on Alice's game console has registered with www.storephotos.example.com (has been issued an identifier)

- Alice has registered with the application running at www.storephotos.example.com for identification and authentication

Post-conditions: Description:

A successful procedure results in the application running on Alice's

game console receiving an access token that enables access to the photographs on www.storephotos.example.com.

Requirements:

- Registration of the application running on the game console with the application running on www.storephotos.example.com is required for identification

- Application running on the game console must be able to poll periodically the application running at www.storephotos.example.com while waiting for Alice's authorization of the requested access to her photographs. The repeating requests include the application's identifier and the verification code that has been issued by www.storephotos.example.com

- Alice is required to use her browser for interacting with the web application running on www.storephotos.example.com. To that end she has to manually direct her browser to the verification URL that is displayed on her game console

- Alice's authentication with www.storephotos.example.com is required

- After authentication with www.storephotos.example.com Alice, if she wishes to approve the request, which is described in her browser's window, must enter the user code. (The user code is also displayed on her game console along with the verification URL)

2.6. Client password (shared secret) credentials

Description:

The company GoodPay prepares the employee payrolls for the company GoodWork. In order to do that the application at www.GoodPay.example.com gets authenticated access to the employees' attendance data stored at www.GoodWork.example.com.

Pre-conditions:

- The application at www.GoodPay.example.com has established through a registration an identifier and a shared secret with the application running at www.GoodWork.example.com

- The scope of the access by the application at www.GoodPay.example.com to the data stored at

www.GoodWork.example.com has been defined

Post-conditions:

A successful procedure results in the application at www.GoodPay.example.com receiving an access token after authenticating to the application running at www.GoodWork.example.com.

Requirements:

- Authentication of the application at www.GoodPay.example.com to the application at www.GoodWork.example.com is required
- The authentication method must be based on the identifier and shared secret, which the application running at www.GoodPay.example.com submits to the application at www.GoodWork.example.com in the initial HTTP request
- Because in this use case GoodPay gets access to GoodWork's sensitive data, GoodWork shall have a pre-established trust with GoodPay on the security policy and the authorization method's implementation

2.7. Assertion

Description:

Company GoodPay prepares the employee payrolls for the company GoodWork. In order to do that the application at www.GoodPay.example.com gets authenticated access to the employees' attendance data stored at www.GoodWork.example.com.
This use case describes an alternative solution to the one described by the use case Client password credentials.

Pre-conditions:

- The application at www.GoodPay.example.com has obtained an authentication assertion from a party that is trusted by the application at www.GoodWork.example.com
- The scope of the access by the application at www.GoodPay.example.com to the data stored at www.GoodWork.example.com has been defined
- The application at www.GoodPay.example.com has established trust relationship with the asserting party and is capable of validating its assertions

Post-conditions:

A successful procedure results in the application at www.GoodPay.example.com receiving an access token after authenticating to the application running at www.GoodWork.example.com by presenting an assertion (e.g., SAML assertion).

Requirements:

- Authentication of the application at www.GoodPay.example.com to the application at www.GoodWork.example.com is required
- The application running at www.GoodWork.example.com must be capable of validating assertion presented by the application running at www.GoodPay.example.com
- Because in this use case GoodPay gets access to GoodWork's sensitive data, GoodWork shall establish trust with GoodPay on the security policy and the authorization method's implementation

2.8. Content manager

Description:

Alice and Bob are having a chat conversation using a content manager application running on a web server at www.contentmanager.example.com. Alice notifies Bob that she wants to share some photographs at www.storephotos.example.com and instructs the application at www.contentmanager.example.com to enable Bob's access to the photographs. The application at www.contentmanager.example.com, after Alice's authorization, obtains an access token for Bob, who uses it to access Alice's photographs at www.storephotos.example.com.

Pre-conditions:

Alice, Bob the content manager application at www.contentmanager.example.com, and the application at www.storephotos.example.com have registered with the same authorization server for authentication

Post-conditions:

A successful procedure results in the application at www.contentmanager.example.com receiving an access token that allows access to Alice's photographs at www.storephotos.example.com. The access token is issued by the authorization server after Alice has authorized the content manager at www.contentmanager.example.com to

get an access token on Bob's behalf. The access token is passed to Bob by the content manager. Bob uses the access token to view Alice's photographs at www.storephotos.example.com.

Requirements:

- The server at www.contentmanager.example.com, must be capable of issuing the HTTP redirect requests to Alice's and Bob's user agents - the browsers
- The authorization server must be able to authenticate Alice, Bob, and the application at www.contentmanager.example.com
- The authorization server is required to obtain Alice's authorization for issuing an access token to www.contentmanager.example.com on Bob's behalf
- Authorization server must be able to identify to Alice the scope of access that www.contentmanager.example.com has requested on Bob's behalf while asking for Alice's authorization

2.9. Access token exchange

Description:

Alice uses an application running on www.printphotos.example.com for printing her photographs that are stored on a server at www.storephotos.example.com. The application running on www.storephotos.example.com, while serving the request of the application at www.printphotos.example.com, discovers that some of the requested photographs have been moved to www.storephotos1.example.com. The application at www.storephotos.example.com retrieves the missing photographs from www.storephotos1.example.com and provides access to all requested photographs to the application at www.printphotos.example.com. The application at www.printphotos.example.com carries out Alice's request.

Pre-conditions:

- The application running on www.printphotos.example.com is capable of interacting with Alice's browser
- Alice has registered with and can be authenticated by authorization server
- The applications at www.storephotos.example.com has registered with authorization server

- The applications at www.storephotos1.example.com has registered with authorization server

- The application at www.printphotos.example.com has registered with authorization server

Post-conditions:

A successful procedure results in the application at www.printphotos.example.com receiving an access token that allows access to Alice's photographs. This access token is used for the following purposes:

- By the application running at www.printphotos.example.com to get access to the photographs at www.storephotos.example.com

- By the application running at www.storephotos.example.com to obtain from authorization server another access token that allows it to retrieve the additional photographs stored at www.storephotos1.example.com

As the result, there are two access token issued for two different applications. The tokens may have different properties (e.g., scope, permissions, and expiration dates).

Requirements:

- The applications at www.printphotos.example.com and www.storephotos.example.com require different access tokens

- The application at www.printphotos.example.com is required to provide its callback URL to the application at www.storephotos.example.com

- Authentication of the application at www.printphotos.example.com to the authorization server is required

- Alice's authentication by the authorization server is required

- The authorization server must be able to describe to Alice the request of the application at www.printphotos.example.com and obtain her authorization (or rejection)

- If Alice has authorized the request, the authorization server must be able to issue an access token that enables the application at www.printphotos.example.com to get access to Alice's photographs at www.storephotos.example.com

- The authorization server must be able, based on the access token presented by the application at www.printphotos.example.com, to generate another access token that allows the application at www.storephotos.example.com to get access to the photographs at www.storephotos1.example.com. In this context the authorization server must validate the authorization of the application at www.storephotos.example.com to obtain the token.

- The application at www.storephotos.example.com must be able to validate an access token presented by the application running at www.printphotos.example.com

- The application at www.storephotos1.example.com must be able to validate the access token presented by the application running at www.storephotos.example.com

2.10. Multiple access tokens

Description:

Alice uses a communicator application running on a web server at www.communicator.example.com to access her email service at www.email.example.com and her voice over IP service at www.voip.example.com. Email addresses and telephone numbers are obtained from Alice's address book at www.contacts.example.com. Those web sites all rely on the same authorization server, so the application at www.communicator.example.com can receive a single authorization from Alice for getting access to these three services on her behalf at once.

Note: This use case is especially useful for native applications since a web browser needs to be launched only once.

Pre-conditions:

- The same authorization server serves Alice and all involved servers

- Alice has registered with the authorization server for authentication and for authorization of the requests of the communicator application running at www.communicator.example.com

- The email application at www.email.example.com has registered with the authorization server for authentication

- The VoIP application at www.voip.example.com has registered with the authorization server for authentication

- The address book at www.contacts.example.com has registered with the authorization server for authentication

Post-conditions:

A successful procedure results in the application at www.communicator.example.com receiving three different access tokens: one for accessing the email service at www.email.example.com, one for accessing the contacts at www.contacts.example.com, and one for accessing the VoIP service at www.voip.example.com.

Requirements:

- The application running at www.communicator.example.com must be authenticated by the authorization server
- Alice must be authenticated by the authorization server
- The application running at www.communicator.example.com must be able to get a single Alice's authorization for access to the multiple services (e.g., email and VoIP)
- The application running at www.communicator.example.com must be able to recognize that all three applications rely on the same authorization server
- A callback URL of the application running at www.communicator.example.com must be known to the authorization server
- The authorization server must be able to issue the separate service-specific tokens (with different, scope, permissions, and expiration dates) for access to the requested services (such as email and VoIP)

2.11. Gateway for browser-based VoIP applets

Description:

Alice accesses a social site on a web server at www.social.example.com. Her browser loads a VoIP applet that enables her to make a VoIP call using her SIP server at www.sipservice.example.com. The application at www.social.example.com gets Alice's authorization to use her account with www.sipservice.example.com without learning her authentication credentials with www.sipservice.example.com.

Pre-conditions:

- Alice has registered with www.sipservice.example.com for authentication

- The application at www.social.example.com has established authentication credentials with the application at www.sipservice.example.com

Post-conditions:

A successful procedure results in the application at www.social.example.com receiving access token from www.sipservice.example.com with Alice's authorization.

Requirements:

- The server at www.social.example.com must be able to redirect Alice's browser to www.sipservice.example.com

- The application running at www.sipservice.example.com must be capable of authenticating Alice and obtaining her authorization of a request from www.social.example.com

- The server at www.sipservice.example.com must be able to redirect Alice's browser back to www.social.example.com

- The application at www.social.example.com must be able to translate the messages of the Alice's VoIP applet into SIP and RTP messages

- The application at www.social.example.com must be able to add the access token to the SIP requests that it sends to www.sipservice.example.com

- Application at www.sipservice.example.com must be able to authenticate the application at www.social.example.com and validate the access token

- Alice's manual involvement in the OAuth authorization procedure (e.g., entering an URL or a password) should not be required. (Alice's authentication to www.sipservice.example.com is not in the OAuth scope)

2.12. Signed Messages

Description:

Alice manages all her personal health records in her personal health data store at a server at www.myhealth.example.com, which manages

authorization of access to Alice's participating health systems. Alice's Primary Care Physician (PCP), which has a Web site at www.pcp.example.com, recommends her to see a sleep specialist (www.sleepwell.example.com). Alice arrives at the sleep specialist's office and authorizes it to access her basic health data at her PCP's web site. The application at www.pcp.example.com verifies that Alice has authorized www.sleepwell.example.com to access her health data as well as enforces that www.sleepwell.example.com is the only application that can retrieve that data with that specific authorization.

Pre-conditions:

- Alice has a personal health data store that allows for discovery of her participating health systems (e.g. psychiatrist, sleep specialist, PCP, orthodontist, ophthalmologist, etc)

- The application at www.myhealth.example.com manages authorization of access to Alice's participating health systems

- The application at www.myhealth.example.com can issue authorization tokens understood by Alice's participating health systems

- The application at www.pcp.example.com stores Alice's basic health and prescription records

- The application at www.sleepwell.com stores results of Alice's sleep tests

Post-conditions:

- A successful procedure results in just the information that Alice authorized being transferred from the Primary Care Physician (www.pcp.example.com) to the sleep specialist (www.sleepwell.example.com)

- The transfer of health data only occurs if the application at www.pcp.example.com can verify that www.sleepwell.example.com is the party requesting access and that the authorization token presented by www.sleepwell.example.com is issued by the application at www.myhealth.example.com with a restricted audience of www.sleepwell.example.com

Requirements:

- The application at www.sleepwell.example.com interacting with www.myhealth.example.com must be able to discover the location of

the PCP system (e.g., XRD discovery)

- The application at www.sleepwell.example.com must be capable of requesting Alice's authorization of access to the application at www.pcp.example.com for the purpose of retrieving basic health data (e.g. date-of-birth, weight, height, etc). The mechanism Alice uses to authorize this access is out of scope for this use case

- The application at www.myhealth.example.com must be capable of issuing a token bound to www.sleepwell.example.com for access to the application at www.pcp.example.com. Note that a signed token (JWT) can be used to prove who issued the token

- The application at www.sleepwell.example.com must be capable of issuing a request (which includes the token issued by www.myhealth.example.com) to the application at www.pcp.example.com

- The application at www.sleepwell.example.com must sign the request before sending it to www.pcp.example.com

- The application at www.pcp.example.com must be capable of receiving the request and verifying the signature

- The application at www.pcp.example.com must be capable of parsing the message and finding the authorization token

- The application at www.pcp.example.com must be capable of verifying the signature of the authorization token

- The application at www.pcp.example.com must be capable of parsing the authorization token and verifying that this token was issued to the application at www.sleepwell.com

- The application at www.pcp.example.com must be capable of retrieving the requested data and returning it to the application at www.sleepwell.example.com

2.13. Signature with asymmetric secret

Description:

Alice accesses an application running on a web server at www.printphotos.example.com and instructs it to print her photographs that are stored on a server www.storephotos.example.com. The application at www.printphotos.example.com, which does not have a shared secret with www.storephotos.example.com, receives Alice's

authorization for accessing her photographs without learning her authentication credentials with www.storephotos.example.com.

Pre-conditions:

- Alice has registered with www.storephotos.example.com to enable authentication
- The application at www.printphotos.example.com has a private and a matching public keys

Post-conditions:

A successful procedure results in the application at www.printphotos.example.com receiving an access token from www.storephotos.example.com for accessing the Alice's photographs.

Requirements:

- The application at www.printphotos.example.com must be capable of issuing the HTTP redirect requests to Alice's user agent - a browser
- The application at www.storephotos.example.com must be able to authenticate Alice
- The application running at www.storephotos.example.com must be able to obtain the public key of the application at www.printphotos.example.com
- The application running at www.printphotos.example.com is required to sign using its private key the requests to the application at www.storephotos.example.com
- The application at www.storephotos.example.com must obtain Alice's authorization of the access to her photos by www.printphotos.example.com
- The application at www.storephotos.example.com is required to identify to Alice the scope of access that www.printphotos.example.com has requested while asking for Alice's authorization
- The application at www.storephotos.example.com must be able to authenticate the application at www.printphotos.example.com by validating a signature of its request using the public key of www.printphotos.example.com

- The application at www.printphotos.example.com must provide a callback URL to the application at www.storephotos.example.com (note: the URL can be pre-registered with www.storephotos.example.com)

- The application at www.storephotos.example.com must be capable of issuing the HTTP redirect requests to Alice's browser

- Alice's manual involvement in the OAuth authorization procedure (e.g., entering an URL or a password) should not be required. (Alice's authentication to www.storephotos.example.com is not in the OAuth scope)

3. Authors of the use cases

The major contributors of the use cases are as follows:

W. Beck, Deutsche Telekom AG
G. Brail, Sonoa Systems
B. de hOra
B. Eaton, Google
S. Farrell, NewBay Software
G. Fletcher, AOL
Y. Goland, Microsoft
B. Goldman, Facebook
E. Hammer-Lahav, Yahoo!
D. Hardt
R. Krikorian, Twitter
T. Lodderstedt, Deutsche Telekom
E. Maler, PayPal
D. Recordon, Facebook
L. Shepard, Facebook
A. Tom, Yahoo!
B. Vrancken, Alcatel-Lucent
Z. Zeltsan, Alcatel-Lucent

4. Security considerations

TBD

5. IANA considerations

This Internet Draft includes no request to IANA.

6. Acknowledgements

The authors thank Igor Faynberg and Hui-Lan Lu for their invaluable help with preparing this document. Special thanks are to the draft reviewers Thomas Hardjono and Melinda Shore, whose suggestions have helped to improve the draft.

7. Normative References

[I-D.draft-ietf-oauth-v2]
Hammer-Lahav, E., Recordon, D., and D. Hardt, "The OAuth 2.0 Authorization Protocol".

Authors' Addresses

George Fletcher
AOL

Email: gffletch@aol.com

Torsten Lodderstedt
Deutsche Telekom AG

Email: torsten@lodderstedt.net

Zachary Zeltsan
Alcatel-Lucent
600 Mountain Avenue
Murray Hill, New Jersey
USA

Phone: +1 908 582 2359
Email: Zachary.Zeltsan@alcatel-lucent.com

FEATURE

INDIA

What is OAuth? How the open authorization framework works

OAuth allows websites and services to share assets among users. It is widely accepted, but be aware of its vulnerabilities.

By Roger A. Grimes and Josh Fruhlinger

CSO |

20 SEPTEMBER 2019 15:30 IST

Since the beginning of distributed personal computer networks, one of the toughest computer security nuts to crack has been to provide a seamless, single sign-on (SSO) access experience among multiple computers, each of which require unrelated logon accounts to access their services and content. Although still not fully realized across the entire internet, myriad, completely unrelated websites can now be accessed using a single physical sign-on. You can use your password, phone, digital certificate, biometric identity, two-factor authentication (2FA) or multi-factor authentication (MFA) SSO solution to log onto one place, and not have to put in another access credential all day to access a bunch of others. We have OAuth to thank for much of it.

OAuth definition

OAuth is an open-standard authorization protocol or framework that describes how unrelated servers and services can safely allow authenticated access to their assets without actually sharing the initial, related, single logon credential. In authentication parlance, this is known as secure, third-party, user-agent, delegated authorization.

[Read reviews of the best tools for single sign-on. | Sign up for CSO newsletters.]

OAuth history

Created and strongly supported from the start by Twitter, Google and other companies, OAuth was released as an open standard in 2010 as RFC 5849, and quickly became widely adopted. Over the next two years, it underwent substantial revision, and version 2.0 of OAuth, was released in 2012 as RFC 6749. Even though version 2.0 was widely

criticized for multiple reasons covered below, it gained even more popularity. Today, you can add Amazon, Facebook, Instagram, LinkedIn, Microsoft, Netflix, Paypal and a list of other internet who's-whos as adopters.

OAuth examples

The simplest example of OAuth is when you go to log onto a website and it offers one or more opportunities to log on using another website's/service's logon. You then click on the button linked to the other website, the other website authenticates you, and the website you were originally connecting to logs you on itself afterward using permission gained from the second website.

Another common example OAuth scenario could be a user sending cloud-stored files to another user via email, when the cloud storage and email systems are otherwise unrelated other than supporting the OAuth framework (e.g., Google Gmail and Microsoft OneDrive). When the end-user attaches the files to their email and browses to select the files to attach, OAuth could be used behind the scenes to allow the email system to seamlessly authenticate and browse to the protected files without requiring a second logon to the file storage system. Another example, one given in the OAuth 2.0 RFC, is an end-user using a third-party printing service to print picture files stored on an unrelated web server.

In all cases, two or more services are being used for one transaction by the end-user, and every end-user would appreciate not being asked to log in a second time for what they feel is a single transaction. For OAuth to work, the end-user's client software (e.g., a browser), the services involved and authentication provider must support the right version of OAuth (1.0 versus 2.0).

OAuth explained

When trying to understand OAuth, it can be helpful to remember that OAuth scenarios almost always represent two unrelated sites or services trying to accomplish something on behalf of users or their software. All three have to work together involving multiple approvals for the completed transaction to get authorized.

It is also helpful to remember that OAuth is about authorization in particular and not directly about authentication. Authentication is the process of a user/subject proving its ownership of a presented identity, by providing a password or some other uniquely owned or presented factor. Authorization is the process of letting a subject access resources after a successful authentication, oftentimes somewhere else. Many people think that OAuth stands for open authentication, but it's more helpful to understand it by thinking about it as open AUTHorization.

An early implementer describes OAuth as similar to a car's valet key, which can be used to allow a valet to temporarily drive and park a car, but it doesn't allow the holder full, unlimited access like a regular key. Instead the car can only be driven a few miles, can't access the trunk or locked glove box, and can have many other limitations. OAuth essentially allows the user, via an authentication provider that they have previously successfully authenticated with, to give another website/service a limited access authentication token for authorization to additional resources.

Additionally, OAuth 2.0 is a framework, not a protocol (like version 1.0). It would be like all the car manufacturers agreeing on how valets would automatically request, receive and use valet keys, and how those valet keys would generally look. What the valet keys could do as compared to the full function keys would be up to each car manufacturer. Just like in real life, valets and car owners don't need to care about how it all works. They just want it all to work seamlessly as possible when they hand off the key.

How OAuth works

Let's assume a user has already signed into one website or service (OAuth only works using HTTPS). The user then initiates a feature/transaction that needs to access another unrelated site or service. The following happens (greatly simplified):

1. The first website connects to the second website on behalf of the user, using OAuth, providing the user's verified identity.
2. The second site generates a one-time token and a one-time secret unique to the transaction and parties involved.
3. The first site gives this token and secret to the initiating user's client software.

4. The client's software presents the request token and secret to their authorization provider (which may or may not be the second site).
5. If not already authenticated to the authorization provider, the client may be asked to authenticate. After authentication, the client is asked to approve the authorization transaction to the second website.
6. The user approves (or their software silently approves) a particular transaction type at the first website.
7. The user is given an approved access token (notice it's no longer a request token).
8. The user gives the approved access token to the first website.
9. The first website gives the access token to the second website as proof of authentication on behalf of the user.
10. The second website lets the first website access their site on behalf of the user.
11. The user sees a successfully completed transaction occurring.
12. OAuth is not the first authentication/authorization system to work this way on behalf of the end-user. In fact, many authentication systems, notably Kerberos, work similarly. What is special about OAuth is its ability to work across the web and its wide adoption. It succeeded with adoption rates where previous attempts failed (for various reasons).

Although not as simple as it could be, web coders seem to readily understand the involved transactions. Making a website OAuth-compatible can be done in a few hours to a day (much faster if you've done it before). For a little bit of extra effort, authenticated website access can be extended to literally hundreds of millions of additional users. There's no need for a website to contain its own authentication system with the ability to scale to gigantic proportions. You can find an example of an individual HTTP transaction packet here.

OAuth vs. OpenID

There are a couple of other security technologies that you might hear about in the same context as OAuth, and one of them is *OpenID.* At a base level, the distinction between the two is simple to grasp. Remember when we said up above that the *auth* in OAuth stood for authorization, not authentication? Well, OpenID *is* about

authentication: as a commenter on StackOverflow pithily put it: "OpenID is for humans logging into machines, OAuth is for machines logging into machines on behalf of humans."

OpenID began life in 2005 as a means for logging into the then-popular LiveJournal blogging site but quickly spread to other sites. The idea, in the early days of Web 2.0, was that rather than having multiple logins for multiple websites, OpenID would serve as a single sign-in, vouching for the identities of users. But in practice OpenID was difficult to implement on the developer side, and never really became that appealing to users, especially as there was competition in that space. By 2011, OpenID had become an also-ran, and, *Wired* declared that "The main reason no one uses OpenID is because Facebook Connect does the same thing and does it better. Everyone knows what Facebook is and it's much easier to understand that Facebook is handling your identity than some vague, unrecognized thing called OpenID." (Facebook Connect turned out to not be a world-beater either, but at least people knew what Facebook was.)

That's not quite the end of the story, though. In 2014, OpenID Connect was released, which reinvented OpenID as an authentication layer for OAuth. In this space, OpenID has found a niche, and the two technologies now complement each other in many implementations.

OAuth vs. SAML

The Security Assertion Markup Language, or *SAML,* is another technology you'll hear talked about in the same breath as OAuth. Strictly speaking, the name SAML refers to an XML variant language, but the term can also cover various protocol messages and profiles that make up part of the open SAML standard. SAML describes a framework that allows one computer to perform *both* authentication *and* authorization on behalf of one or more other computers, unlike OAuth, which requires an additional layer like OpenID Connect to perform authentication. SAML can provide single sign-on functionality on its own.

SAML is older than OAuth, and indeed one of the driving factors behind OAuth's creation was that XML protocols like SAML began falling out of vogue; OAuth uses the lighter weight JSON for encoding data, and thus has better support for mobile devices.

In practice, SAML is more often used for enterprise applications — Salesforce uses it for single sign-on, for instance — whereas OAuth is more often in use on the open internet.

OAuth2

There are no perfect universal internet-wide authentication standards. OAuth is particularly maligned because of the drastic changes between versions 1.0 and 2.0. In many ways, OAuth2 is *less* secure, more complex and less prescriptive than version 1.0. Version 2.0 creators focused on making OAuth more interoperable and flexible between sites and devices. They also introduced the concept of token expiration, which did not exist in version 1.0. Regardless of the intent, many of the original founders and supporters threw up their hands and did not support version 2.0.

The changes are so significant that version 2.0 is not compatible with version 1.0, and even different implementations of version 2.0 may not work seamlessly with each other. However, nothing prevents a website from supporting both 1.0 and 2.0, although the 2.0 creators released it with the intent of all websites completely replacing version 1.0.

One of the biggest criticisms of OAuth 2.0 is that the standard intentionally does not directly define or support encryption, signature, client verification or channel binding (tying a particular session or transaction to a particular client and server). Instead, OAuth expects implementers to use an outside protection protocol like Transport Layer Security (TLS), to provide those features.

Is OAuth safe?

TLS can provide all those protections, but it's up to the implementers, on all sides, to require it to be used. Coders and users should look to ensure that OAuth is running inside of TLS protection. Developers can implement code to enforce TLS use and users should be aware that TLS is being used whenever they are begin asked to input authentication credentials (just like they should anytime they are entering in credentials).

Because of the lack of inherent security binding, it's possible for a rogue website to phish a user's legitimate credentials during the part of the process where the user is being required to authenticate themselves to the authorization provider. For example, a

user is using the first service and chooses a feature that forces an OAuth transaction to a second service. It's possible for the first website to fake the second website, where user authentication is often taking place. The rogue website can then collect the user's authentication credentials and react as if the OAuth transaction had successfully taken place.

This is not just a theoretical threat. In the second quarter of 2017, a million Google accounts were successfully phished. The defense is for users to ensure they are entering their credentials in the legitimate second website's domain if prompted for credentials, and to avoid nebulous first websites. There is no perfectly safe, universally accepted, SSO that works on all websites, but with OAuth, we're getting closer.

More on single sign on and identity management:

- What is identity management? Its definition, uses and solutions
- The best identity management advice right now
- What is SAML, what is it used for and how does it work?

Next read this

- *The 10 most powerful cybersecurity companies*
- *7 hot cybersecurity trends (and 2 going cold)*
- *The Apache Log4j vulnerabilities: A timeline*
- *Using the NIST Cybersecurity Framework to address organizational risk*
- *11 penetration testing tools the pros use*

22 cybersecurity myths organizations need to stop believing in 2022

DEFINITION

DMZ in networking

Ben Lutkevich, Technical Writer

What is a DMZ in networking?

In computer networks, a DMZ, or demilitarized zone, is a physical or logical subnet that separates a local area network (LAN) from other untrusted networks -- usually, the public internet. DMZs are also known as *perimeter networks* or *screened subnetworks*.

Any service provided to users on the public internet should be placed in the DMZ network. External-facing servers, resources and services are usually located there. Some of the most common of these services include web, email, domain name system, File Transfer Protocol and proxy servers.

Servers and resources in the DMZ are accessible from the internet, but the rest of the internal LAN remains unreachable. This approach provides an additional layer of security to the LAN as it restricts a hacker's ability to directly access internal servers and data from the internet.

Hackers and cybercriminals can reach the systems running services on DMZ servers. Those servers must be hardened to withstand constant attack. The term *DMZ* comes from the geographic buffer zone that was set up between North Korea and South Korea at the end of the Korean War.

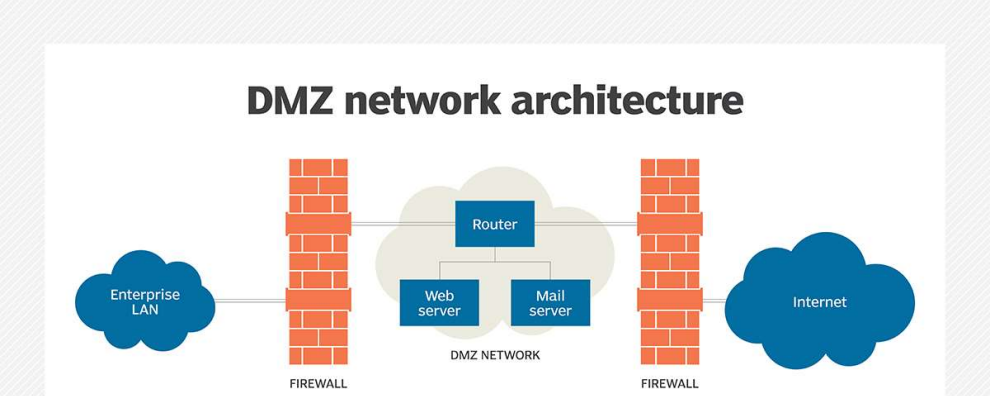
Why are DMZs important?

DMZs provide a level of network segmentation that helps protect internal corporate networks. These subnetworks restrict remote access to internal servers and resources, making it difficult for attackers to access the internal network. This strategy is useful for both individual use and large organizations.

Businesses place applications and servers that are exposed to the internet in a DMZ, separating them from the internal network. The DMZ isolates these resources so, if they are compromised, the attack is unlikely to cause exposure, damage or loss.

How does a DMZ work?

DMZs function as a buffer zone between the public internet and the private network. The DMZ subnet is deployed between two firewalls. All inbound network packets are then screened using a firewall or other security appliance before they arrive at the servers hosted in the DMZ.



A network DMZ sits between two firewalls, creating a semisafe buffer zone between the internet and the enterprise LAN.

If better-prepared threat actors pass through the first firewall, they must then gain unauthorized access to the services in the DMZ before they can do any damage. Those systems are likely to be hardened against such attacks.

Finally, assuming well-resourced threat actors take over a system hosted in the DMZ, they must still break through the internal firewall before they can reach sensitive enterprise resources. Determined attackers can breach even the most secure DMZ architecture. However, a DMZ under attack will set off alarms, giving security professionals enough warning to avert a full breach of their organization.

What are the benefits of using a DMZ?

The primary benefit of a DMZ is that it offers users from the public internet access to certain secure services, while maintaining a buffer between those users and the private internal network. There are several security benefits from this buffer, including the following:

- **Access control.** A DMZ network provides access control to services outside an organization's network perimeters that are accessed from the internet. It simultaneously introduces a level of network segmentation that increases the number of obstacles a user must bypass before gaining access to an organization's private network. In some cases, a DMZ includes a proxy server, which centralizes the flow of internal -- usually, employee -- internet traffic and makes recording and monitoring that traffic simpler.
- **Network reconnaissance prevention.** A DMZ also prevents an attacker from being able to scope out potential targets within the network. Even if a system within the DMZ is compromised, the internal firewall still protects the private network, separating it from the DMZ. This setup makes external active reconnaissance more difficult. Although the servers in the DMZ are publicly exposed, they are backed by another layer of protection. The public face of the DMZ keeps attackers from seeing the contents of the internal private network. If attackers do manage to compromise the servers within the DMZ, they are still isolated from the private network by the DMZ's internal barrier.
- **Protection against Internet Protocol (IP) spoofing.** In some cases, attackers attempt to bypass access control restrictions by spoofing an authorized IP address to impersonate another device on the network. A DMZ can stall potential IP spoofers, while another service on the network verifies the IP address's legitimacy by testing whether it is reachable.

What DMZs are used for

DMZ networks have been an important part of enterprise network security for almost as long as firewalls have been in use. They are deployed for similar reasons: to protect sensitive organizational systems and resources. DMZ networks are often used for the following:

- **isolate** and keep potential target systems separate from internal networks;
- **reduce and control access** to those systems by external users; and
- **host corporate resources** to make some of them available to authorized external users.

More recently, enterprises have opted to use virtual machines or containers to isolate parts of the network or specific applications from the rest of the corporate environment. Cloud technologies have largely removed the need for many organizations to have in-house web servers. Many of the external facing infrastructure once located in the enterprise DMZ has migrated to the cloud, such as software-as-a service apps.

Architecture and design of DMZ networks

There are various ways to design a network with a DMZ. The two basic methods are to use either one or two firewalls, though most modern DMZs are designed with two firewalls. This approach can be expanded to create more complex architectures.

A single firewall with at least three network interfaces can be used to create a network architecture containing a DMZ. The external network is formed by connecting the public internet -- via an internet service provider connection -- to the firewall on the first network interface. The internal network is formed from the second network interface, and the DMZ network itself is connected to the third network interface.

Different sets of firewall rules for monitoring traffic between the internet and the DMZ, the LAN and the DMZ, and the LAN and the internet tightly control which ports and types of traffic are allowed into the DMZ from the internet, limit connectivity to specific hosts in the internal network and prevent unrequested connections either to the internet or the internal LAN from the DMZ.

The more secure approach to creating a DMZ network is a dual-firewall configuration, in which two firewalls are deployed with the DMZ network positioned between them. The first firewall -- also called the *perimeter firewall* -- is configured to allow only external traffic destined for the DMZ. The second, or internal, firewall only allows traffic from the DMZ to the internal network.

The dual-firewall approach is considered more secure because two devices must be compromised before an attacker can access the internal LAN. Security controls can be tuned specifically for each network segment. For example, a network intrusion detection and intrusion prevention system located in a DMZ could be configured to block all traffic except Hypertext Transfer Protocol Secure requests to Transmission Control Protocol port 443.

Examples of DMZs

Some of the various ways DMZs are used include the following:

- **Cloud services.** Some cloud services, such as Microsoft Azure, use a hybrid security approach in which a DMZ is implemented between an organization's on-premises network and the virtual network. This method is typically used in situations where the organization's applications run partly on premises and partly on the virtual network. It's also used where outgoing traffic must be audited or where granular traffic control is required in between the virtual network and the on-premises data center.
- **Home networks.** A DMZ can also be useful in a home network in which computers and other devices are connected to the internet using a broadband router and configured into a LAN. Some home routers include a DMZ host feature. This can be contrasted with the DMZ subnetwork used in organizations with many more devices than would be found in a home. The DMZ host feature designates one device on the home network to function outside of the firewall, where it acts as the DMZ while the rest of the home network lies inside the firewall. In some cases, a gaming console is chosen to be the DMZ host so that the firewall doesn't interfere with gaming. Also, the console is a good candidate for a DMZ host because it likely holds less sensitive information than a personal computer.
- **Industrial control systems (ICS).** DMZs provide a potential solution to the security risks of ICSes. Industrial equipment, such as turbine engines, or ICSes are being merged with information technology (IT), which makes production environments smarter and more efficient, but it also creates a larger threat surface. Much of the industrial or operational technology (OT) equipment connecting to the internet is not designed to handle attacks in the same way IT devices are. A DMZ can provide increased network segmentation that can make it harder for ransomware or other network threats to bridge the gap between IT systems and their more vulnerable OT counterparts.

The takeaway

A DMZ is a fundamental part of network security. These subnetworks create a layered security structure that lessens the chance of an attack and the severity if one happens. They are used to isolate a company's outward-facing applications from the corporate network. If a system or application faces the public internet, it should be put in a DMZ.

Learn how a honeypot can be placed in the DMZ to attract malicious traffic, keep it away from the internal network and let IT study its behavior.

This was last updated in July 2021

Continue Reading About DMZ in networking

- Do DMZ networks still provide security benefits for enterprises?
- Prevent a network security attack by isolating the infrastructure
- SASE challenges include network security roles, product choice
- Proper network segments may prevent the next breach
- 3 DDoS mitigation strategies for enterprise networks

Related Terms

AAA server (authentication, authorization and accounting)

An AAA server is a server program that handles user requests for access to computer resources and, for an enterprise, provides ... See complete definition

Luhn algorithm (modulus 10)

The Luhn algorithm, also called modulus 10 or modulus 10 algorithm, is a simple mathematical formula used to validate a user's ... See complete definition

Open System Authentication (OSA)

Open System Authentication (OSA) is a process by which a computer could gain access to a wireless network that uses the Wired ... See complete definition

Dig Deeper on Network security

screened subnet

By: Rahul Awati

access control list (ACL)

By: Ben Lutkevich

What to know about VPN termination

By: Terry Slattery

Security Think Tank: Practical steps to achieve zero trust

By: Petra Wenham

SearchCloudSecurity

Cloud security still needs a lot more work, say European experts

Security and privacy remain a stumbling block for cloud computing, according to information experts at the Trust in the Digital ...

Amazon Workspaces gets MFA security update

Amazon Web Services has added multifactor authentication to its WorkSpaces cloud desktop service, the first step in a larger ...

Next-Generation Firewall (NGFW)

Learn why Fortinet is a Leader in the 2021 Gartner® Magic Quadrant™ for Network Firewalls and received highest scores in 3 out of 5 use cases in the companion Critical Capabilities report

DOWNLOAD BOTH REPORTS

Overview

FortiGate NGFWs deliver industry-leading enterprise security for any edge at any scale with full visibility and threat protection. Organizations can weave security deep into the hybrid IT architecture and build security-driven networks to achieve:

- Ultra-fast security, end to end
- Consistent real-time defense with FortiGuard Services
- Excellent user experience with security processing units
- Operational efficiency and automated workflows

FortiGate NGFWs enable organizations to build high-performance, ultra-scalable, and security-driven networks

To ensure malware doesn't slip into your network via encrypted traffic, high-performance, reliable inspection must be ensured. See how FortiGate 7121F delivers in one of the most important performance-intensive tests of SSL deep inspection and threat protection.

Watch Now »

What's new in FortiOS 7.2

Packed with new features and enhancements, FortiOS 7.2 delivers a powerful combination of AI-driven actionable intelligence, SOC and NOC process automation, and inline prevention for evasive and previously unknown threats.

New in FortiOS 7.2 for FortiGate NGFW:

- **HTTP/3 support:** Inspection into HTTP/3 and QUIC delivers better visibility, superior protection and support for emerging standards
- **Unified policy**: Unified firewall policy configuration means all policies are unified in a single place, including ZTNA.
- **SaaS application segmentation:** This allows inline CASB protection for SaaS applications.

Learn More

Models and Specifications

Case Studies / Reviews

Services

Resources

Ecosystem

Training and Certification

Free Product Demo

Features and Benefits

FULL VISIBILITY AND PROTECTION

Stop ransomware, command-and-control attacks, and other hidden threats with SSL inspection (including TLS 1.3) and automated threat protection.

FORTIGUARD SECURITY SERVICES

Consolidate and concurrently run IPS, web and

video filtering, and DNS security services to reduce costs and manage risks.

NATIVELY INTEGRATED PROXY

Add FortiClient and deliver seamless user experience and security to the hybrid workforce with Zero Trust Network Access (ZTNA).

HYPERSCALE SECURITY

Build ultra-scalable security-driven networks to meet escalating business demands.

SECURITY FABRIC INTEGRATION

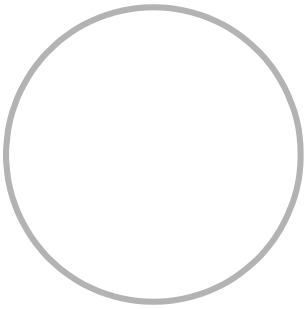
Share actionable threat intelligence across the entire attack surface to build a consistent and coordinated end-to-end security posture.

AUTOMATION-DRIVEN NETWORK MANAGEMENT

Build large-scale and efficient operations with an easy-to-use centralized management console.

FortiGate NGFW Use Cases

As network edges explode, you need effective security everywhere. With FortiGate, you can:

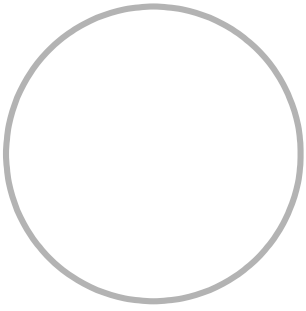


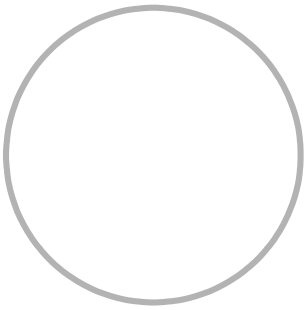
SECURE HYBRID, MULTI-CLOUD

Weave security deep into hybrid data center networks to secure any edge at any scale with end-to-end security across multiple clouds.

SEGMENT, PREVENT LATERAL SPREAD

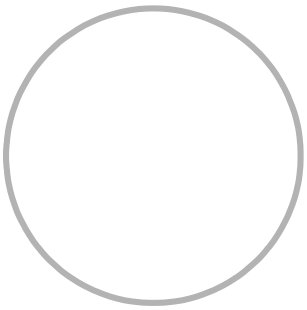
Prevent lateral spread, manage internal risks, and enforce security for any segmentation whether VXLAN-based, network, endpoint, or application. Achieve dynamic trust and port-level segmentation with Fortinet Security Fabric integration.





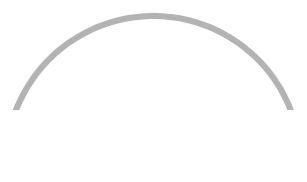
MANAGE VULNERABILITIES AND STOP THREATS

Protect against known and zero-day attacks, plus get virtual patching with included FortiGuard IPS.



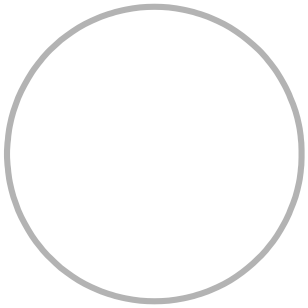
PROTECT USERS, PERIMETER

Gain full visibility, detect and remediate ransomware and other threats hiding in HTTPS traffic without performance impact.



SECURE HYPERSCALE ARCHITECTURES

Deliver hyperscale security that performs efficiently, with no network impact, to meet escalating business demands.



SECURE INDUSTRIAL AND OT ENVIRONMENTS

Deliver enterprise security for operational technology (OT) environments with FortiGate Rugged NGFWs. Gain full network visibility and threat protection.

Security-Driven Networking

Traditional security strategies can’t keep up with the challenges of your expanding

attack surface – from remote work, to mobility, to multi-cloud networks. **Fortinet Security-Driven Networking** addresses these challenges by tightly integrating network

infrastructure with security architecture, meaning your network will remain secure as it scales and changes.

Next-Generation Firewall (NGFW)

eBook

Five Mistakes to Avoid While Securing a Hybrid Network

Protecting Every Edge to Make Hackers' Jobs Harder, Not Yours

Hybrid Data Centers Require Additional NGFW Capabilities

Solution Guides

Protect the Expanding Attack Surface with FortiGate

How to Achieve Optimal Internal Segmentation

FortiGate NGFWs Provide Data Center Security

Power Advanced Research with Scalable, High-performance Security

7 Critical Considerations for Firewall Performance

White Papers

Grow Business with Secure Hybrid/Hyperscale Data Centers

Power Security at the Speed of the Business

Find solution guides, eBooks, data sheets, analyst reports, and more. Go to Resource Center >

Learn more about Fortinet Next-Generation Firewalls Contact Us >

FortiGate: Next-Generation Firewall News

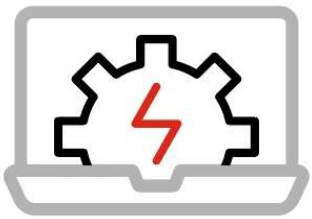
NEW FORTINET FIREWALL INCREASES SECURITY AND NETWORKING CONVERGENCE ACROSS HYBRID IT TO ENABLE SECURE DIGITAL ACCELERATION

FortiGate 3000F is the latest FortiGate NGFW powered by NP7 SPUs to deliver scalable, high-performance convergence of networking and security to enable Security-Driven Networking

RANKED #1 IN THRI GARTNER® CRI

Use Cases: Enterpris

Quick Links



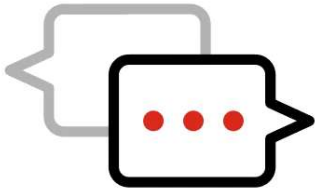
Free Product Demo

Explore key features and capabilities, and experience user interfaces.

Resource Center

Download from a wide range of educational material and documents.

Get Free Cyber Threat Assessment Program



Contact Sales

Have a question? We're here to help.

PRODUCTS

PARTNERS

DISCOVER MORE

CONNECT WITH US

Enter Email Address

I want to receive news and product emails. Read our privacy policy.

Also of Interest

entry level firewall

high end firewall

Network Firewall Price: Comparing Security Costs

Generation Firewall

By: Danny Mareco

October 16, 2013

0

Category: Healthcare, Hospitality, K-12 Education, Higher Education, Retail, Industrial, Security

In our dealings with technology leaders at schools, universities and hospitals there are many different opinions as to how they would each go about solving

Features to Look for in Your Next Firewall:

1. Identify and control applications on any port.

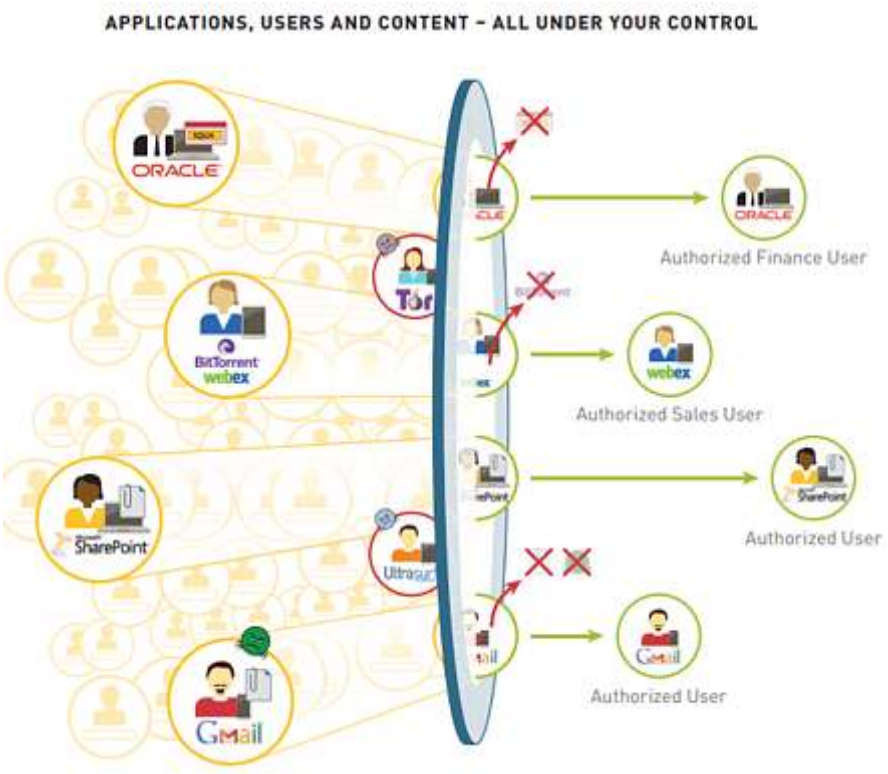
With the rise of mobile devices and the multitude of operating systems, application development has also become easier, which opens the door for less quality control prior to applications being made available to the public. Since application developers no longer stick to the old standard of port/protocol/application mapping, more applications are now capable of operating on non-standard ports or can hop ports (e.g., Facebook or Yahoo! instant messaging applications, peer-to-peer file sharing like UTorrent, or VOIP). Today's younger users are increasingly savvy enough to force some applications to run over non-standard ports (e.g., MS RDP, SSH). While the 4-year-old at our house racing through the gaming applications may not be at that level just yet, the point is that children are growing up with these devices in their hands while we were stuck sitting with a Commodore 64 or a Tandy 80, so imagine what they'll be capable of in 20 years.

2. Identify and control circumventors

Most organizations have network security policies and controls designed to enforce those policies they have spent so much time trying to implement. Villainous hackers use proxies, remote access, and encrypted tunnel applications which specifically used to circumvent security controls like firewalls as they attempt to attack networks. Without the ability to control these circumventors, organizations cannot enforce their security policies and thus expose themselves to the very risks they imagined their controls would prevent. Your next firewall solution must be capable of dealing with these types of circumventors while also ensuring the application intelligence is regularly updated.

3. Decrypt outbound SSL

traffic via policy. An example here would be web traffic from a healthcare organization - huge liability and data would be exposed to risk.



4. Provide application function control

Many applications have significantly different functions, presenting different risk profiles and value of good and bad. Some examples of this include WebEx vs. WebEx Desktop Sharing and Yahoo! Instant Messaging vs. the file transfer feature. In regulated corporate IT environments, or in organizations heavily dependent on their sensitive intellectual property, this is a significant issue. (Think about Coca-Cola's original recipe being compromised online). Your next firewall must continually evaluate the traffic and watch for changes - if a different function or feature is introduced during the session, the firewall should note it and perform a policy check, while maintaining full functionality.

5. Scan for viruses and malware in allowed applications

Enterprises continue to adopt collaborative applications hosted outside the physical locations. Whether it's hosted Sharepoint, Box.net, Google Docs or

There is always going to be unknown traffic and it will always represent significant risks to any organization, large or small. There are several important elements to consider with unknown traffic- it can be minimized, custom applications can be easily characterized so they are "known" in any network security policy and having predictable visibility and policy control over traffic that remains unknown. Your next firewall must attempt to classify all traffic, which provides a positive enforcement model (default deny). Solutions with a negative (default allow) model allows all unknown traffic. In that case, what you don't know will indeed hurt you - so "allow all" should not be the foundation of your policy structure.

7. Identify and control applications sharing the same connection

Applications share sessions. To ensure that users are continuously using an application "platform" such as Google, Facebook, Microsoft or Salesforce, application developers integrate many different applications - which often have very different risk profiles and business value. To better explain, here's an example with Gmail, which has the ability to spawn a Google Talk session from within the Gmail user interface. These are fundamentally different applications and your next firewall must recognize that and enable the appropriate policy response for each of them.

8. Enable the same visibility and control for remote users

Users are increasingly outside the four walls of the school, university or hospital. A significant portion of the enterprise user population is now capable of working remotely and they expect to connect to their applications via WiFi, wireless broadband or by any means necessary to complete their job duties. Regardless of the location of the user and the network they are accessing (home, library, coffee shop) your internal network, or even where the application they're employing might be, the same standard of control should apply. If your next firewall enables application visibility and control over traffic inside the four walls of your enterprise but not outside, it misses the mark on some of the most risky traffic.

going to increase complexity by several orders of magnitude. (Doesn't that thought make your head hurt?) Your next firewall must apply policy based on the user and application which significantly simplifies policy modeling and management. (May want to check and see if this is a feature that other pieces of your network infrastructure allow as well - would make integration much easier).

10. Deliver the same throughput and performance with application control fully activated

Many enterprises struggle with what many consider to be a forced compromise between performance and security. Previously, enabling network security features meant turning down throughput and performance. If your next firewall is built the right way, this compromise isn't necessary. Given the requirement for computationally intensive tasks like application identification that are performed on high traffic volumes with low latency, your next firewall must have hardware optimized for specific tasks such as networking, security and content scanning and perform all those tasks without sacrificing speed or safety.

11. Price or Cost

So many times we go out looking for a solution but also have a budget in mind. The goal is to get the most features at the lowest possible price. It happens when you're buying cars, toys and even groceries. So, when considering your next enterprise-grade firewall solution, it's important to remember the cost associated with NOT moving forward. How much tolerance do you have for price vs. cost? Risk can come in all shapes and sizes, but if you have a solution in place that not only protects your organization but helps you sleep easier at night knowing that sensitive data and your network are safe from attack certainly has to be worth any extra upfront costs.

There are a number of solutions in the marketplace now that handle the above scenarios and they have various price tags attached to the physical appliance. When it comes to firewalls, we recommend Palo Alto Firewall,

First Name*

Last Name

Email*

Comment*

protected by reCAPTCHA
Privacy - Terms

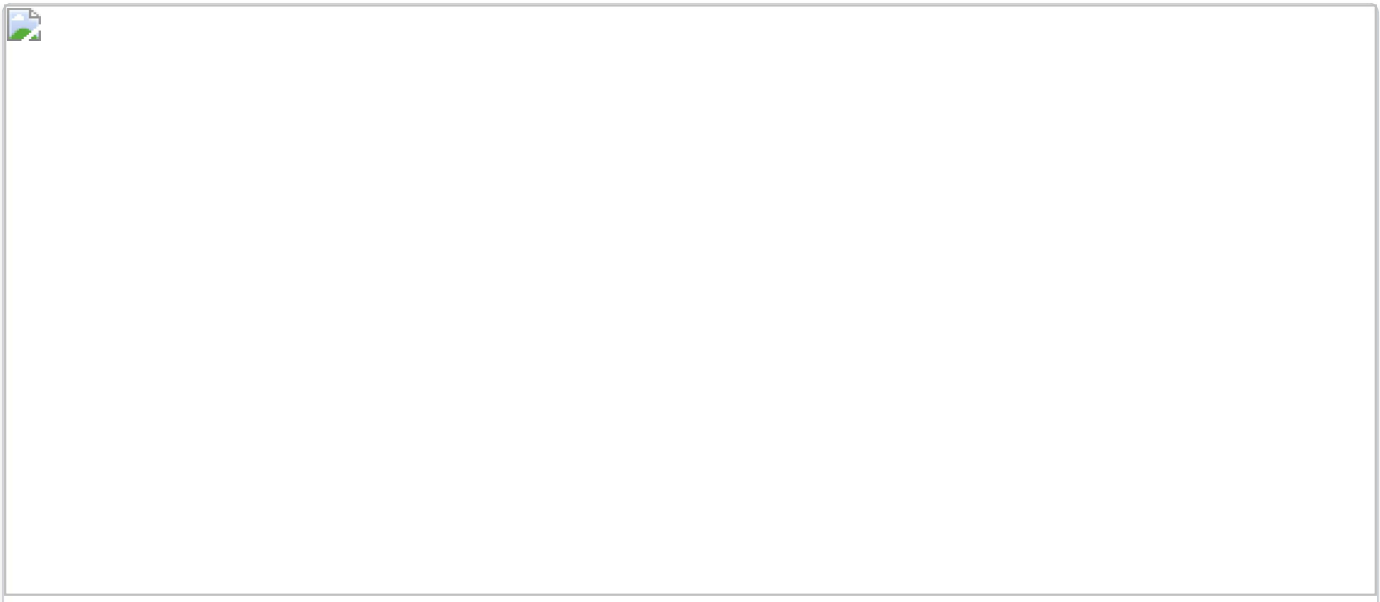
Related Articles

Medical Grade WiFi: Essential Features for Every Hospital or Clinic

HEALTHCARE MANAGED WIFI

3 Ways WiFi has Changed the Healthcare Industry and How to Prepare

HEALTHCARE DESIGN MANAGED WIFI PERFORMANCE



WiFi as a Service: A Game-Changer for Businesses in 2018

HEALTHCARE HOSPITALITY K-12 EDUCATION HIGHER EDUCATION RETAIL INDUSTRIAL STRATEGY SMB MANAGED WIFI

Services
Software
Managed
WiFi
Solutions
Pricing
Contact Us

Why SecurEdge

About Us
Case Studies
Methodology
Careers

How-To

WiFi Blog

Tools

WiFi Calculator

Stay up to date with SecurEdge

Enter your email*

Subscribe

What is LDAP used for?

Asked 13 years, 6 months ago Modified 3 years, 11 months ago Viewed 198k times

I know that LDAP is used to provide some information and to help facilitate authorization.

162

But what are the other usages of LDAP?

ldap

60

Share Follow

edited Jan 11, 2011 at 18:10

Kev

115k 50 293 378

asked Oct 27, 2008 at 9:13

gizmo

16 Answers

Sorted by:

Reset to default

Trending (recent votes count more)

I will focus on why using LDAP, not what is LDAP.

276

The use model is similar like how people use library cards or phonebooks. When you have a task that requires "write/update once, read/query many times", you might consider using LDAP. LDAP is designed to provide extremely fast read/query performance for a large scale of dataset. Typically you want to store only a small piece of information for each entry. The add/delete/update performance is relatively slower compared with read/query because the assumption is that you don't do "update" that often.

Imagine you have a website that has a million registered users with thousands of page requests per second. Without LDAP, every time users click a page, even for static page viewing, you will probably need to interact with your database to validate the user ID and its digital signature for this login session. Obviously, the query to your database for user-validation will become your bottleneck. By using LDAP, you can easily offload the user validation and gain significant performance improvement. Essentially, in this example, LDAP is another optimization layer outside your database to enhance performance, not replacing any database functions.

LDAP is not just for user validation, any task that has the following properties might be a good use case for LDAP:

1. You need to locate ONE piece of data many times and you want it fast
2. You don't care about the logic and relations between different data
3. You don't update, add, or delete the data very often
4. The size of each data entry is small
5. You don't mind having all these small pieces of data at a centralized place

Share Follow

edited Jun 21, 2016 at 2:25

Iharob Al Asimi
51.9k 5 57 93

answered Feb 13, 2014 at 20:38

user3307545
2,777 1 10 2

11 Very good description! You answered the very question I keep asking myself each time I read LDAP : Why? Thanks! – DhafirNz Oct 29, 2015 at 23:46

1 is LDAP suitable for internet applications? in the sense NON enterprise internal applications? – pinkpanther Mar 7, 2017 at 6:16

1 are there any alternatives for ldap? – Shekhar Reddy Mar 15, 2018 at 5:50

8 What do you mean by saying "By using LDAP, you can offload the user validation.."? With LDAP you post a query to an LDAP server which (probably) uses a databases: isn't it the same bottleneck? – Marco Stramezzi Jun 12, 2018 at 9:18

@user3307545 Very nice description! Just loved it. Instead of giving a textbook answer you have just explained how it is used in real time.Thank you! – Alekya May 29, 2019 at 6:27

76

That's a rather large question.

LDAP is a protocol for accessing a directory. A directory contains objects; generally those related to users, groups, computers, printers and so on; company structure information (although frankly you can extend it and store anything in there).

LDAP gives you query methods to add, update and remove objects within a directory (and a bunch more, but those are the central ones).

What LDAP does not do is provide a database; a database provides LDAP access to itself, not the other way around. It is much more than signup.

Share Follow

answered Oct 27, 2008 at 9:23

blowdart
53.9k 11 110 146

50

Well, there are LDAP servers and the LDAP protocol. Combined, it's a data store, or a database. It's not relational, but it's just a place to store data, and it's optimized to be efficient at reads more than writes. It doesn't support transactions.

Now, it happens to be very popular for storing credentials, but that's by no means its only purpose, and not its original purpose.

Share Follow

answered Feb 26, 2009 at 20:38

Don Branson

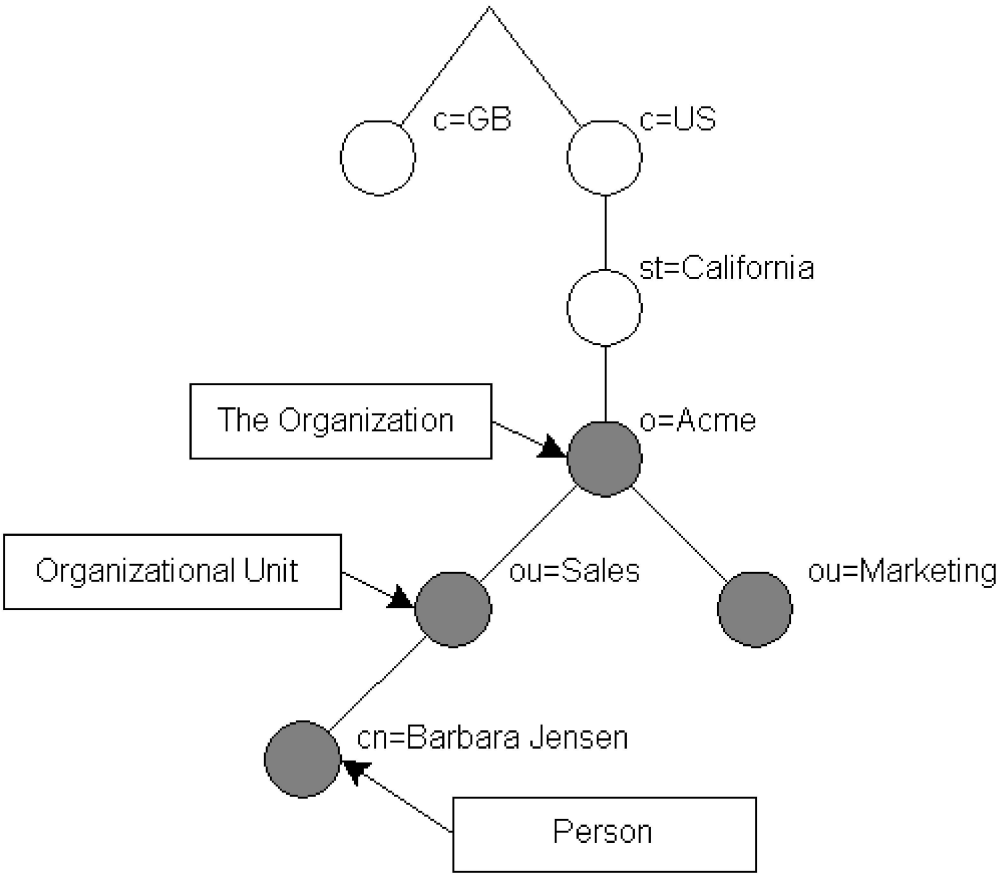
13.4k 10 57 100

32

LDAP stands for Lightweight Directory Access Protocol. As the name suggests, it is a lightweight protocol for accessing directory services, specifically X.500-based directory services. LDAP runs over TCP/IP or other connection oriented transfer services. The nitty-gritty details of LDAP are defined in RFC2251 "The Lightweight Directory Access Protocol (v3)" and other documents comprising the technical specification RFC3377. This section gives an overview of LDAP from a user's perspective.

What kind of information can be stored in the directory? The LDAP information model is based on entries. An entry is a collection of attributes that has a globally-unique Distinguished Name (DN). The DN is used to refer to the entry unambiguously. Each of the entry's attributes has a type and one or more values. The types are typically mnemonic strings, like `cn` for common name, or `mail` for email address. The syntax of values depend on the attribute type. For example, a `cn` attribute might contain the value `Babs Jensen`. A mail attribute might contain the value `babs@example.com`. A `jpegPhoto` attribute would contain a photograph in the JPEG (binary) format.

How is the information arranged? In LDAP, directory entries are arranged in a hierarchical tree-like structure.



Share Follow

edited Aug 31, 2017 at 5:59

Matt Komarnicki

4,752 6 35 77

answered Jun 3, 2015 at 10:21

Nitin Pawar

1,518 16 14

7

- LDAP main usage is to provider faster retrieval of data . It acts as a central repository for storing user details that can be accessed by various application at same time .
- The data that is read various time but we rarely update the data then LDAP is better option as it is faster to read in it because of its structure but updating(add/updatee or delete) is bit tedious job in case of LDAP
- Security provided by LDAP : LDAP can work with SSL & TLS and thus can be used for sensitive information .
- LDAP also can work with number of database providing greater flexibility to choose database best suited for our environment
- Can be a better option for synchronising information between master and its replicase
- LDAP apart from supporting the data recovery capability .Also , allows us to export data into LDIF file that can be read by various software available in the market

Share Follow

answered Jun 7, 2017 at 17:12

harpreet
96 1 3

17

LDAP is the Lightweight Directory Access Protocol. Basically, it's a protocol used to access data from a database (or other source) and it's mostly suited for large numbers of queries and minimal updates (the sort of thing you would use for login information for example).

LDAP doesn't itself provide a database, just a means to query data in the database.

Share Follow

answered Oct 27, 2008 at 9:42

paxdiablo
811k 224 1528 1890

9

The main benefit of using LDAP is that information for an entire organization can be consolidated into a central repository. For example, rather than managing user lists for each group within an organization, LDAP can be used as a central directory accessible from anywhere on the network. And because LDAP supports Secure Sockets Layer (SSL) and Transport Layer Security (TLS), sensitive data can be protected from prying eyes.

LDAP also supports a number of back-end databases in which to store directories. This allows administrators the flexibility to deploy the database best suited for the type of information the server is to disseminate. Because LDAP also has a well-defined client Application Programming Interface (API), the number of LDAP-enabled applications are numerous and increasing in quantity and quality.

Share Follow

answered May 15, 2013 at 9:59

Aditya
91 1 1

17

The main idea of LDAP is to keep in one place all the information of a user (contact details, login, password, permissions), so that it is easier to maintain by network administrators. For example you can:

- use the same login/passwd to login on an Intranet and on your local computer.
- give specific permissions to a group of user. For example some could access some specific page of your Intranet, or some specific directories on a shared drive.
- get all the contact details of the people in a company on Outlook for example.

Share Follow

answered Oct 27, 2008 at 9:51

Mapad
8,044 5 38 40

15 No no no. That's the function of a directory, *not* of LDAP. That's like saying the function of SQL is to provide a database; it isn't; directories provide LDAP access, not the other way around. – blowdart Oct 27, 2008 at 9:53

5 I have to agree, this is mistaking the data store for the access protocol. – geoffc Nov 20, 2008 at 17:46

@blowdart Mainly this answer is very useful and concise, but it is for LDAP Server as I gathered from info all around. – Geeocode Jul 25, 2018 at 15:17

1 Guys, you are arguing that miso-soup is not a soup. Technically it is correct. But for practical reasons the answer is very good, because it simply explains WHY and WHERE that "obscure LDAP thing" is used, putting aside the technical explanation. Most people in that world have no idea what "Directory" is, or how DB works. So simply, running LDAP on server allows storing all the network credentials and serving all the domain users in one place. Yes, there are dependencies to DBs, tricky wrappers for serving unix and windows credentials at the same server, but it tells nothing to simple user. – Asdf Jul 24, 2019 at 8:31

7

I have had the opportunity to start a project for school about ldap, from scratch, but before getting to know what is ldap, I had to understand what is a directory, there are many (most used directories are novell and windows), here you can see what the directory in Wikipedia.

And ldap is the protocol to communicate with the board, one of the best books I've found is this one.

Share Follow

edited May 9, 2018 at 15:35

answered Jul 11, 2012 at 22:16

samaniego
149 2 9

5

In Windows Server LDAP is a protocol which is used for access Active Directory object, user authentication, authorization.

Share Follow

answered Mar 25, 2014 at 6:02

Anirban_ITPro


51 1 3

6

LDAP is just a protocol to access structured information. LDAP has standardized local and remote access to information whereas in case of database there is only standardized local access and remote access is proprietary.

LDAP server is optimized for search operation and database for transactions(Insert/Delete).

For more information refer following link:

http://www.itszero.in/2013/09/what-is-ldap-ad-adam.html

Share Follow

answered Sep 7, 2013 at 16:41


AnandSonake
518 8 18

3

To take the definitions the other mentioned earlier a bit further, how about this perspective...

LDAP is Lightweight Directory Access Protocol. DAP, is an X.500 notion, and in X.500 is VERY heavy weight! (It sort of requires a full 7 layer ISO network stack, which basically only IBM's SNA protocol ever realistically implemented).

There are many other approaches to DAP. Novell has one called NDAP (NCP Novell Core Protocols are the transport, and NDAP is how it reads the directory).

LDAP is just a very lightweight DAP, as the name suggests.

Share Follow

answered Nov 20, 2008 at 17:49


geoffc
3,954 6 42 49

0

Light weight directory access protocal is used to authenticate users to access AD information

Share Follow

answered Dec 11, 2013 at 9:32


kudrat
19 1

4

Well, LDAP is a protocol(way) to access structured info. LDAP uses client-server model so, LDAP client makes request to access required info. LDAP server stores info not in relational way but in attribute and value pair. You can use LDAP to assign same privilege to group of user or same credential to access multiple services. For more details refer following link : http://www.zytrax.com/books/ldap/ch2/

Share Follow

answered Aug 16, 2013 at 9:15

AnandSonake
518 8 18

2

LDAP stands for Lightweight Directory Access Protocol.It is used in Active Directory for communicating user queries..e.g.. LDAP can be used by users to search and locate a particular object like a laser printer in a domain.

Share Follow

edited Jul 21, 2013 at 8:18

Baby Groot
4,617 39 52 69

answered Jul 21, 2013 at 7:59

Arun Varshney
21 1

0

LDAP is also used to store your credentials in a network security system and retrieve it with your password and decrypted key giving you access to the services.

Share Follow

answered Feb 26, 2009 at 20:34

Naushin Shaikh