Setting up single sign-on with Zendesk® Remote Authentication

Zendesk Inc.

Notice

Copyright and trademark notice

Setting up single sign-on with Zendesk® Remote Authentication

Zendesk - 989 Market St, Ste 300 - San Francisco - CA 94103 - USA

www.zendesk.com

Contents

Chapter 1

Setting up single sign-on with Zendesk Remote Authentication

As of 4/29/13, single sign-on using Zendesk Remote Authentication has been deprecated and replaced with JSON Web Token (JWT). If you've already set up single sign-on with Zendesk Remote Authentication, it will continue to work but we recommend that you upgrade to JWT. See *Setting up single sign-on with JWT (JSON Web Token)*.

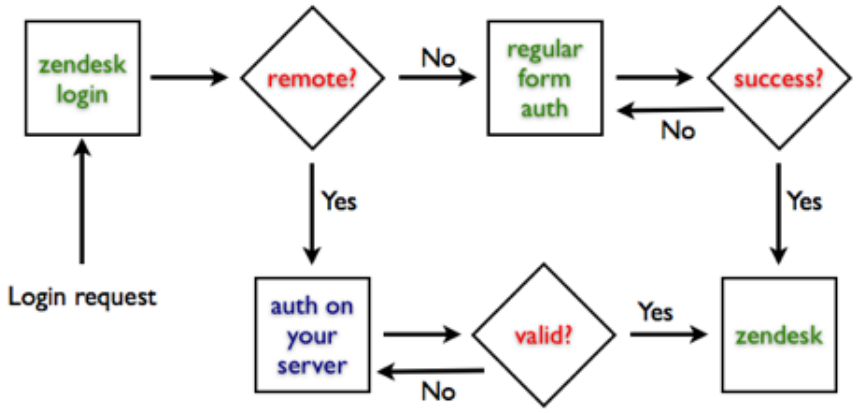
Remote authentication allows you to authenticate Zendesk users using a locally hosted script. In order to enable remote authentication, go to the Account menu and choose Settings > Security – you must be an Admin in order to do this.

The principles

Remote authentication is based on a “shared secret” between your local authenticating script and Zendesk. This secret is used to securely generate a hash (one-way encryption) which Zendesk uses to ensure that people who log on to your account using remote authentication, are who they claim to be, and have been pre-approved to do so by implicitly knowing the “shared secret”.

That was a somewhat abstract explanation, here’s what happens in simple steps:

1. Client attempts to log on to a Zendesk account which has remote authentication enabled
2. Client is from an IP address which Zendesk is configured to authenticate remotely
3. Zendesk redirects client to the remote authentication URL, adding a time stamp
4. The script running at the remote authentication URL (i.e. on your web server) gets a request
5. The script routes the client via your local/internal authentication mechanism to ensure that the client is logged on (to e.g. the domain)
6. The script retrieves the client email address and name and creates a hash value of this alongside the “shared secret” and time stamp
7. The script redirects the client back to Zendesk
8. Zendesk gets the request, and also hashes the same parameters using the “shared secret”. If the hashes match, Zendesk know that this request has been authorized by your local script. The user gets logged in.



This means that the task at hand, is to write the authentication script which Zendesk can then redirect the login request to, determine the authenticity of the user sending this request, and either display a login error to the user, or authenticate the user and redirect him back to Zendesk. You can download a commented sample script for an ActiveDirectory/IIS setup on the page where you configure your remote authentication setup

Security notes

The server your users authenticate against does not have to be available via the internet, as long as they can reach it by their browser. These are simple redirects, there is **no inbound connection to your facilities from Zendesk**.

Parameters

When you redirect back to Zendesk, you must hit the URL https://domain.zendesk.com/access/remoteauth and include all the parameters that were sent to you plus the following parameters:

Parameter	Definition	Mandatory?
name	The name of the user, minimum 2 charactersThis name gets set on the user, whether it's a new user or an existing user	✔
email	Valid email address for the userUsed to find the user unless a valid external_id is provided (more below). Users get created on the fly if no user by this email address exists in Zendesk.	✔
timestamp	The timestamp sent to your authentication script by Zendesk	✔

hash

```
input = name + "|" + email
 + "|" + external_id
 + "|" + organization
 + "|" + tags + "|" +
 remote_photo_url + "|" +
 token + "|" + timestamp
hash  = md5(input)
```

In the above, use an empty string for the values that you do not make use of or send to Zendesk, e.g. in case you only make use of the mandatory parameters name, email, token and timestamp, then the input string will look like this:input = name + "|" + email + "|||||" + token + "|" + timestampRemember that order matters when you build the input string. You can see a sample Ruby implementation of the hashing mechanism *here*. Zendesk uses the hash value to validate that the account token is valid. For the timestamp, you should use the timestamp sent to you from Zendesk, or generate your own for a direct login call without having to go via your Zendesk login page first.

Since the vertical bar character is used as a delimiter, Zendesk requires that you convert any existing vertical bars in other parameters to an escape sequence: %7C. For example, if a user's external id is "123|enduser", then you would encode the input string as `name + "|" + email + "|" + "123%7Cenduser" + "||||" + token + "|" + timestamp`. No special handling is necessary when sending parameters to Zendesk, only when generating a signature so that Zendesk can verify the input.

external_id	A unique user identifier in your systemIf your users are uniquely identified by something else than an email address, and their email addresses are subject to change, send the unique id from your system.If you send along an external_id no user is found by that id, then we try to find the user by email address. If a user gets found by email address and has no external_id, or updating external id's is enabled, then we set the external_id of the user to the given value. If no user was found by either email or external_id, we create the user on the fly.Be aware, that email addresses are unique per Zendesk account and if the email address is in use by a user with different external_id, the login will fail.	—
organization	Organization to add the user toIf you have a logical grouping of users in your current system which you want to retain in Zendesk, this can be done setting the organization parameter. If you set the parameter in the query string, but do not supply the **name of an existing organization**, the user will be removed from his current organization (if any). If you do not set the parameter, nothing will happen.	—
tags	Tags to be put on the user profile. This should be in "tag1, tag2, tag3# format and URL encoded.**Note:** This is a PUT action, meaning any tags specified will be used to replace any existing tags on the user.	—
remote_photo_url	A profile image of the user. This must be a public image	—

	URL, which is not behind any authentication. Remote Authentication will not follow redirects.

Usage of the above parameters is elaborated in the sample script. But don't hesitate to contact us if there's something that's unclear to you.

User attributes get updated each time you login a user. This means that data from your system overrules data in Zendesk. You can disable user access to their profiles to further lock down users to your system. If a user is found by email, name gets updated. If a user is found byexternal_id, name and email get updated unless the email address is already in use by another user. In this case an error message gets displayed to the user.

Return URL

You can set a return URL to which the user gets redirected to when logging out of Zendesk or when Zendesk rejects the login. This URL gets parameterized with email and external_id if a such exists. Optionally, it will be parameterized with a kind and a messageparameter in case the user could not be logged in or created. For example: http://yourcompany.com/logout/?email=someone@someone.foo&kind=error&message=Invalid token

Skipping a routing step

It's perfectly possible to log directly into Zendesk using remote authentication, and not having the user go by the Zendesk login page, only to get redirected back to your authentication script. All you need to do, is to set the timestamp parameter on your own, the value should be *the number of seconds since epoch* (in UTC). The time stamp must be current.

Error messages

If Zendesk cannot login a user by remote authentication, the user will either get an error message displayed, or – if you have entered a return URL, an error will get sent back to that.

If we send an error to the return URL, there will be set two parameters on the URL message and kind. On an error, the value of kind will beerror and the message will be one of the following:

Message	Explanation
Invalid data from remote login mechanism. Missing name, email, hash or timestamp	One of the mandatory parameters were missing. Please verify that your local authentication script is correct.
Remote authentication timestamp expired	The timestamp was older than 30 minutes. Please create a new as elaborated above, and ensure that your servers clock is properly synchronized.
Invalid token for remote authentication, check that your security token is up to date	Verify that the token in your local authentication script is identical to the one on your Zendesk remote authentication configuration page.

User exists with different external_id	Happens if you have not enabled that user can have their external_idupdated via remote authentication and the external_id in the request does not correspond to the one found on the Zendesk user record. If a Zendesk user has no external_idand there is one in the request, the user record will get theexternal_id set, irregardless of the aforementioned “allow update” setting.
Failed to create user with given properties: …	It was not possible to create the user for the specified reason (after the colon). Could be an invalid email address, name or other.
Failed to update user with new properties: …	It was not possible to update the user for the specified reason (after the colon). Could be an invalid email address, name or other.

Use cases

This section describes how Zendesk acts under different use cases. The use cases revolve around the submitted email and external_id’s, and the current state of the users in the Zendesk account.

These use cases should be read independently of one another, i.e. there’s no “resulting state” that’s carried on to the next use case.

Auth request	Users in account / outcome
email=bob@domainexternal_id=123	None The user gets created with given attributes
email=bob@domainexternal_id=123	email:bob@domainexternal_id:456 If allow update of external ids is **enabled**:The user bob@domain gets hisexternal_id changed to 123If allow update of external ids is **disabled**:An attempt to create the user is made. This will fail as the email address is already in use by another user. The request gets redirected to the logout page with a kind=error parameter and a message=… parameter.
email=bob@domainexternal_id=123	email:bob@domainexternal_id:456email :joe@domainexternal_id:123 An error is returned to the remote auth calling page stating that the email address is already taken. The remote auth mechanism finds the user with the given external_id 123 and tries to

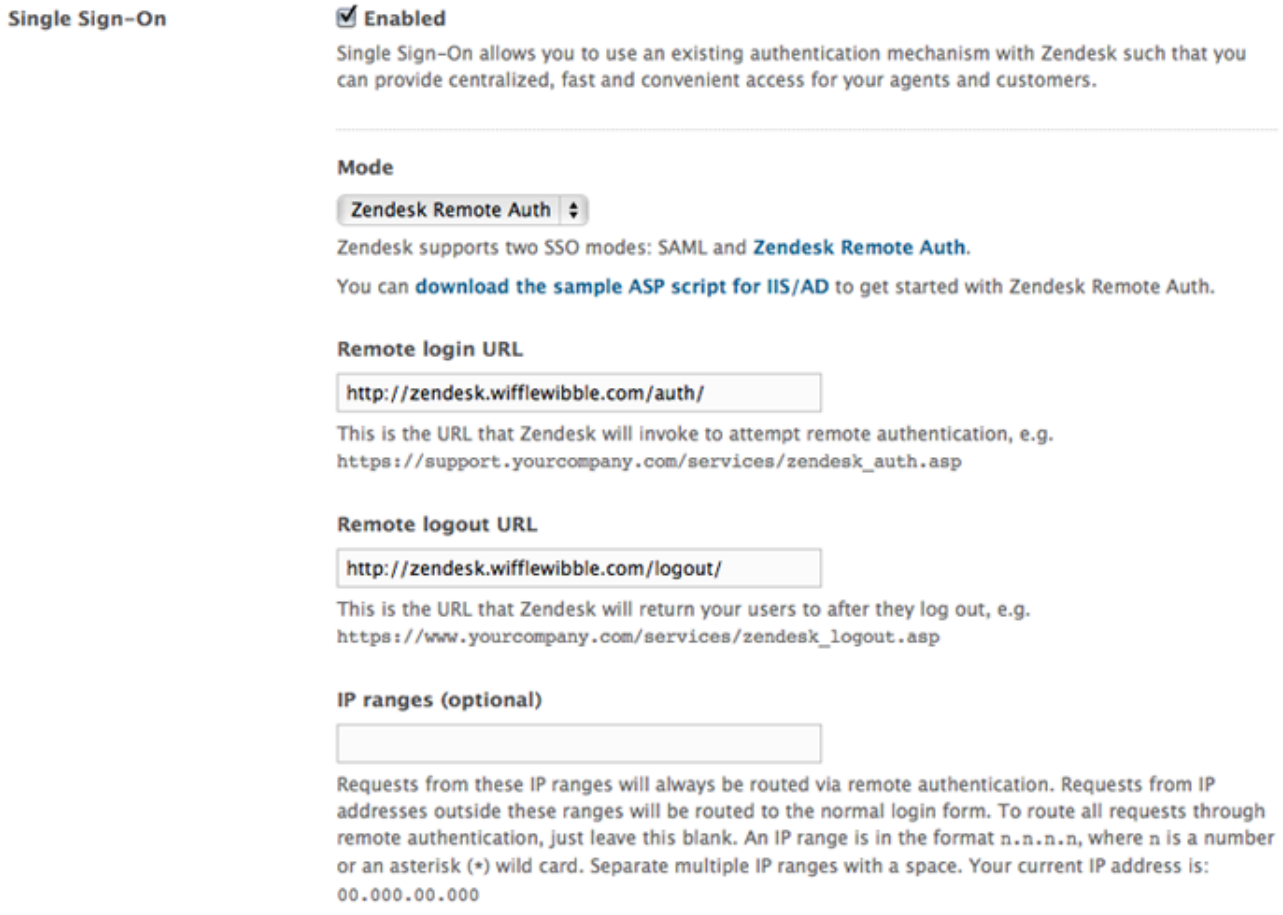
	set his email address to bob@domainwhich is already taken by the user withexternal_id 456
email=bob@domainexternal_id=123	email:joe@domainexternal_id:123 The user joe@domain gets his email address changed to bob@domain

Full use case example

Your new hot SaaS startup is going to offer remote authentication for your Zendesk account at .

Setting up Zendesk

Go to the “Settings” tab and click the “Security” menu item to enable Single Sign-On (Remote authentication).



Save. Now click the “ASP script for IIS/AD” link to the right, and download a preconfigured sample script for an IIS/AD setup. Since you left the IP Ranges blank, remote authentication will henceforth be enabled for all users, no matter where they come from.

Should you get unlucky and somehow lock yourself out of Zendesk, you make a mental note that the /access/normal/ URL at will let you use the regular login at any time.

Setting up your IIS/AD

You begin by reading the script downloaded, and skim the functionality.

1. Place this script in a folder on your IIS, and disable anonymous access for the script.
2. Add a valid user/password for the LDAP lookups by setting the variables sLdapReaderUsername ‘ and sLdapReaderPassword below. ‘ ‘ For debugging, call this script with debug=1 as a parameter (e.g. http://yourserver/sso.asp?debug=1).

   You’re going for external_id authentication, it doesn’t get more exciting than this. You configure the vital parts of the script, and you’re ready to go.

Logging in

You add a link to your intranet with the Zendesk login, and click the link.

Zendesk recognizes that the login request comes from an IP address for which remote authentication is enabled, and promptly redirects the request to where it hits your authentication script.

The authentication script gets the request, and validates the current domain login. The user is okay to to send to Zendesk. The user then gets redirected to Zendesk.

Once the redirect hits Zendesk, the values get verified. This includes checking that the timestamp is recent and that the hash value is correct. These pass. There’s no user by external_id=4 in Zendesk at this point, so the user found by email addressroger.wilco@wifflewibble.com gets this set, and the name gets updated in case they differ.

You’ve now logged in using remote authentication. You solve the tickets in your view, and logout. Zendesk terminates your Zendesk session and redirects you to the return URL defined under the “Remote authentication” page where your logout script resides, and writes a nice logout message.