Salt Group

Echidna Operations Guide

Version 15.2

October 2015

Contents

Chapter 1: Before starting

This *Operations Guide* is part of a set of books about Echidna. It describes how support staff manage users and tokens in Echidna. The system administrator uses the other Echidna guides.

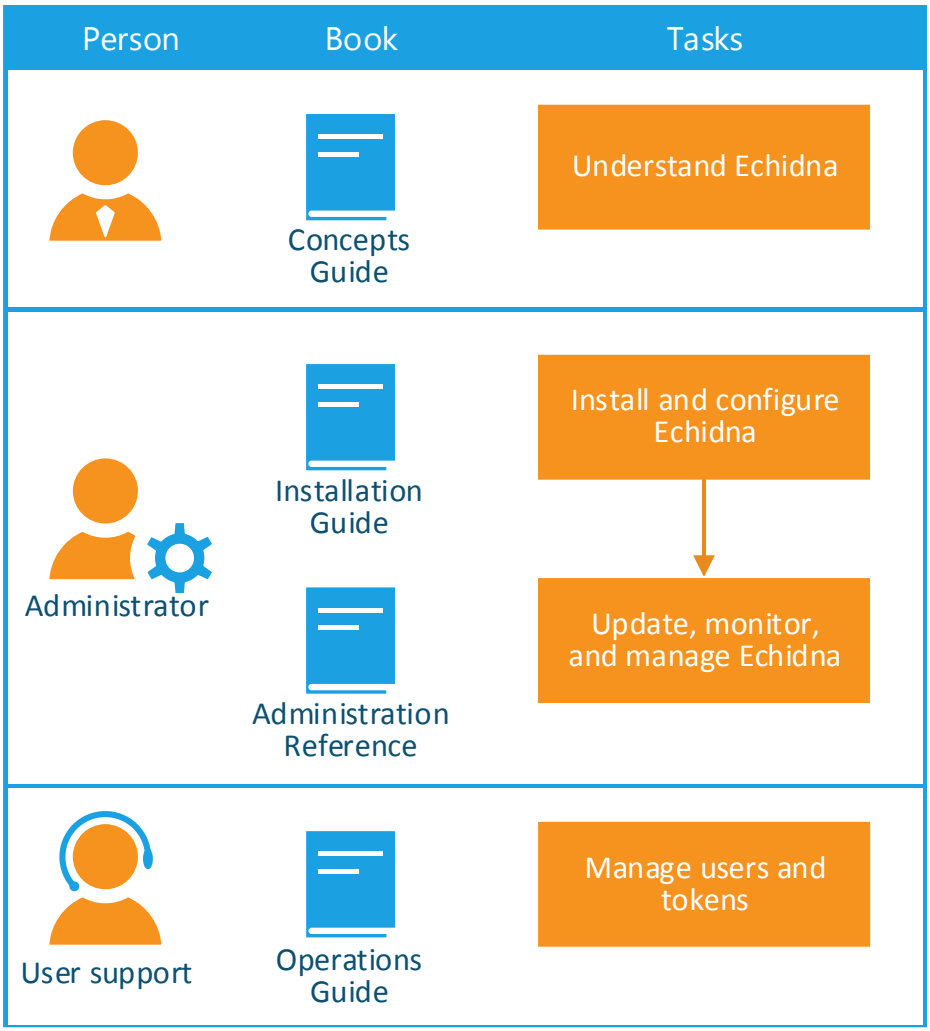


Figure 1: When to use each book in the Echidna documentation set

1.1 About Echidna

Echidna is an authentication server.

When a user tries to use a protected application, the application needs to check whether the user should be given access. The application sends a request to Echidna, and Echidna checks the user's credentials. Echidna tells the protected application if the user's credentials are correct.

Echidna is designed for two-factor authentication, where a user has to have at least two forms of authentication. For example, a user might need to give the correct password, and then supply the correct code from a app on their phone.

1.2 About tokens

Each user has one or more tokens. Echidna supports a wide range of tokens, including the following:

- User name and password
- Hardware token
- Soft token (such as an app on a phone or tablet)
- Temporary password
- CAPTCHA

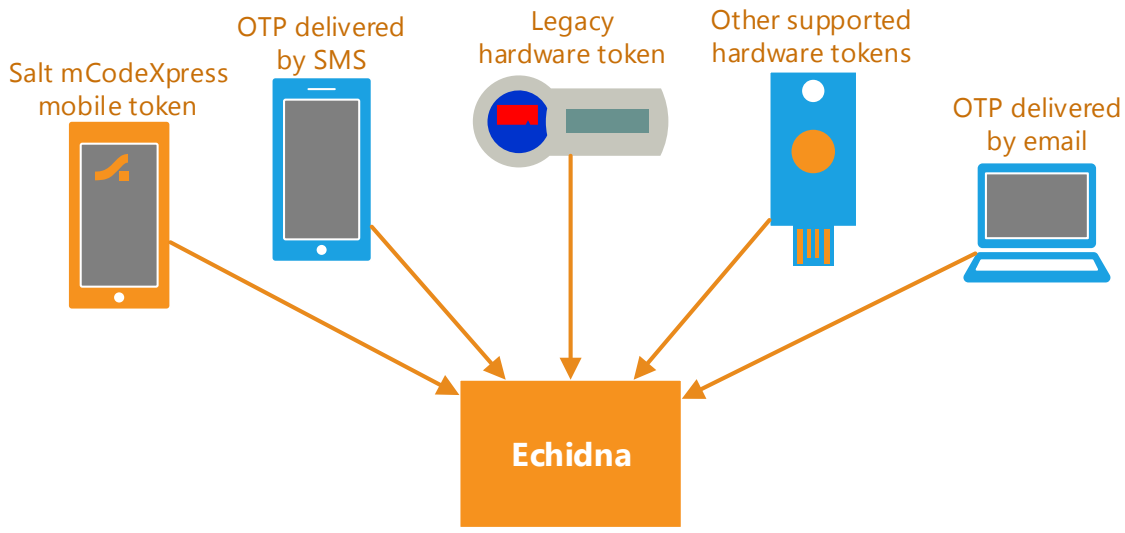


Figure 2: Echidna supports many different types of token

If the administrator has not configured Echidna to use a particular type of token, it will not appear in the User Console screens. For this reason, it is unlikely that the User Console screens will show all of the possible token types.

Echidna stores token records in a database.

1.3 About users

Echidna needs user records to authenticate against.

If an organisation already has a user store (such as a database or Active Directory), Echidna can connect to that user store. Alternatively, Echidna can store user accounts in its own database.

The following diagrams show some of the ways that Echidna can connect to user accounts.

1.3.1 Users are stored in an existing user store

This diagram shows an Echidna server that connects to an existing user store. It stores other data in its own database.

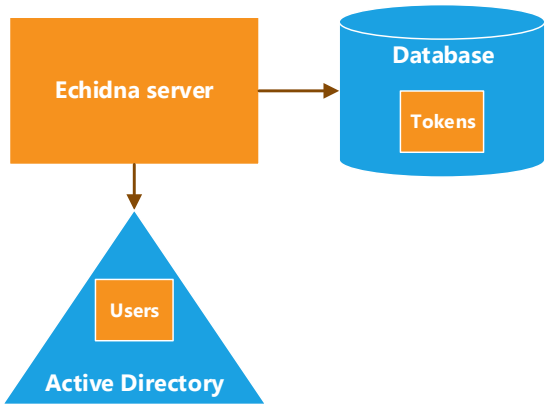


Figure 3: Echidna server connected to an existing Active Directory user store

If an organisation already has a user store that other applications use, Salt recommends that Echidna is not used to create and delete those users. Instead, continue to use the usual management tools. Use Echidna to create and delete only the authentication-related attributes.

1.3.2 Users are stored in the Echidna database

Organisations that do not already have a separate user store can store all users in the Echidna database, as shown here:

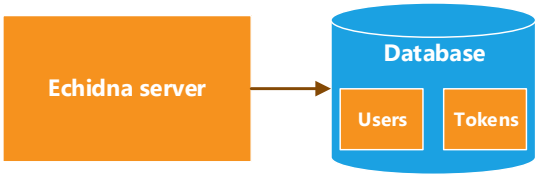


Figure 4: All users are stored in the Echidna database

This is useful if there is no existing user store, or if Echidna protects only a small number of user accounts. It is useful when the user accounts are not used by any other application.

In this situation, use the User Console to create and delete users, as described in this book.

1.4 Components of Echidna

Echidna consists of a central server, and three web consoles. This diagram shows the three consoles, and the people who use them:

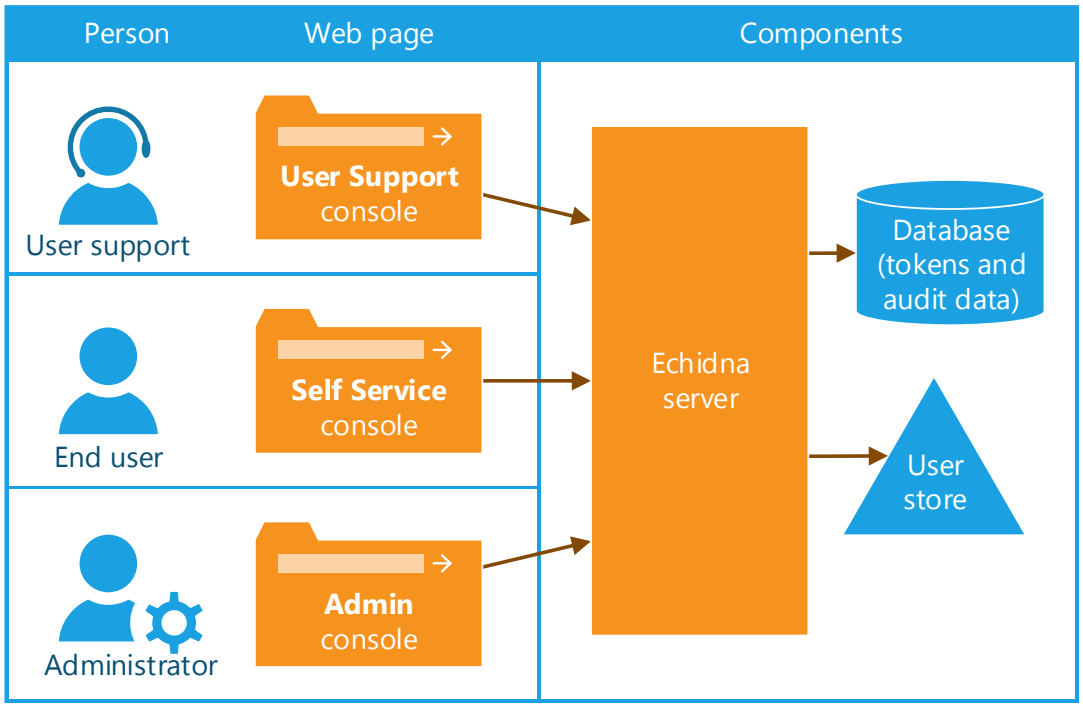


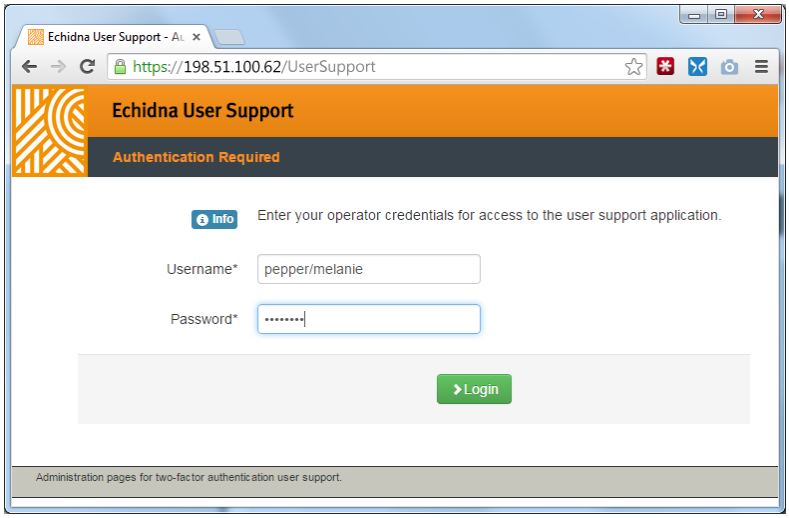
Figure 5: Components of Echidna, and who uses them

This book describes how to use the **User Support** and **Self Service** consoles:

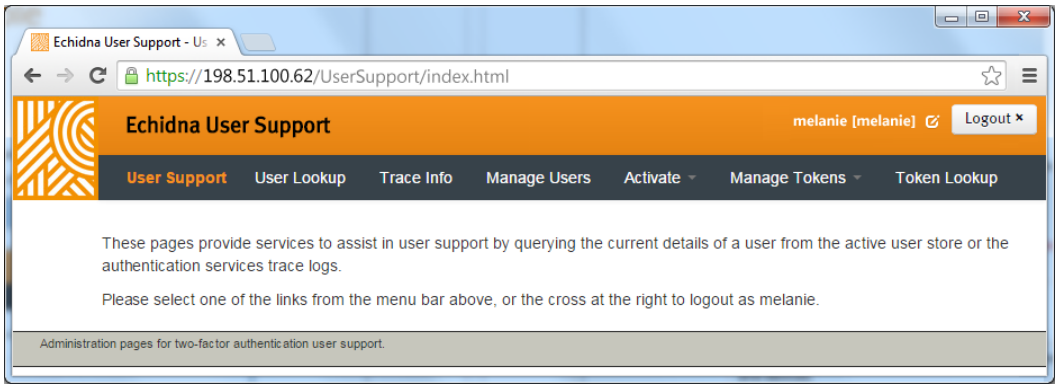
- The **User Support console** lets some staff members manage users and tokens.
- The **Self Service console** lets end users register their own supported tokens and manage their own login preferences.
- The **Administration console** lets administrators configure, monitor, and maintain Echidna. This book does not cover the Administration console. For more information, see the *Installation Guide* and the *Administration Reference*.

Chapter 2: Sign in to the User Support console

1. Ask the administrator for access to the **User Support** console. The administrator will supply the URL.
2. In a web browser, go to the URL. The User Support console appears. The following screenshot shows how the User Console appears if it has not been updated to match the organisation's look:



3. Sign in to the User Support console. The main page appears:



Chapter 3: Managing tokens

This chapter describes how to work with tokens using the User Support console. It contains these sections:

3.1 About tokens

The User Support console lets operators manage the tokens that have been set up for Echidna. If a token type has not been set up for Echidna and published through the web services, it does not appear in the User Console.

This table lists the types of token that can be set up for Echidna:

Authentication method	**Description**
Salt mCodeXpress OTP	The Salt mCodeXpress app is a soft token that works on many different phones and devices.
SMS-delivered OTP	OTPs are sent as a text message to the mobile phone number that is recorded in the user store.
OATH HOTP (Generic)	OATH event-based hardware tokens that produce OTPs.
OATH TOTP	OATH time-based hardware tokens that produce OTPs.
OATH OCRA	OATH time- or event-based hardware tokens that can produce OTPs and signatures.
OATH HOTP (YubiKey)	YubiKey hardware tokens that are programmed as OATH event-based OTP generators.
Limited-use temporary password	Temporary passwords for emergency use, such as when the main authentication method is unavailable (perhaps due to a lost token or phone). These passwords expire after a set number of uses or a set time. These can be delivered over the phone by user support staff.
Password-only	Authenticates against the user-store password only.
CAPTCHA images	Handles passwords from generated images, used to stop robot logins.

3.1.1 How token registration works

Users can use the **Self Service** console to register their own tokens. If an organisation has not set up the **Self Service** console, or if a user is having problems, support staff can register tokens on behalf of a user.

The options on the screen depend on which tokens the organisation uses. In the following example, the organisation uses OATH HOTP tokens (which appear in this menu as Generic Tokens), mCodeXpress tokens, YubiKey OATH tokens, and temporary passwords:

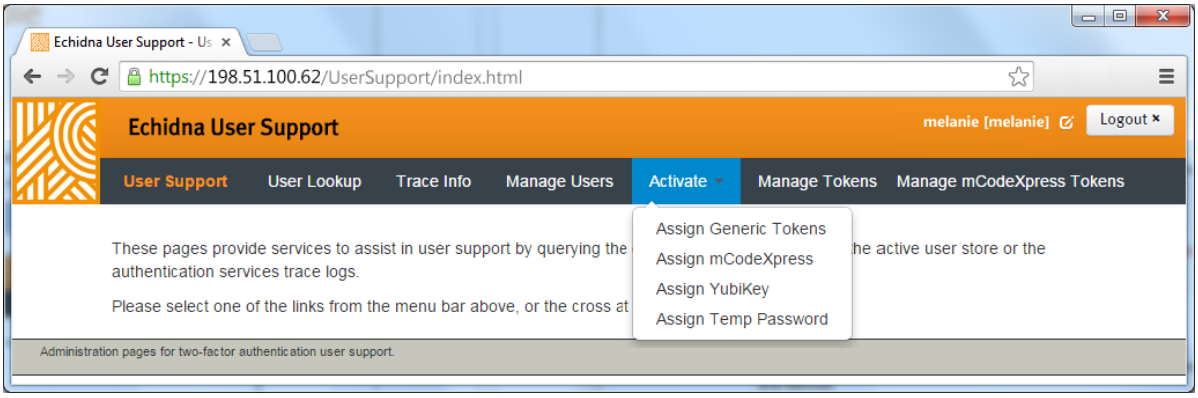


*Figure 6: The **Activate** menu, showing four token types*

The options in the following sections are available only if they have been set up by the system administrator.

Note: Operators cannot assign **CAPTCHA** and **Password Only** tokens to users. **Password Only** is used where a one-factor authentication process is sufficient. **CAPTCHA** is used where configured (typically together with Password Only) only after a set number of failed login attempts.

3.2 Manage Salt mCodeXpress tokens

Salt mCodeXpress is an app that securely generates one-time-passwords (OTPs). Users can install Salt mCodeXpress on their phone or tablet and then register it with their organisation's Echidna server. After registration, Salt mCodeXpress generates OTPs.

This section describes how support staff can use the User Console to do the following tasks:

- Assign a Salt mCodeXpress token to a user (see page 13)
- List all registered Salt mCodeXpress tokens (see page 13)
- Validate a Salt mCodeXpress token (see page 14)
- Synchronise a Salt mCodeXpress token (see page 15)

In addition, support staff can use the following instructions to manage Salt mCodeXpress tokens:

- Check the tokens that are registered to a user (see page 33)

3.2.1 Assign a Salt mCodeXpress token to a user

1. Reset the Salt mCodeXpress app to reveal the registration information. To do this, ask the user to follow the steps in Activate the Salt mCodeXpress token on page 41.
2. Log in to the **User Support** console.
3. Click **Activate**, then click **Assign mCodeXpress**.
4. Search for a user.
5. Enter the registration information that is currently displayed on the device. Ensure that the Time ID is used promptly. An old Time ID can cause the app to produce OTPs that will not be validated by Echidna.
6. Click **Register**.

The Salt mCodeXpress token is now assigned to the user.

3.2.2 List all registered Salt mCodeXpress tokens

1. Log in to the User Support console.
2. Click **Manage Tokens**, then click **Manage mCodeXpress Tokens**.

   A list of all mCodeXpress tokens appears, showing the following information:

   - **Token/User ID**: The primary identifier of the token. By default, this is the same as the assigned user ID, but it can be a generated serial number if the administrator has chosen to configure it that way. Click this link to manage the token.
   - **Key Reg**: This is the Key ID shown during registration and in the About screen in the Salt mCodeXPress app. The Key ID can be used to confirm that the correct mCodeXpress registration from the device has been recorded with the current Echidna server.
   - **Status**: Possible values:
     - **Enabled**: Token can be used.
     - **Failed Threshold**: The token has been suspended after too many failed OTP validation attempts. It can be unsuspended by an administrator.
     - **Admin Suspended**: The token has been suspended and cannot be used. It can be unsuspended by an administrator. After it is unsuspended, the token is still associated with the same user.
     - **Revoked**: This token cannot be used, and cannot be registered to another user.
   - **Clock Offset**: The difference between the times set on the device and on the Echidna server. This is measured in 30-second time blocks. For example, a clock offset of 3 means that the times differ by about 90 seconds.
   - **Created**: The time and date that the app was registered with Echidna.
   - **Last Success**: The most recent time that the token was used to successfully log in.
   - **Last Failure**: The most recent failed attempt to use the token to log in.
   - **Failure Count**: The number of failed attempts when validating an OTP.
   - **Assigned User**: The user to whom this token is assigned. Click this link to see the user details.

3.2.3 Validate a Salt mCodeXpress token

To check that a token is correctly registered, validate its OTP.

This can be useful to check that the Salt mCodeXpress app really was registered with the organization's Echidna server. It can also be useful to check that Echidna and the device are not out of synch.

1. On the mobile device, follow these steps to generate an OTP:
   a. Open the **Salt mCodeXpress** app.
   b. Enter the **PIN** and touch **Generate**.
      A new OTP appears.
2. In the User Console, List all registered Salt mCodeXpress tokens (see page 13).
3. Click on the token ID in the left column, or enter the token ID in the search box and click **Search**.
4. Enter the OTP in the **One Time Password** box, then click **Validate OTP**.
5. Look for the result near the top of the screen.

3.2.4 Synchronise a Salt mCodeXpress token

One-time passcodes (OTPs) generated by Salt mCodeXpress mobile tokens are valid for a certain period of time. The clocks on the Echidna server and on the mobile device must remain synchronised.

If the time from the clock on the Salt mCodeXpress token drifts significantly from the clock on the Echidna server, the OTP values may not be able to be validated.

This problem is not common, because most mobile devices have their clocks synchronised from a network time source, and the server will be synchronised using an NTP source. However, if users manually adjust the device's time instead of adjusting to the default timezone, problems may occur. This is more likely at the start and end of daylight savings, or when the user crosses time zones.

If the generated OTPs suddenly stop working, check the relative Time IDs of Echidna and the device. If the clock offset is outside a range of about -10 to +10, it is worth checking the time settings on the handset and the server.

To fix this problem, resynchronise the token with the server.

1. On the device, open the **Salt mCodeXpress app**.
2. Open the menu, then tap **About**. The About screen shows the **Time ID** for this Salt mCodeXpress token.
3. Log in to the **User Support** console.
4. Select **Manage mCodeXpress Tokens**. A table of token records appears.
5. Click on a **Token/User ID** link in the left column of the table, or enter the domain/userID in the search field at the top of the page.

   The token record appears, with buttons for **Validate OTP, Resynch Clock, Suspend, Revoke**, and **Delete**.
6. Enter the **Time ID** from the device into the **Time ID** box on the User Console page.
7. Click **Resync Clock**.
8. If the synchronisation is successful, a message similar to the following appears at the top of the page:

   success `Resynchronized mcodex token ad-users/melanie`

If this process does not resolve the problem, reset and re-register the user's mCodeXpress token.

3.3 Manage OATH tokens

The User Support console uses the **Generic Tokens** menu to manage the following types of tokens:

- OATH HOTP tokens
- OATH TOTP tokens
- OATH OCRA tokens

3.3.1 How OATH tokens work with Echidna

OATH tokens are software or hardware devices that conform to the OATH standards. OATH tokens provide OTPs to be used for authentication. Each OATH token contains a cryptographic algorithm that lets it generate new OTPs whenever the user needs one.

When a hardware token manufacturer supplies OATH tokens, it also supplies the corresponding cryptographic algorithms and key material (seeds) for the hardware tokens. These seeds must be imported into Echidna.

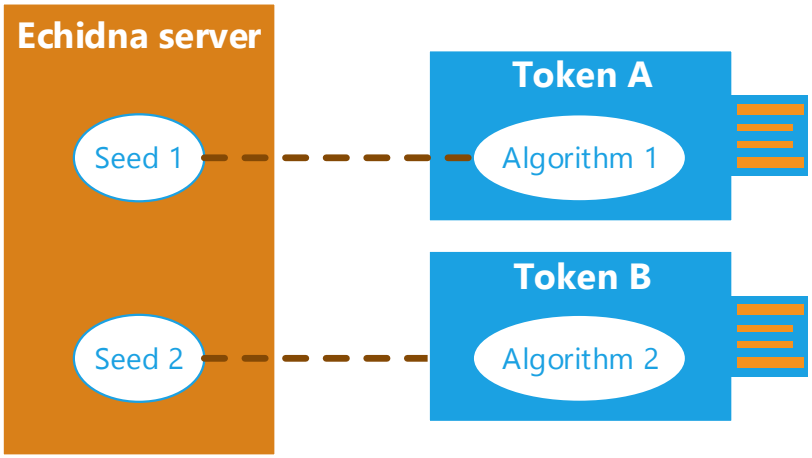


Figure 7: Echidna uses the token seeds to validate OTPs produced by OATH hardware tokens

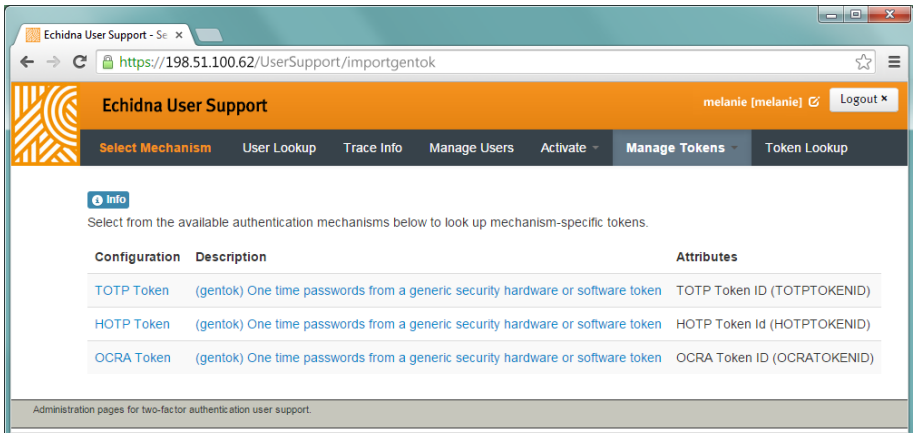
The seed material must be protected while it is being transferred from the token vendor to Echidna. If the seeds are exposed, the tokens are not secure. Some vendors use a proprietary method to protect their seeds, and some use the method defined in the OATH standards. Most methods format the seed files into a file that is protected with a password or secret key.

3.3.2 Import OATH seed files into Echidna

Echidna uses the seed files to create the records for each OTP for each OATH token. These records are used to calculate the correct OTPs.

1. Log in to the **User Support** console, then click **Import Generic Tokens**.

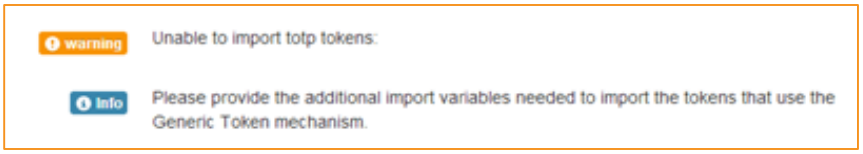
   If more than one type of OATH token is available, they are all listed:



2. Click on the name of the token type.
3. Click **Choose File** and navigate to the seed file.
4. Select a **Key Translator** and a **Key Verifier**.

   The key translator is needed if the import file has encrypted seeds (recommended for production systems). The key verifier is optional but recommended.

5. Click **Load**. Echidna checks the format of the PSKC file.
6. If extra information is needed, Echidna asks for it. In the example below, a password is required:



7. For test tokens using a PBE-AES key, type the plaintext password in the field.

   For production tokens, paste the contents of the accompanying transport key .asc file, which looks similar to the following:

```
-----BEGIN PGP MESSAGE-----
Version: GnuPG v2.0.22 (MingW32)
hIwDdMxBms6QDTcBA/9+BlY+FNyD70LBegjOLUzjAd4labG5605w4teajHGXy2n7
TNjkWZIL9cYY8xnkJUeYsYIDrNehZcwz0HJQKiNrqulHlkGdvycomRqNSziziv+2
g2YmWhBZEwTB6UgCxpbFyrpKnM5yEQWUkBt5nO75gQ==
=3DiF
-----END PGP MESSAGE-----
```

8. Echidna checks that the data is valid and at least one token can be imported.

   If the import is successful, a summary of the result appears with links to manage the first few imported tokens.

3.3.3 Assign an OATH token to a user

Use this option to activate all tokens that support the OATH HOTP, TOTP, or OCRA standards, except for YubiKey tokens.

If more than one type of OATH token is set up, they are all listed under **Generic Tokens**.

1. Log in to the **User Support** console.
2. Click **Activate**, then click **Assign Generic Tokens**.
3. If necessary, choose from the list of OATH token types.
4. Search for the user to assign the token to.

   If this user already has a token of this type, a note appears with details of the currently linked token and noting that assigning a new token will clear this link.
5. Do one of the following:
   - Click **Continue** to apply the new token.
   - Enter the serial number of the token to be assigned and click **Search**.

     If a record for the new token serial number is found, it is displayed.
6. Click **Assign** to assign this token to the user.

3.3.4 List all registered OATH tokens

1. Log in to the User Support console.
2. Click **Manage Tokens**, then click **Manage Generic Tokens**.
3. If more than one type of OATH token is registered, a list of the types appears. Click on the token type.

   A list of all of the recorded OATH tokens appears, showing the following information:

   - **Token/User ID**: The primary identifier of the token. This is usually the physical token serial number, sometimes with a model number in front. Click this link to manage the token.
   - **Key Reg**: If present, this is a short key-check-value calculated on the token seed material.
   - **Status**: Possible values:
     - **Enabled**: Token can be used.
     - **Failed Threshold**: The token has been suspended after too many failed OTP validation attempts. It can be unsuspended by an administrator.
     - **Admin Suspended**: The token has been suspended and cannot be used. It can be unsuspended by an administrator. After it is unsuspended, the token is associated with the same user.
     - **Revoked**: This token cannot be used, and cannot be registered to another user.
   - **Clock Offset**: For time-based tokens, this is the estimated number of time blocks that the server time differs from the hardware token time.
   - **Created**: The time and date that the token was imported into Echidna.
   - **Last Success**: The most recent time that the token was used to successfully validate an OTP or signature.
   - **Last Failure**: The most recent failed attempt to use the token to validate an OTP or signature.
   - **Failure Count**: The number of failed attempts when validating an OTP.
   - **Assigned User**: The user to whom this token is assigned. Click this link to see the user details.

3.3.5 Validate an OATH token

To check that a token is correctly registered, validate its OTP.

This can be useful to check that the token really was registered with the organization's Echidna server. It can also be useful to check that Echidna and the token are not out of synch.

1. Use the token to generate an OTP as usual.
2. In the User Console, List all registered OATH tokens (see page 19).
3. Click on the token ID in the left column, or enter the token ID in the search box and click **Search**.
4. Enter the OTP in the **First OTP** box, then click **Validate OTP**.
5. Look for the result near the top of the screen.

3.3.6 Resynchronise an OATH token

The time from the clock on an OATH TOTP token is used to calculate the TOTP values. If this time drifts significantly from the clock on the Echidna server, the OTP values may not be able to be validated.

OATH HOTP tokens use an event counter instead, which is incremented for each OTP generated. If multiple OTPs are generated but not submitted to the server, the counter may become too far out-of-synch with the server record, and the OTP values will not be able to be validated.

The clock or event counter difference between the token and the server can be re-calibrated by following these steps.

1. Log in to the **User Support** console.
2. Navigate to the token.

   The token management screen shows two input fields (First OTP and Second OTP).
3. Generate an OTP from the token, then enter the new OTP in the **First OTP** box.
4. Generate another OTP from the token, then enter the second new OTP in the **Second OTP** box.
5. Click **Resynchronize**.

   If the synchronisation is successful, a message similar to the following appears at the top of the page:

   success `Resynchronized HOTP Token token HOTP8331581884`

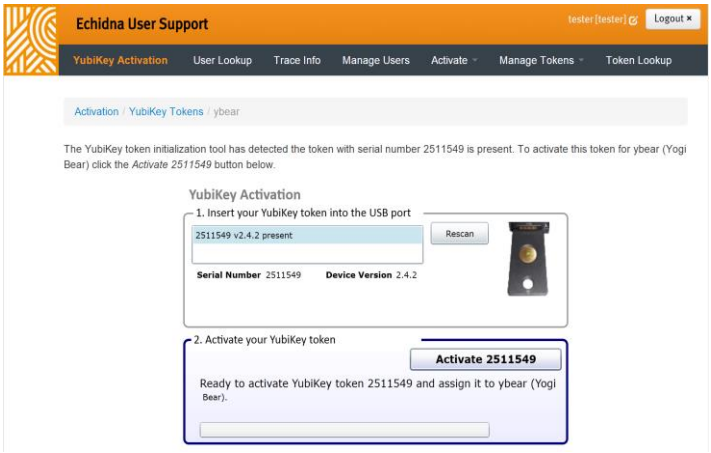
If this process does not resolve the problem, the token might need to be replaced.

3.4 Manage YubiKey tokens

YubiKey tokens comply with the OATH HOTP standard. However, they are set up differently from most other OATH HOTP tokens. For this reason, Echidna treats them differently.

3.4.1 Assign a YubiKey token to a user

1. Log in to the **User Support** console.
2. Click **Activate**, then click **Assign YubiKey**.
3. Enter a user's details, then click **Search**.
   The **YubiKey Activation** section appears.
4. This section uses a Silverlight plugin, which requires permissions to run. If these permissions have already been set up, skip to Step 5. Otherwise, follow these steps:
   a. Run the tool.
   b. Add all new permissions.
   c. Return to the User Support page. The page appears similar to the following:



5. Insert the YubiKey token into a USB port. The token is scanned, and its details appear in the User Support console.
6. Click the **Activate** button to assign the YubiKey token to the user.

3.4.2 Check the status of a user's YubiKey token

1. In the **User Support** console, click **Activate**, then click **Assign YubiKey**.
2. Enter the details of a user who already has a YubiKey assigned, then click **Search**.

3.4.3 Unassign a YubiKey token

1. Search for the user who currently has this YubiKey token.
2. In the **2FA Mechanism** tab, scroll down to the **YubiKey Token** row.
3. In the YubiKey Token row, click the **Clear** button  .

3.5 Manage SMS OTP tokens

3.5.1 Assign an SMS OTP token to a user

1. Log in to the **User Support** console.
2. Click **Activate**, then click **Assign SMS OTP Device**.
3. Enter a user's details, then click **Search**.
   The **Registered SMS OTP** section appears.
4. Enter the device ID.

3.5.2 Generate an SMS OTP

1. Log in to the User Support console.
2. Click **Manage Tokens**, then click **Manage SMS Tokens**.
   A list of all of the recorded SMS tokens appears.
3. Click the **Device ID** of one of the tokens.
4. Click **Generate Challenge**.

An SMS message containing an OTP is sent to the user.

3.6 Manage temporary passwords

1. Log in to the **User Support** console.
2. Click **Activate**, then click **Assign Temp Password**.
3. Search for the user.
4. Click **Register** to assign a new temporary password to this user. If the user already had a temporary password, it is overwritten by the new password.
5. If this user already has a temporary password, click **Deregister** to remove it.

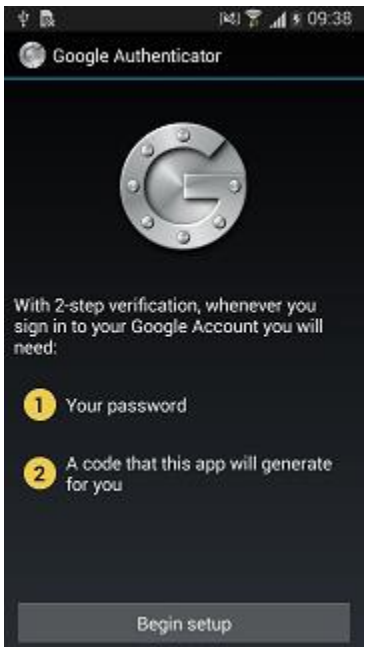
Note: To check how many times the password has been used, return to the **Assign Temp Password** page.

3.7 Manage Google Authenticator tokens

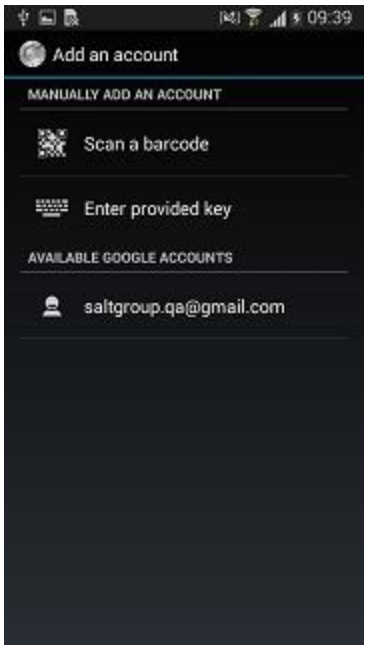
Google Authenticator tokens comply with the OATH TOTP/HOTP standards. Registration is done differently, but management of the tokens after registration is the same as the generic OATH TOTP/HOTP tokens.

3.7.1 Assign a Google Authenticator token to a user

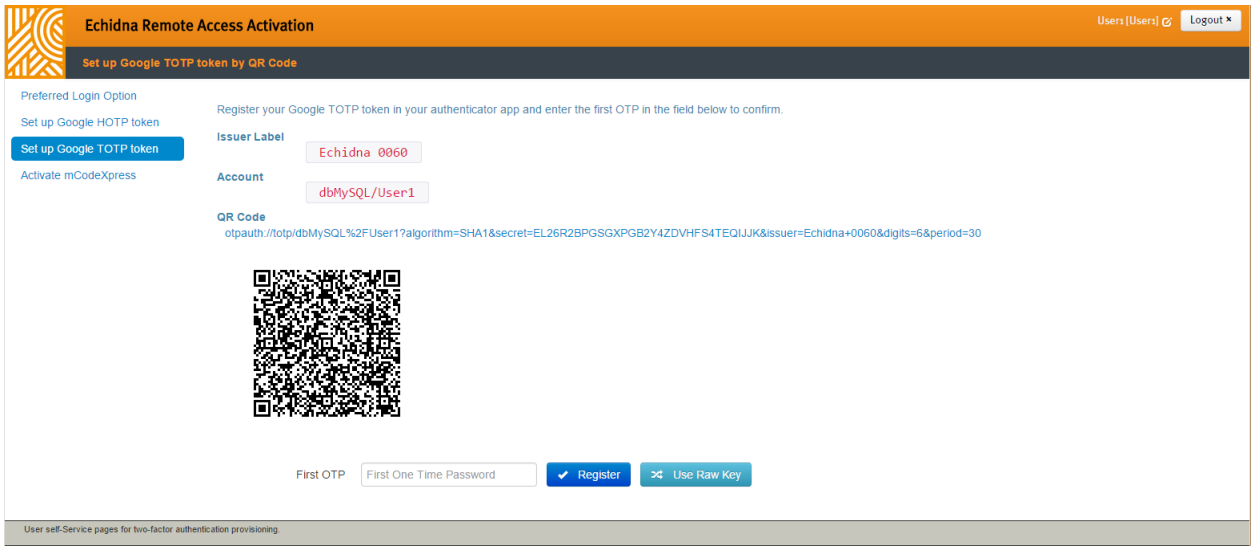
1. Ensure that the device has network access.
2. Open the Google Authenticator app and touch **Begin Setup**:



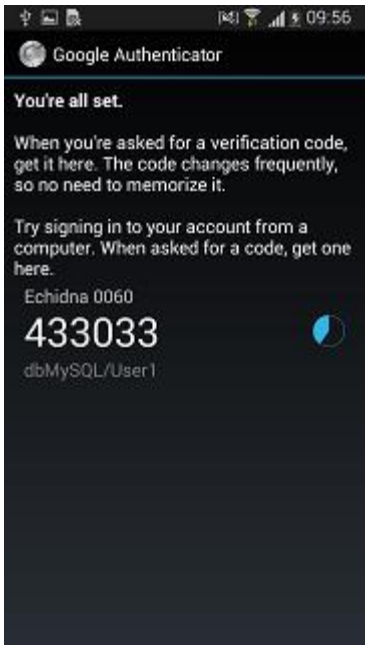
3. An account can be manually added by scanning a QR code or entering the key manually. Touch **Scan a barcode**:



4. Login to the **Self Service** console.
5. To register a Google TOTP token, select **Set up Google TOTP token**. For HOTP token, select **Setup Google HOTP token**.
6. Scan the QR code with the device:

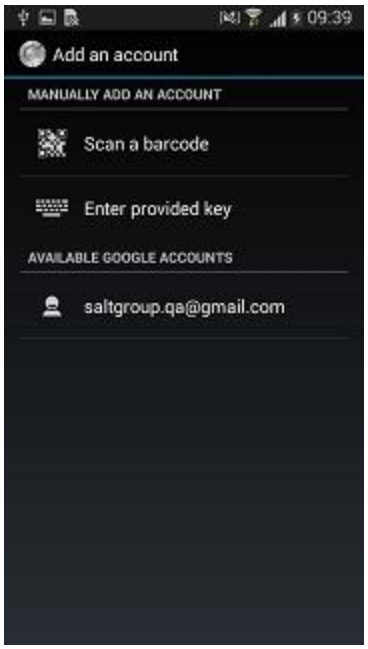


7. Here's the screen after scanning a QR code for a Google TOTP token:

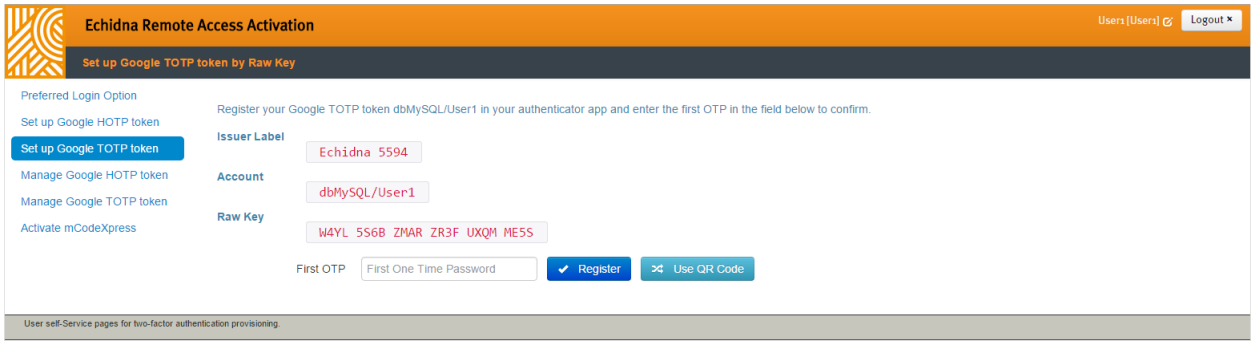


8. Here's the screen after scanning a QR code for a Google HOTP token:

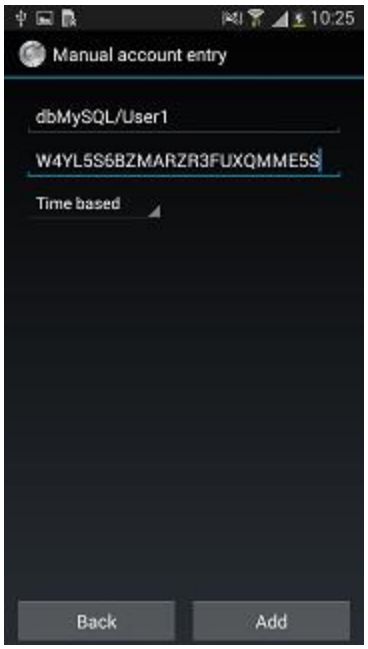
9. Alternatively, a Google token can be registered by entering the raw key manually. On the device, touch **Enter provided key**.



10. In the Self Service console, click on **Use Raw Key**.

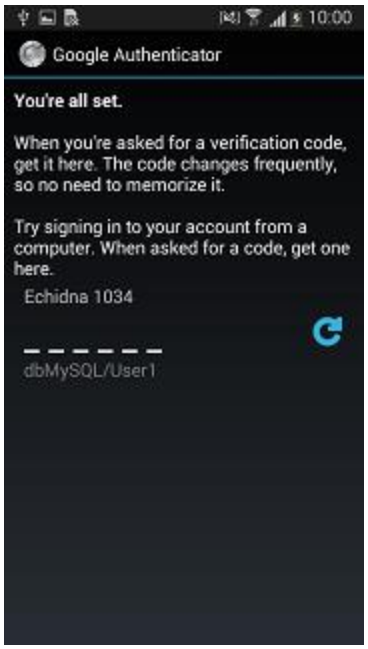


11. Enter the **Account** name and **Raw Key** from the Self Service console to the device. Select **Time based** for TOTP token or **Counter based** for HOTP token. Touch **Add**.



12. Here's the screen after manual raw key entry for a TOTP token:

13. Here's the screen after manual raw key entry for a HOTP token:



14. Enter the generated one time password to the Self Service console's **First OTP** field and click **Register**.

3.7.2 List all registered Google Authenticator tokens

1. Log in to the User Support console.
2. Click **Manage Tokens**, then click **Manage Generic Tokens**.
3. If more than one type of OATH token is registered, a list of the types appears. Click on the token type.

   A list of all of the recorded OATH tokens appears, showing the following information:

   - **Token/User ID**: The primary identifier of the token. This is usually the physical token serial number, sometimes with a model number in front. Click this link to manage the token.
   - **Key Reg**: If present, this is a short key-check-value calculated on the token seed material.
   - **Status**: Possible values:
     - **Enabled**: Token can be used.
     - **Failed Threshold**: The token has been suspended after too many failed OTP validation attempts. It can be unsuspended by an administrator.
     - **Admin Suspended**: The token has been suspended and cannot be used. It can be unsuspended by an administrator. After it is unsuspended, the token is associated with the same user.
     - **Revoked**: This token cannot be used, and cannot be registered to another user.
   - **Clock Offset**: For time-based tokens, this is the estimated number of time blocks that the server time differs from the hardware token time.
   - **Created**: The time and date that the token was imported into Echidna.
   - **Last Success**: The most recent time that the token was used to successfully validate an OTP or signature.

- **Last Failure**: The most recent failed attempt to use the token to validate an OTP or signature.
- **Failure Count**: The number of failed attempts when validating an OTP.
- **Assigned User**: The user to whom this token is assigned. Click this link to see the user details.

3.7.3 Validate a Google Authenticator token

To check that a token is correctly registered, validate its OTP.

This can be useful to check that the token really was registered with the organization's Echidna server. It can also be useful to check that Echidna and the token are not out of synch.

1. Use the Google Authenticator app to generate an OTP as usual.
2. In the User Console, List all registered Google Authenticator tokens (see page 28).
3. Click on the token ID in the left column, or enter the token ID in the search box and click **Search**.
4. Enter the OTP in the **First OTP** box, then click **Validate OTP**.
5. Look for the result near the top of the screen.

3.7.4 Resynchronise a Google Authenticator token

The clock or event counter difference between the token and the server can be re-calibrated by following these steps.

1. Log in to the **User Support** console.
2. Navigate to the token.

   The token management screen shows two input fields (First OTP and Second OTP).
3. Generate an OTP from the Google Authenticator app, then enter the new OTP in the **First OTP** box.
4. Generate another OTP, then enter the second new OTP in the **Second OTP** box.
5. Click **Resynchronize**.

   If the synchronisation is successful, a message similar to the following appears at the top of the page:

   success `Resynchronized Google HOTP Token token dbMySQL\User1`

If this process does not resolve the problem, the token might need to be replaced.

3.8 Suspend, revoke, or delete a token

When a token should not be used, the operator has the following options:

- **Suspend**: Prevent the token being used. When a suspended token is later unsuspended, it can be used again. Use **Suspend** when transferring a token from one user to another.

  In Echidna, the administrator can configure an authentication method to automatically suspend tokens after some number of failed validation attempts.

- **Revoke**: Suspend a token permanently, and leave the token's record in the database. If a token has been lost or damaged, revoke it to prevent it being used again.
- **Delete**: Remove the token entirely, including its database record. This option is available only to users with the **token-admin** role. **Delete** should be used only when persistent records are not required to be kept for audit or tracing purposes.

  Note: The **token-admin** role is granted by default to members of the GRANT 2FA-Token Administration group.

The following steps describe how to suspend, revoke or delete a token:

1. Log in to the **User Support** console.
2. Use a **Manage** menu item to navigate to the page that lists the tokens. For example, to change a Salt mCodeXpress token, click **Manage mCodeXpress**.
3. Click on the token ID in the left column, or enter the token ID in the search box and click **Search**.
4. Click **Suspend**, **Revoke**, or **Delete**. The change occurs immediately.

In addition, operators can Clear a token from a user (see page 34).

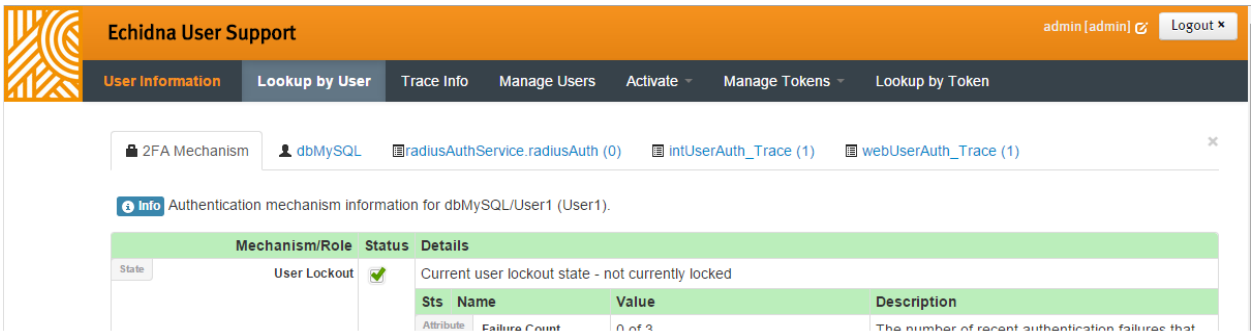
Chapter 4: Managing assignment of tokens to users

This chapter contains these sections:

4.1 Find a user's record

1. Click **Lookup by User**, then enter a username.
2. Enter a domain name. This is required only if more than one user store is set up. If no domain is entered here, the User Support console uses the default domain that the administrator specified.
3. Click **Search**. The user details appear in five tabs.

The following screenshot shows the tabs with example names. These tab names are different if an administrator has changed their names:



Tab name	Description
2FA Mechanism	Lists the user details stored in Echidna.
Name of a user store	Lists the attributes in the user record.
radiusAuthService.radiusAuth	Shows the user's trace records for the RADIUS service.
webServices.intUserAuth	Shows the user's trace records for the web service used by this User Support console. This tab lists the outcome of each attempt to log in to the User Support console.
webServices.webUserAuth	Shows the user's trace records for the web service used by clients.

4.2 Check the tokens that are registered to a user

1. Find a user's record (see page 32).
2. On the **2FA Mechanism** tab, look in the **Status** column.
   The available authentication methods have a green check mark.

For example, the following screenshot shows that this user can use CAPTCHA, and has mCodeXpress and SMS OTP tokens. However this user has no OATH tokens (TOTP, HOTP, and OCRA).

To skip to the **Activate** page for a token, click its **Activate** button .

4.3 Clear a token from a user

When the operator clears a token from a user, the user no longer has a connection with that token. The user cannot use the token.

1. Check the tokens that are registered to a user (see page 33).
2. Click the **Clear** button  for a token. The token is immediately disassociated from the user, plus any other actions, described below.

Each type of token behaves differently after being cleared:

- **OATH tokens**: Clearing an OATH token only removes its association with the current user. The token is not otherwise changed. It is now ready to be assigned to a new user.
- **mCodeXpress tokens**: Clearing an mCodeXpress token also deletes the token. This is because the token is of no use once it is no longer associated with the current user. It cannot be assigned to another user.
- **Temporary passwords**: Clearing a temporary password also deletes the token, because it cannot be assigned to another user.
- **SMS OTP tokens**: This works only if the administrator has allowed this mobile number to be edited in the User Console. If it is editable, clearing the token means setting the mobile number to empty in the user store.

  This may be appropriate if the user store or the identified attribute is dedicated to use with SMS authentication. If the attribute is also used by other applications, it should not be editable from Echidna.

For details about other ways to stop a token being used, see Suspend, revoke, or delete a token on page 29.

4.4 Check a user's status

1. Find a user's record (see page 32).
2. On the **2FA Mechanism** tab, look for the **User Lockout** section.

   This section lists the number of recent authentication failures, and whether the user has been temporarily locked out due to too many failures.

   The user store itself (Active Directory or LDAP) may also lock the user out if the wrong store password has been used too many times. Manage this situation with the usual user management tools.

   For Active Directory, the user attributes shown on the second tab can include **badPasswordTime**, **badPwdCount**, **pwdLastSet** and **userAccountControl** with the UF_LOCKOUT bit set.

4.5 Check the trace for a user's access requests

If a user has trouble logging in with their token, check their access requests. If a request failed, the error message can reveal the reason.

See Appendix: Authentication result codes on page 44 for a list of all possible messages.

To check the access requests for **a single user**:

1. Find a user's record (see page 32).
2. Click on a tab that corresponds with a service that is protected by Echidna. These tab names have been defined by the administrator at the organisation.

   The tab shows a list of the user's access attempts for that service.

To check the access requests from **all users for a service**:

1. Click **Trace Info**. The services protected by Echidna are listed.
2. Click one of the service names to show a list of all access requests for that service.

Chapter 5: Managing users

This chapter describes how to manage users in the User Support console.

These steps are recommended only if there are no existing tools or processes to manage the users. If the organisation already has user management tools, continue to use them, and skip this chapter.

This works only if the administrator has configured the User Support console to be able to edit user records. The User Support console cannot create users or groups in an external user store such as Active Directory. Any new users or groups created with this page are stored in the Echidna database.

5.1 Add a user

1. Click **Manage Users**.
2. Under the name of the user store, click the link to the users.
3. Click **New**.
4. Enter the user's details in the list of attributes.

5.2 Edit or delete a user

1. Click **Manage Users**.
2. Under the name of the user store, click the link to the users.
   A list of all editable users in this store appears.
3. Click on the user's name. The user details appear.
4. To update the user details, click **Edit Details**. Make the changes, then click **Update**.
   To delete the user record, click **Delete**.

Chapter 6: Using the Self-Service console

Echidna comes with a web page that lets end users manage their own tokens. The administrator can prevent access to this page if it is not required for an organisation.

Use this chapter to guide users as they use the Self Service console.

The Self Service console shows only the pages that a user can use. For example, if an organisation does not use mCodeXPress, no pages about mCodeXpress appear in its user Self Service console.

6.1 Open the Self Service console

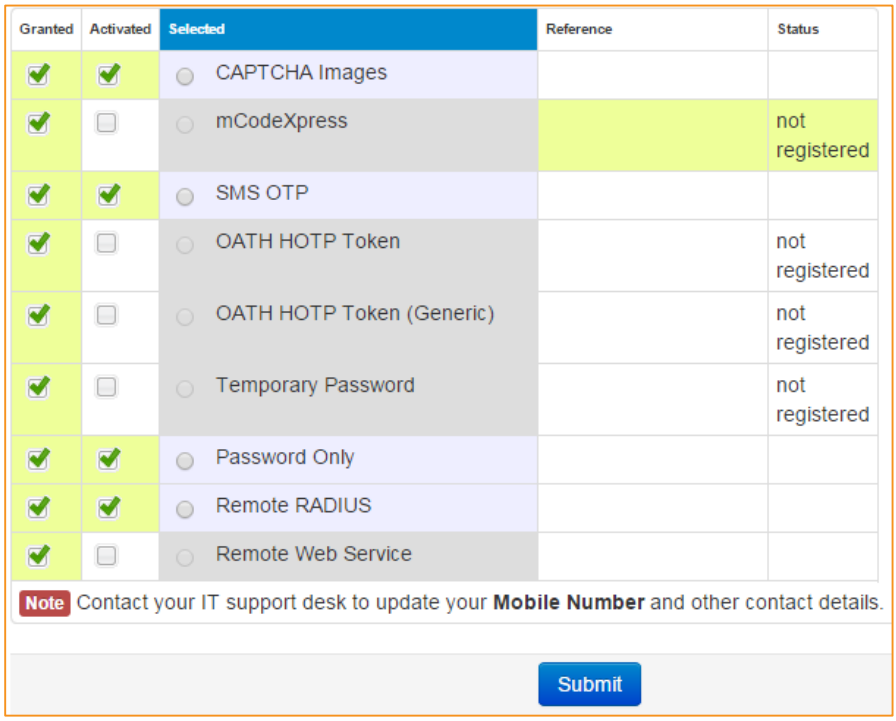
1. Ask the administrator where to find the Self Service console. It can be published on the organization's website, on an intranet, or embedded in another application.
2. Sign in using the usual username and password.

6.2 Change the user's preferred login option

If the administrator has allowed users to select their preferred login methods, the steps below work.

However, if the administrator has **not** allowed users to set their own preferred login methods, the page still appears but the user cannot make changes.

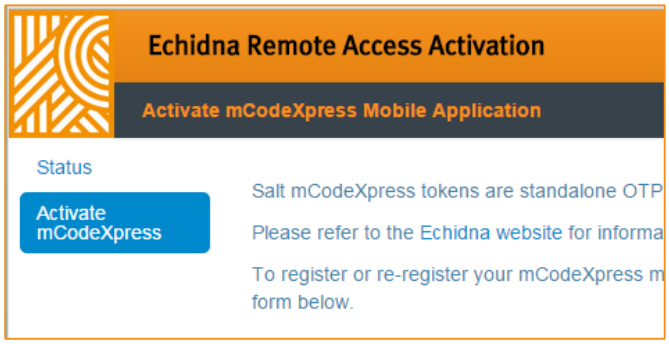
1. Sign in to the Self Service console.
2. Click **Preferred Login Option** on the left.
   The table lists the authentication methods that are available to this user. The ones with a green check box  have been activated.



3. Click on the name of an activated authentication method to make it the preferred method.
   In the example above, the user could click on **CAPTCHA Images**, **SMS OTP**, **Password Only**, or **Remote RADIUS**.
4. Click **Submit**.

6.3 Activate a token

1. Sign in to the Self Service console.
2. Click an **Activate** link on the left.
   In the following example, the user can activate an mCodeXpress token:



For more information about activating particular types of tokens, see these sections:

- Assign an OATH token to a user (see page 18)
- Assign a YubiKey token to a user (see page 21)
- Assign a Google Authenticator token to a user (see page 23)
- Activate the Salt mCodeXpress token (see page 41)

6.4 Test a token

1. Sign in to the Self Service console.
2. Click a **Test** link on the left.
3. Use the token to generate an OTP.
4. Enter the OTP in the Self Service console, and then click **Submit**.
5. Check for information near the top of the page.

Chapter 7: Using the Salt mCodeXpress app

If the organisation uses mCodeXpress to create one-time passcodes, support staff might need to support users who use the Salt mCodeXpress app.

This chapter describes the tasks that users can do with the Salt mCodeXpress app.

7.1 About Salt mCodeXpress

Salt mCodeXpress is an app that securely generates one-time passcodes (OTPs). Users can install Salt mCodeXpress on their mobile device and then register it with their organisation's Echidna server. After registration, the Salt mCodeXpress app generates OTPs that let the user sign in to their organization's VPN or other remote access system.

Salt mCodeXpress requires network access while it is being set up. After that, it does not need network access.

Salt mCodeXpress is available in these app stores:

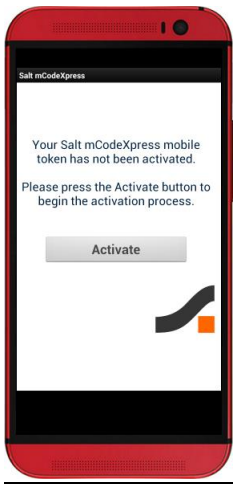
- iTunes App Store
- Google Play
- Windows Phone Store

7.2 Install Salt mCodeXpress

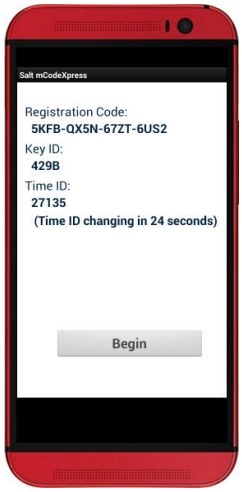
1. On the mobile device, search in the appropriate app store for **Salt mCodeXpress**.
2. Install the app.

7.3 Activate the Salt mCodeXpress token

1. Open the Salt mCodeXpress app and touch **Activate**:



2. Enter a new 6-digit PIN, then tap **OK**. The app shows three new codes, which will be used in the next section:
   - Registration code
   - Key ID
   - Time ID



3. In the web browser, go to the **Self Service** console. Ask the administrator where this page is.
4. Sign in to the **Self Service** console using the usual name and password.
5. Enter the following information that appears on the device:

Code	Comment
Registration code	Include every character, including hyphens (-). This code is not case-sensitive.
Key ID	This code is not case-sensitive.
Time ID	This code changes frequently. If an old code is used, registration will not work.

6. On the **Self Service** console, click **Register Token**.
7. When the registration is complete, tap **Begin** in the app. The three codes from Step 5 will never be displayed again, so make sure that registration was successful.

The user can now use Salt mCodeXpress to create a one-time-passcode (OTP) to access the organisation's systems.

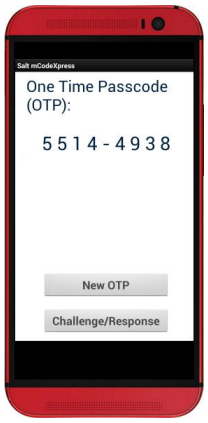
7.4 Get a one-time password from Salt mCodeXpress

To use the organisation's VPN or other remote access service, use Salt mCodeXpress to generate a one-time password (OTP).

The device does not need network access for these steps.

1. Follow these steps on the **device**:
   a. Open **Salt mCodeXpress**:

   b. Enter the PIN, then tap **Generate**.
      The OTP appears on the screen.



2. Follow these steps on the **computer**:
   a. Open VPN or remote access software.
   b. Enter the usual username.
   c. In the password box, enter the OTP exactly as it appears on the device.
3. Click **OK** to start the remote connection.

7.5 Reset the Salt mCodeXpress PIN

If the user has forgotten their Salt mCodeXpress PIN, they can reset the app to create a new PIN. However, this means registering Salt mCodeXpress again.

The PIN can be updated without network access.

1. On the device, open **Salt mCodeXpress**.
2. Open the menu, then tap **About**.

   Note: The Salt mCodeXpress menu is in a different place on each type of device.
3. Open the menu again, then tap **Reset**, then tap **OK** at the messages.

   The Salt mCodeXpress app is now unregistered. To register the Salt mCodeXpress token again, follow the steps in Activate the Salt mCodeXpress token on page 41.

Appendix: Authentication result codes

This appendix lists the messages that appear in traces, as described in Check the trace for a user's access requests on page 35.

Name	Type	Description
CHALLENGE_ALTAUTH_RES	CHALLENGE	The initially selected authentication mechanism for the user is not available (due to a service or token-unavailable response), but a fall-back has been configured. The user is being challenged to enter the credentials appropriate to the fall-back mechanism.
CRED_VALIDATION_FAILED	REJECT	Validation of the user credential failed due to some kind of service error -- the validity or otherwise of the credential cannot be established.
HEARTBEAT_USER_RES	CHALLENGE	The result recorded when an access-request is received for a heart-beat monitoring user.
IGNORE_BADMSGAUTH	DISCARD	The validation of the message-authenticator in the access-request failed. The packet will be ignored.
IGNORE_DUPLICATE	DISCARD	The received RADIUS request packet was a duplicate of a previously seen packet that is already being processed (potentially due to a timeout-retry mechanism at the client). This packet will not be processed.
IGNORE_INVALID_CLIENT	DISCARD	The RADIUS access-request appears to originate from a client address that has not been configured as a RADIUS client. The packet will be ignored.
IGNORE_MALFORMED	DISCARD	The received RADIUS request packet was malformed in some way and will be ignored in conformance with the RADIUS RFC 2865 rules.
IGNORE_PACKET_TYPE	DISCARD	The RADIUS request packet was not of the type (Access-Request) supported by this listener. The packet will be ignored.

Name	Type	Description
INVALID_AUTHSTATE	REJECT	The RADIUS state attribute included in the access-request was not a valid state that was created by this RADIUS server.
INVALID_CONFIG	REJECT	Some part of the configuration at the server appears to be invalid or incomplete and has caused a failure in processing the request.
INVALID_CREDS	REJECT	The access-request provided invalid user credentials (password).
INVALID_REQUEST	REJECT	The request was invalid due to a problem decoding the password attribute. The client's RADIUS shared-secret may be incorrect.
INVALID_USER	REJECT	The access-request specified an invalid user name.
LOCKEDOUT_USER	REJECT	The timed user-lockout mechanism has been triggered for the user name specified in the access request. The user will have to wait for the lockout to expire before authentication can be re-attempted.
NOT_ENABLED_RES	REJECT	The registered OTP token for the user has been disabled or locked.
NOT_REGISTERED_RES	REJECT	There is no registered OTP token associated with the user.
NO_AUTHMECH_SELECTED	REJECT	No authentication mechanism was selected, since for all the configured mechanisms the pre-conditions for application to the specific user were not met.
OATH_BAD_FORMAT_RES	REJECT	The submitted OTP had an invalid length or format, or it had a KCV prefix, and the KCV didn't match the recorded key id.
OATH_OTP_CHAL_RES	CHALLENGE	The user-store password has been collected, the user should now be prompted for the next OATH one-time-password.
OATH_WRONG_OTP_RES	REJECT	The submitted OATH one-time-password was incorrect or possibly outside the time window.
OTP_CHAL_RES	CHALLENGE	The user-store password has been collected, the user should now be prompted for the one-time-password.
PASSWORD_NOT_SET	REJECT	No password has been set for the user in the user store, so the access-request password cannot be validated.

Name	Type	Description
REJECT_MALFORMED	REJECT	The access-request packet was malformed in some way that does not require it to be ignored. An explicit access-reject will be returned.
RR_ACCEPT_RES	ACCEPT	The remote RADIUS server has returned an access-accept, and the local authentication process has completed successfully.
RR_CHALLENGE_LOCAL_RES	CHALLENGE	The user-store password has been collected and now the user should be challenged for the remote RADIUS server password. No remote RADIUS call has been made yet.
RR_CHALLENGE_NOMSG_RES	CHALLENGE	The remote RADIUS server returned an access-challenge but with no specific reply-message. The challenge with a generic reply-message should be passed through back to the caller.
RR_CHALLENGE_RES	CHALLENGE	The remote RADIUS server returned an access-challenge including a reply-message which should be passed through back to the caller.
RR_ERROR_RES	REJECT	There was a problem invoking the remote RADIUS server, or the user has not been properly configured for it (such as missing a remote token-identifier).
RR_REJECT_RES	REJECT	The remote RADIUS server has returned an access-reject.
RR_TIMEOUT_RES	REJECT	The timeout has expired waiting for the remote RADIUS server to respond.
SERVICE_ERROR_RES	REJECT	The remote authentication service is failing with some specific error message.
SERVICE_UNAVAILABLE_RES	REJECT	The remote authentication service (RADIUS or Web Service) is not responding as expected, making the service unavailable.
SMS_ALREADY_RES	CHALLENGE	Similar to SMS_CHAL_RES except that the relevant SMS message was already sent on a previous call (the one that returned SMS_CHAL_RES), and the OTP has not yet been used or expired -- so a new OTP has not been generated.
SMS_CHAL_RES	CHALLENGE	The OTP was successfully generated and sent via the configured SMS gateway. The user should be prompted now for the OTP using the specified reply-message text.

Name	Type	Description
SMS_EXPIRED_RES	REJECT	Returned when validating a submitted OTP when the previously generated OTP has either expired or already been used once. Repeat the access request to generate a fresh one.
SMS_FAILURE_RES	REJECT	Some other failure has occurred within the call, such as a failure to contact the SMS gateway or other service availability problem.
SMS_LOCKED_RES	REJECT	The guess-limit within the time-window for the given phone number has been exceeded, so the user will have to wait for the time window to expire before they can attempt authentication again.
SMS_NOWLOCKED_RES	REJECT	The submitted OTP was incorrect, and the lock-threshold has now been exceeded, so any subsequent calls to generateChallenge for this phone will return SMS_LOCKED_RES until the lockout expires.
SUCCESS	ACCEPT	Generic authentication-success result
TOKEN_UNAVAILABLE_RES	REJECT	The specific token identified for the resolved user does not exist in the remote service or has been disabled.
USER_MISSING_ATTR	REJECT	There is a user attribute missing from the user's context (the user-store entry) that is required for the selected authentication mechanism. For example, the mobile phone number is missing for SMS authentication.
USER_RESOLVE_FAILED	REJECT	Resolving the user in a user store failed due to some kind of service exception, such as the directory or database not being available.
USER_UNAUTHORIZED_RES	REJECT	Some group or role membership is required to allow the user to authenticate, and they do not meet the required condition.
WRONG_CHALRESPONSE_RES	REJECT	The submitted OTP was incorrect, but the lock-threshold has not been exceeded yet, so it is possible for the user to try again immediately.
emailAuth.SMS_ALREADY_RES	CHALLENGE	Same as SMS_ALREADY_RES but with messages formats specific to the emailAuth mechanism.
emailAuth.SMS_CHAL_RES	CHALLENGE	Same as SMS_CHAL_RES but with messages formats specific to the emailAuth mechanism.

Name	Type	Description
smsAuth.SMS_ALREADY_RES	CHALLENGE	Same as SMS_ALREADY_RES but with messages formats specific to the smsAuth mechanism.
smsAuth.SMS_CHAL_RES	CHALLENGE	Same as SMS_CHAL_RES but with messages formats specific to the smsAuth mechanism.