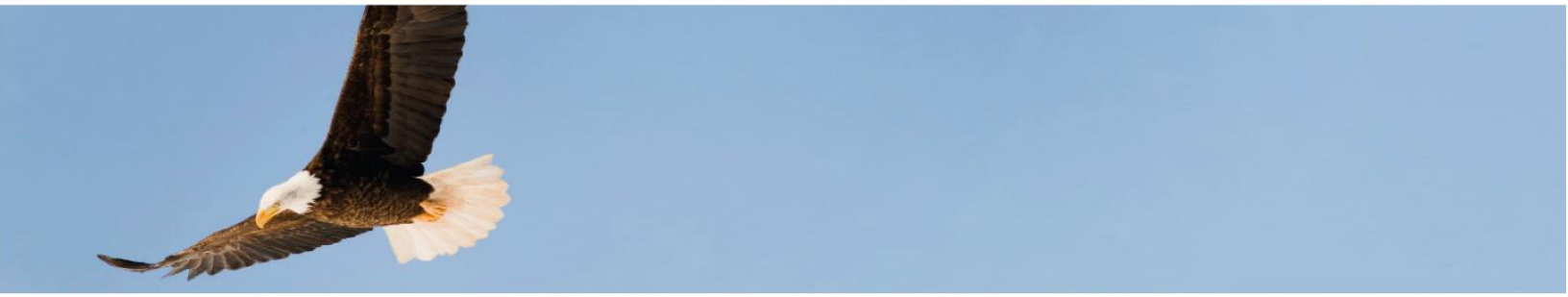
Salt Group



Echidna Installation Guide

Version 15.2

October 2015

Contents

Chapter 1: Before starting

This *Installation Guide* is part of a set of books about Echidna. It describes how to install and configure Echidna. To extend the configuration, use the *Administration Reference*.

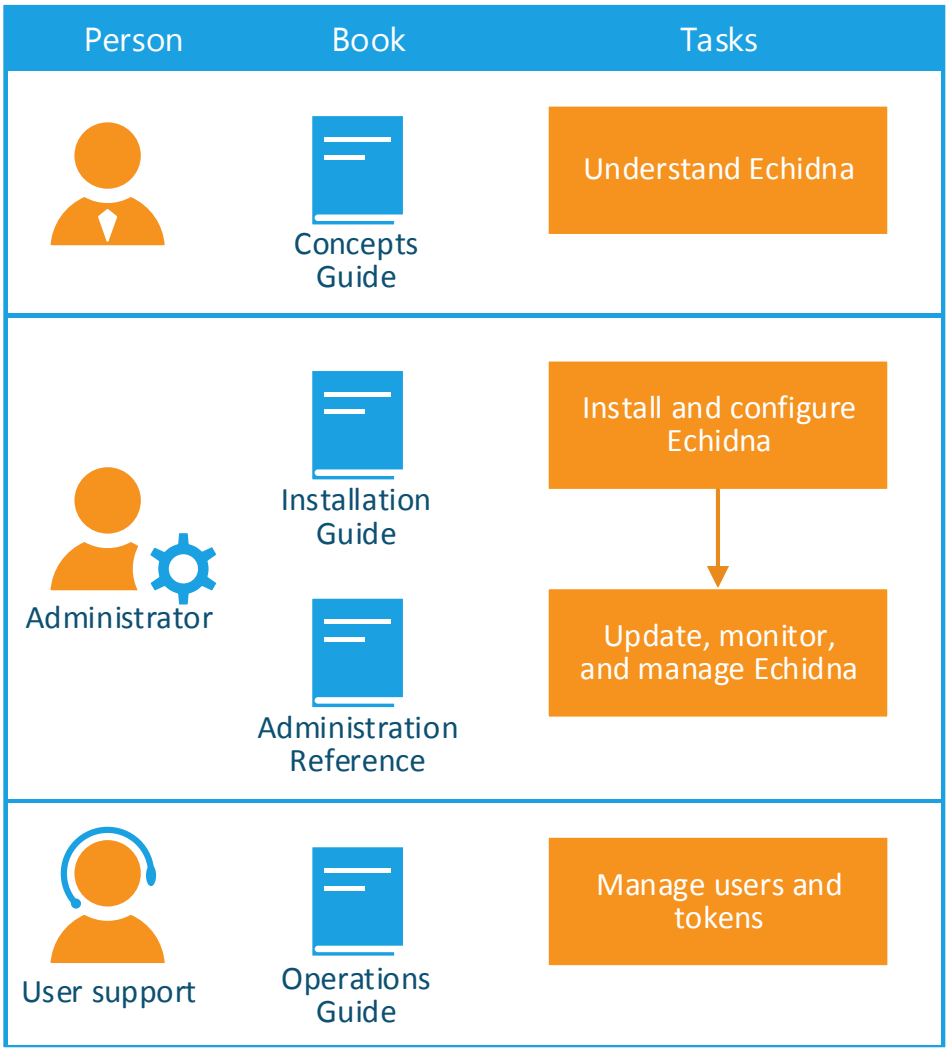


Figure 1: When to use each book in the Echidna documentation set

1.1 About Echidna

Echidna is an enterprise-grade security platform for two-factor authentication (2FA) supporting multi-method authentication:

- Salt mobile tokens
- SMS OTP
- OATH hardware tokens

Echidna is distributed as a virtual appliance. This is a virtual machine (VM) that already has Echidna installed on it, plus the other software that Echidna requires.

After finishing the steps in this book, the administrator will use the *Administration Reference* to expand the Echidna system to work in their organisation.

1.2 Components of Echidna

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

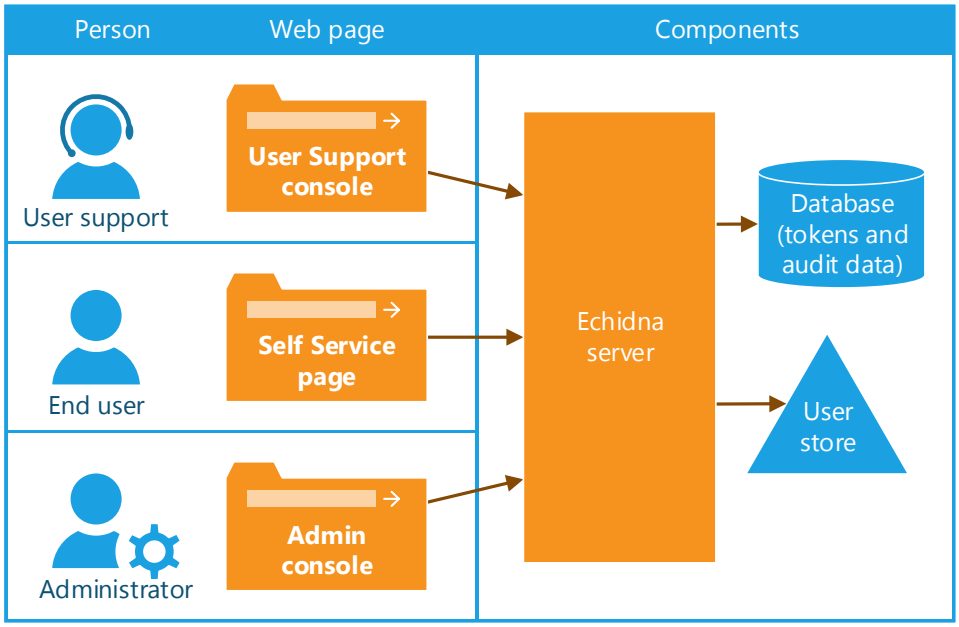


Figure 2: Components of Echidna, and who uses them

Each console lets a particular group of people do relevant tasks:

- 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 describes how to use the Administration console to set up Echidna.

1.3 Software supplied in the Echidna appliance

Echidna is supplied as a virtual appliance, which is a virtual machine that already contains Echidna. Echidna can be supplied as an installation package, upon request. This guide covers only the virtual appliance, not the installation package.

The software supplied with Echidna is open-source, and the organisation does not need to purchase licenses for anything other than Echidna.

Echidna is served by Apache Tomcat. In addition, the virtual machine automatically includes OpenJDK and a selection of JDBC drivers.

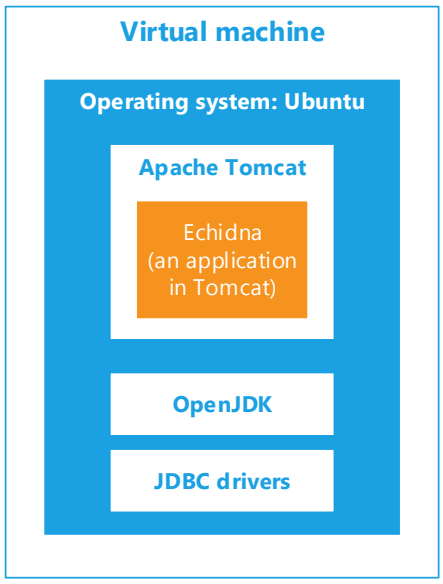


Figure 3: Echidna is an application within Tomcat, which is already set up in a virtual appliance

Salt Group supplies the virtual appliance in Open Visualization Format (OVF) format, which most hypervisors can use. To set up Echidna, the administrator deploys a new VM, using the OVF file as a template. In the new VM, the administrator then configures the new Echidna server, and connects it to other systems in the organisation.

The Ubuntu operating system is configured to restrict access to only the services provided by Echidna, the Virtual Appliance System Management Interface, and SSH.

1.4 Check the system requirements

Ensure that the system meets the following requirements:

System	Purpose of system	Requirements
Virtualisation system	The hypervisor system that runs the virtual appliance that includes Echidna.	One of the following: • VMware ESXi server • VMware Workstation and VMware Server • Any other hypervisor that supports the Open Virtualization Format (OVF)
Client computers	Connect to the following consoles to work with Echidna: • Administration console • User Support console • Self Service console (for end users)	Any of the following Internet browsers: • Internet Explorer 8 and later • Mozilla Firefox 3 and later • Google Chrome (any version) Note: These requirements are for connecting to Echidna itself. Some tokens may support a smaller range of browsers.

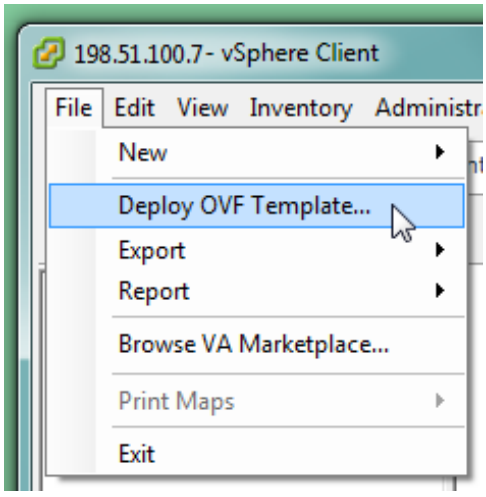
Chapter 2: Deploy the virtual appliance

Echidna is supplied as a virtual appliance, which is a virtual machine (VM) that contains the Echidna software.

Salt Group supplies the virtual appliance in Open Visualization Format (OVF) format, which most hypervisors can use. The instructions below show how to deploy the OVF file using VMware.

2.1 Deploy the Echidna virtual appliance

1. Obtain the virtual appliance file `echidna.ovf`.
2. Uncompress the OVF file and locate the OVA file.
3. In the virtualisation software, deploy a new virtual machine from the OVA file.
   For example, in VMware vSphere Client use **Deploy OVF Template**, as shown here:



4. Follow the steps in the virtualisation software.
   For example, follow these steps in VMware:
   a. Navigate to the OVA file.
   b. Accept the summary of the OVA details.
   c. Accept the End User License Agreement for Salt Group.
   d. Create a name for the new virtual machine.

e. Choose a storage location for the virtual machine files.

f. Specify the disk format, including the provisioning. Use thin provisioning to save space.

g. Map the networks in Echidna to the destination networks available in the organisation.

   - For a small trial system, map all three Echidna networks to the same destination network.
   - For a production system, assign each Echidna network to a separate destination network, if required.

     This table lists the services that use each interface if Echidna is configured with **three network interfaces**:

Source network	**Gives access to Echidna services**
Network 1	Default network for all services if no other is present. If other networks are present, used for administration services: • Administration console • User Support console
Network 2	Client services: • Web services requests • RADIUS requests
Network 3	User services: • Self-Service console

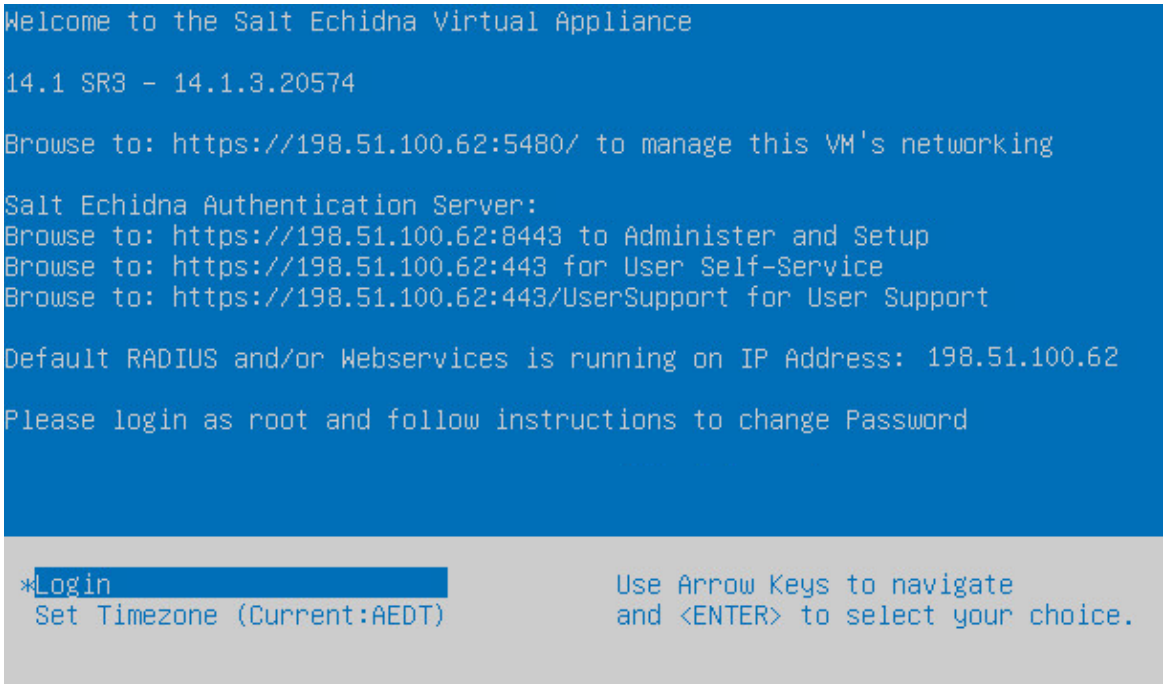
h. Click **Finish** to deploy the new virtual machine.

The new VM has been deployed. The next step is to configure the VM with a new root password and time zone.

2.2 Configure the new virtual machine

After deploying the virtual appliance, change its root password and time zone.

1. Start the virtual appliance.
2. Read the terms and conditions, pressing **Enter** to page through the agreement.
3. Type **yes** to accept the terms and conditions if you agree with them.
   After accepting the license agreement, a blue console page appears. This is the console for the VM.



4. Press **Enter** to choose the **Login** option.
5. Log in with the following credentials:
   - **User name**: root
   - **Password**: changeit
6. Create a new password. The original password cannot be used again.
7. The virtual appliance can be accessed via SSH, however the ssh user “echidna” must be configured with a new password. Type **passwd echidna** and follow the prompts to assign a new password.
8. Type **exit** to return to the console page.
9. Select the **Set Timezone** option, and then follow the prompts to set the time zone.
   Skip this step if Echidna is in the UTC time zone.
10. Note the URLs that are listed on the console page. In particular, record the URL for **Administer and Setup**. This URL is used in the next section.

Tip: To return to the console home page without making a change, press **Ctrl-C**.

The VM is now deployed and ready to use. The next step is to configure Echidna.

Chapter 3: Configure Echidna

This chapter describes how to create a basic configuration to get Echidna working quickly. Administrators can update the Echidna configuration later.

It describes how to set up the mCodeXpress authentication method. This method authenticates end users using a one-time passcode (OTP) generated on their mobile device using the Salt mCodeXpress app. The mCodeXpress authentication method is quick to set up and the app is free.

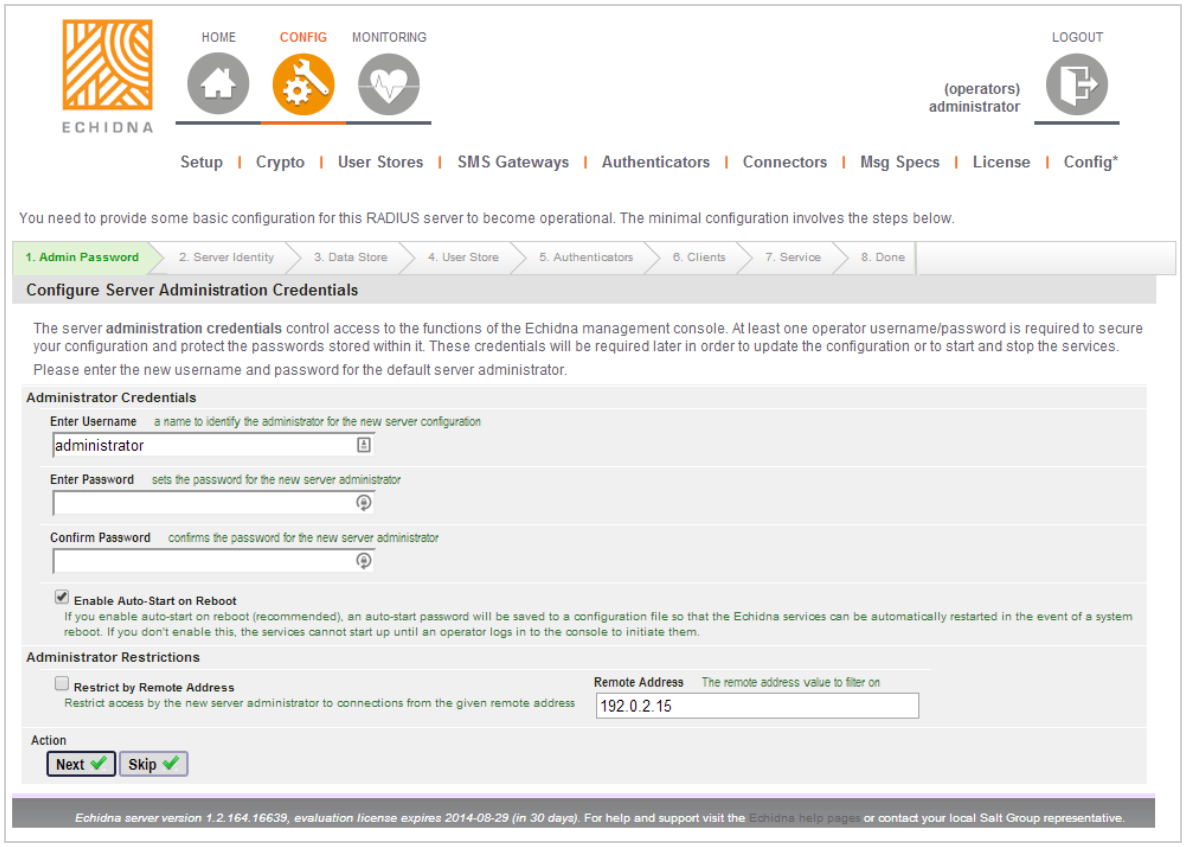
At the end of this book, use the *Administration Reference* for information about updating the Echidna configuration.

3.1 Sign in to the Administration console for Echidna

The Echidna Administration console is a web application that is secured by a self-signed certificate. For now, you can either trust this certificate or accept the security warning. When you have finished the steps in this book, use the instructions in the *Administration Reference* to generate and install a public or other SSL certificate.

1. Find the URL for the Administration console for Echidna on the VM console page shown on p. 11. Use the URL listed for **Administer and Setup**.
2. On a computer in the same LAN as Echidna, open this URL in a web browser.
3. The Administration console is secured with a certificate. When the security alert appears, accept the warning or install the certificate.

   The first page of the Setup wizard opens:



The next step is to use the Setup wizard to configure Echidna.

3.2 Configure Echidna

The following steps describe how to use the Setup wizard to get Echidna working quickly. These settings can be updated later.

1. **Admin Password**: Create new administration credentials for signing in to this Administration console, then click **Next**.
2. **Server Identity**: Click **Next** to accept the default keystore values, which creates an initial self-signed SSL certificate.
3. **Data Store**: Click **Next** to accept the defaults, which creates an embedded Derby database to store audit records, token data, and user information. This can be switched to a different database later.
4. **User Store**: Connect Echidna to the user store. Additional user stores can be added later. The following steps show how to connect to Active Directory or another LDAP user store:
   a. Choose the user store type: **ACTIVE_DIRECTORY** or **GENERIC_LDAP**.
   b. Enter a name for this user store. This name will appear in menus throughout this console.
   c. Enter the username required to bind to the user store.
   d. Enter the password required to bind to the user store.
   e. Update the URL to connect to the user store.
   f. Enter a base DN to start searches for users.
   g. Leave the search and list filters unchanged.
   h. Click **Validate** to test the connection to the user store. Echidna attempts to connect to the user store, and the log appears at the bottom of the page. Fix any errors, then click **Validate** again.
   i. Click **Next** when the connection is successful.
5. **Authenticators**: Configure the authentication methods that will be set up automatically.
   a. Enter **PREFMECH** in the first box. This attribute name is set up by default. Later, use an alias to connect an existing userstore attribute to the PREFMECH attribute.
   b. Select all checkboxes. The steps in this book assume that all three methods are selected.
   c. Click **Next**.
6. **Clients**: For this simple Echidna system, choose whether to set up only the local computer as a client that can delegate authentication to Echidna, or to add genuine clients. More clients can be added later.
   a. To enable the local computer can act as a RADIUS client to Echidna, click **Set Random**.
   b. (Optional) Enter the hostname or IP address of a client, and then create a shared secret for the client.
   c. Click **Validate**. Echidna tests whether the password values match and then tries to resolve the name or IP addresses using the network. Repair any errors that appear at the bottom of the page.
   d. Click **Next**.
7. **Service**: Click **Next** to create two basic authentication services (RADIUS and web services). Applications can use one of these addresses to connect to Echidna.

Echidna is now configured to connect to a user store and accept authentication requests.

The next step is to tell Echidna to listen for requests to these authentication services.

3.3 Start the RADIUS and HTTP listeners

This allows Echidna to listen for requests for its RADIUS and HTTP services.

1. In the Administration console, click **CONFIG**, then click **Connectors**.
2. Note the list of access points on the left. Any access points with this icon are stopped: 
3. For any stopped listeners, click on the listener name to see its details.
4. Click **Start** at the bottom of the page. The icon changes to show that the listener is running: 

3.4 Allow clients to invoke Echidna

Clients are other services or applications that ask Echidna to perform authentication services. Clients can communicate with Echidna via RADIUS or web services. For RADIUS, the client must be identified using the host name or IP address of the server making the connection.

To allow a client to use Echidna for user authentication, the client needs to be added to the configuration of Echidna.

To add a client to Echidna, follow these steps:

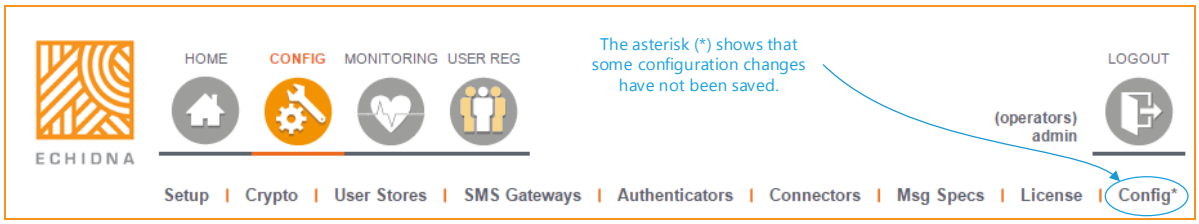
1. In the Administration console, click **CONFIG**, then click **Connectors**.
2. In the **Clients** section on the left, click **new**.
3. Enter the host name of the Echidna client. This name must be resolvable into an IP address.
4. Click **Next**. The details of the new client appear.
5. If required, update the details.
6. Ensure that the client's shared secret is entered in the **Password** fields.
7. Click **Update**.
8. Update the SSL certificate.

3.5 Save the configuration

After changing the configuration, save the changes to the configuration:

1. In the Administration console, click **CONFIG**, then click **Config***, which is on the far right under the **LOGOUT** button.

   The asterisk (*) shows that the configuration includes changes that are not saved yet.



2. Click **Save Current**.

Consider backing up the new configuration as a file. For information about backing up the configuration as a file and restoring from a file, see “Backup and Recovery” in the *Administration Reference*.

3.6 Update the SSL certificate

When the Echidna appliance starts for the first time, it generates an SSL certificate and a private key for use by the Apache Tomcat server. Echidna uses this self-generated certificate for all Tomcat HTTPS connectors. After deployment, the administrator should update this certificate.

This section descibes how to use Echidna to generate new keys and certificates. If these steps are followed, Echidna lets the administrator manage and view its own certificates.

3.6.1 Configure Echidna to create a certificate request

During installation, the Echidna appliance creates its own keystores and generates its own keys. The following steps describe how to update these keystores to use externally issued certifications from a recognised certification authority (CA).

1. In the Administration console, click **CONFIG**, then click **Crypto**.
2. In the **Keystores** section on the left, click ***new***.
3. Enter the following information:
   - **New keystore name**: This name will appear in other screens in the Administration console. Suggested name is **tomcatKS**.
   - **Select keystore type**: Choose **JKS**.
4. Click **Next**.
5. In the **Cryptographic Keystore Configuration** page, make the following changes to identify the keystore that Echidna has already created:
   - **Keystore Path**: Enter **/opt/Salt/Echidna/tomcatid.jks**.
   - **Password Protector**: Ensure that this is set to **utilProt**.
   - **Password**: Set this to **Password_1**.
6. Click **Update**.

   The configuration now includes the details of the Tomcat server's own keystore.
7. Note the existing key aliases that are listed in the **Keystore Keys** section. For the standard appliance script, this is **tomcatid**. This will be used in the next section.

3.6.2 Generate a certificate request

The following steps describe how to generate a certificate request that includes the organisation's details.

1. In the Administration console, click **CONFIG**, then click **Crypto**. Alternatively, continue directly from Step 6 in the previous section.
2. In the **Password Protectors** section on the left, click ***new***.
3. Enter a name such as **tomcatIfProt**, then click **Next**.
4. On the **Cryptographic Password Protection Configuration** page, make the following changes to specify the keystore that this protector will appy to:
   - **Key Store**: Select the keystore name that was specified in Step 3 in the previous section (**tomcatKs**).
   - **Key Alias**: Enter the alias name that was noted in Step 7 in the previous section (**tomcatid**).
   - **Algorithm**: RSA
   - **Subject DN**: Enter the subject DN that should appear in the certificate to allow Echidna to generate the appropriate certificate request. The CN should be set to the fully-qualified domain name that the Echidna server will be known as externally.

     For example, CN=localechidna1.exco.net.au,OU=Staff,O=Example Corp,C=AU.
5. Click **Update** to save the changes.

   The details of the new password protector appear, including the details of the existing certificate. Note that the DN of the existing certificate is not the same as the DN requested for the new certificate, in Step 4.
6. Click the **[PKCS#10 Certificate Request]** link, which is next to **Public Key Certificate Chain**. This generates the .P10 request that will be submitted to the certification authority (CA).
7. Download the P10 request file.
8. Send the P10 request file to the CA. Also ask the CA to return the entire certificate chain in a PKCS#7 file (p7c or p7b), rather than just the end-entity certificate in a .cer file.

When the CA sends the new certificate, follow the steps in the next section.

3.6.3 Import the new certificate into Echidna

After the CA sends the new certificate, follow these steps to import it into Echidna.

1. If the certificate chain is available in a PKCS#7 format, make sure the file is named with a .p7c extension. Otherwise, use the X.509 certificate (.cer or .crt).
2. Save the certificate in a location that is accessible from the computer that runs the Administration console browser.
3. In the Administration console, click **CONFIG**, then click **Crypto**.
4. In the **Password Protectors** section on the left, click on the password protector that was created in Configure Echidna to create a certificate request on page 17
5. In the **Import Certificate Chain** section near the bottom, click **Choose File**, then navigate to the certificate.

   The certificate is listed next to the **Choose File** button.
6. Click **Import** to update the SSL certificate.
7. Save the configuration (see page 16).
8. Restart Echidna in one of these ways:
   - Restart the virtual machine.
   - Log in to the VM console and restart Tomcat.

3.6.4 Use an alternative SSL context for outgoing connections

Echidna may be configured to make outgoing SSL connections to LDAP servers (LDAPS) or HTTPS-based SMS gateways or web services. In those cases, there is an **SSL context** configuration item to allow selection of the relevant SSL settings.

To use an alternative configuration to the default **sslctx** set, create a new SSL context:

1. In the **Administration** console, click **CONFIG**, then click **Crypto**.
2. In the **SSL Contexts** section on the left, click **new**.
3. Enter a name for the new context. For example, use **sslctx-out**.
4. Click **Next**.
5. On the **SSL Context Configuration** page, update the following:
   - **Protocol**: Enter **TLS** or **TLSv1.1**.
   - **Priv Key Name**: If SSL mutual authentication will be used, enter the the password protector that was configured with the private key, in Generate a certificate request on page 18. For example, **tomcatIdProt**.
   - **Trust Store Name**: Select the keystore that contains the trusted root certificates that will be allowed in the SSL connections. This is likely to be **trustKs**.

3.7 Upgrade from the evaluation license

The default license installed with Echidna is an evaluation license. This evaluation license limits the number of tokens that Echidna can use, and the license expires one month after Echidna is configured. To remove these limitations, register Echidna by applying for a license.

3.7.1 Apply for a license

1. In the Administration console, click **CONFIG**, then click **License**.
2. On the left, click **request new**.
3. Enter the following information:
   - **Organization Name**
   - **Server Expiry Date**: Request the length of this license. Leave this unchanged to request the default length (10 years).
   - **Registered Token Limit**: The number of tokens permitted per authenticator.
   - **Initial SMS credits**: (Optional) Request that Salt Group set up an account with an SMS Gateway, and provide this number of SMS messages.
4. Click **Next**.
5. Make any changes, and then click **Update**.

   Echidna generates a license request.
6. Send the license request to Salt Group in one of these ways:
   - Click **Email license request** to open a pre-written email in your email client.
   - Click **Download license request** and then attach the request file to an email to sales@saltgroup.com.au.

   Include any further requests in the body of the email.

3.7.2 Add a license to Echidna

A Salt Group representative will email back a license file, which is an XML file that updates the Echidna configuration.

1. In the Administration console, click the **Configuration** tab.
2. Click **Import**, and then navigate to the license file.
3. To check that the new license imported correctly, click the **License** tab and check that the license type is listed as `registered`.
4. Save the configuration (see page 16).

Chapter 4: Set up the User Support and Self Service consoles

Echidna comes with two supporting consoles:

- **User Support console**: Lets support staff manage end users and their tokens.
  By default, **no** users have access. This protects Echidna until it is ready for deployment.
- **Self Service console**: Lets end users register and manage their own tokens.
  By default, **all** users in the user store have access.

The URLS for these consoles are listed on the appliance console screen, as shown in Configure the new virtual machine on page 11.

4.1 Give access to the User Support console

Access to the User Support console is controlled by roles stored in Echidna. For a full list of these roles and the permissions they grant, see Chapter 4 of the *Administration Reference*.

To grant access, update the membership rules for the roles in Echidna or assign the users to the relevant user store groups. Roles in Echidna can be granted by group membership or by the presence of named attributes with matching values in the user store entries.

After running through the Setup wizard, Echidna specifies the following roles that can sign in to the **User Support** console:

Role name	Privileges granted to members of this role
GRANT 2FA-User Full Administration	Members can sign in to the User Support console and make changes to the tokens and users.
GRANT 2FA-User Readonly Administration	Members can sign in to the User Support console but not make any changes. This role is useful for auditors.
GRANT 2FA-Token Administration	Members can sign in to the User Support console to import and manage tokens only.
GRANT 2FA-User Registration	Members can sign in to the User Support console to create, edit, and delete user records in a database store, if that service has been enabled. See Allow users and groups to be managed in the User Support console on page 23.

A user store is very unlikely to contain groups with those names, which effectively means that no-one can sign in to the User Support console. This protects Echidna until it is ready for use.

To allow support staff to sign in to the User Support console, either update the Echidna configuration, or create or rename a group in the user store. The following steps describe how to update the Echidna configuration to match groups that already exist in the user store.

1. In the user store, identify or create a group that contains only users that should be allowed to manage users and tokens in the User Support console.
2. Click **CONFIG**, then click **Authenticators**.
3. In the list on the left, select **AuthenticationProc**.
4. Scroll down to the **User Roles** section, then find the following sections:
   - **user-support-updater**: Defines the groups with full access to the User Support console
   - **user-support-reader**: Defines the groups with read-only access to the User Support console
5. In the **Granting Groups** section, change **Group Name** to match the actual group in the user store.
6. If required, use the **Add** button  on the far right to add more groups.
7. At the bottom of the page, click **Update**. Echidna immediately uses the new configuration.
8. Save the configuration (see page 16).

4.2 Allow users and groups to be managed in the User Support console

By default, Echidna does not allow users and groups to be edited in the User Support console.

In many organisations, this works well, because they already have tools for editing users. However, if an organisation needs the User Support console to let support staff edit users, this can be turned on. It works only for users that are stored in a database, not in an LDAP directory or Active Directory.

The following steps describe how to expand the User Support console to allow its users to edit, create, and delete users and groups.

1. Click **CONFIG**, then click **Connectors**.
2. In the **Services** section on the left, click **webServices**.
3. Add a new service for the extra functionality:
   a. Scroll down to the **User Reg Services** section.
   b. Click the green **Add** button  .
   c. Type **userReg** in the **New Name** box, then click **Add**.

   The new **userReg** service is now listed in both the **User Reg Services** section, and in the **Published Services** section.
4. In the **Published Services** list, check the **Service Enabled** box for the **userReg** service.
5. At the bottom of the page, click **Update**. Echidna immediately uses the new configuration.
6. Save the configuration (see page 16).

4.3 Restrict access to the Self Service console

Echidna includes a **Remote Users** role, which controls access to the **Self Service** console. All users in this role can access the Self Service console.

After the Setup wizard is complete, the **Remote Users** role includes all users from all user stores as members. This lets all users sign in to the Self Service console and set their own login method preference and register their own tokens.

If this works well in an organisation, there is no need to change the Echidna configuration. However, it is possible to restrict the Self Service console to only some users.

Use these steps to allow only some users to access the Self Service console.

1. Click **CONFIG**, and then click **Authenticators**.
2. In the list on the left, select **authenticationProc**.
3. Scroll down to the **User Roles** section, and then find the **remote-user** section. This section defines the **Remote Users** role.
4. To restrict access to only these users with a particular *attribute*:
   a. Choose an option in the **Authentication Attributes Combination** list:
      - **ONE_OF**: Users with at least one of the following attributes will be able to sign in to the Self Service console
      - **ALL_OF**: Only users with all of the following attributes will be able to sign in to the Self Service console.

   b. Add an attribute by clicking the **Add** button on the far right of the **Auth Attributes** table. Use the **Delete** button to remove an attribute.
5. To restrict access to only these users in a particular *group*:
   a. Choose an option in the **Granting Groups Combination** list:
      - **ONE_OF**: Users in at least one of the following groups will be able to sign in to the Self Service console. If the list is empty, no users will have access.
      - **ALL_OF**: Only users in all of the following groups will be able to sign in to the Self Service console. If the list is empty, all users will have access.

   b. Add a group by clicking the **Add** button on the far right of the **Granting Groups** table. Use the **Delete** button to remove a group.
6. At the bottom of the page, click **Update**. Echidna immediately uses the new configuration.
7. Save the configuration (see page 16).

Chapter 5: Set up tokens

When Echidna has been connected to the organisation's other systems, users can use the self-service console to register their own tokens, or an operator can use the User Support console to register the token on the user's behalf.

This chapter describes how to use the User Support console to register the following types of tokens:

- Set up Salt mCodeXpress mobile tokens (see page 26)
- Set up OATH tokens (see page 34)
- Set up Google Authenticator tokens (see page 42)
- Set up temporary password tokens (see page 51)

For a full description of all supported token types, see the *Administration Reference*. The full list of supported tokens is as follows:

- Salt mCodeXpress OTP
- SMS-delivered OTP
- OATH HOTP (Generic)
- OATH TOTP
- OATH OCRA
- OATH HOTP (YubiKey)
- Limited-use temporary password
- Password only
- CAPTCHA images

5.1 Set up Salt mCodeXpress mobile tokens

Token records for registered mCodeXpress mobile tokens are stored in the Echidna database, but no record exists until the application on the device is registered. This means that there is no batch seed data import process as there is for the OATH tokens.

Once registered, mCodeXpress tokens can be resynchronised, suspended, or cancelled.

mCodeXpress tokens also do not have any intrinsic serial number, but they can be assigned one.

This section includes the following:

To follow this section, you will need:

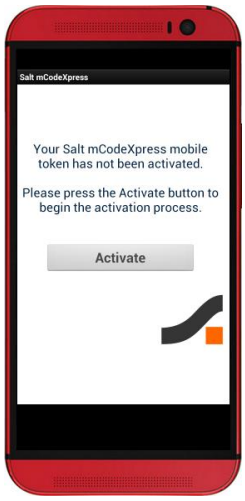
- A user record that is available for testing
- A smartphone or tablet that can install apps from one of the following app stores:
  - iTunes App Store
  - Google Play
  - Windows Phone Store

5.1.1 Install the Salt mCodeXpress app

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

5.1.2 Activate the Salt mCodeXpress token

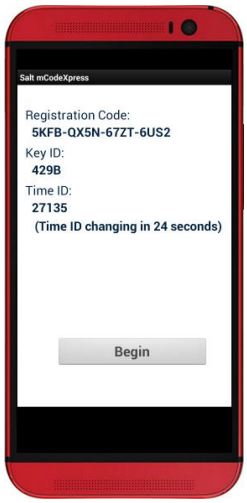
1. Ensure that the device has network access.
2. Open the Salt mCodeXpress app and touch **Activate**:



3. 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



5.1.3 Register a Salt mCodeXpress token using the Self Service console

The user can register their own mCodeXpress token, using the **Self Service** console. Alternatively, these other people can register the token on behalf of the user:

- User support staff can Register a Salt mCodeXpress token on behalf of a user (User Support console) (see page 29).
- Administrators can Register an mCodeXpress token on behalf of a user (Administration console) (see page 30).

Ask the user to follow these steps to register their own token:

1. In the web browser, go to the **Self Service** console. Ask the administrator where this page is. For a list of supported browsers, see Check the system requirements on page 7.
2. Sign in to the **Self Service** console using your usual domain username and password.
3. Enter the following information that appears on the device:

Code	Comment
Registration code	Include every character. This code is not case-sensitive. Any hyphens (-) are ignored.
Key ID	This code is not case-sensitive.
Time ID	This code changes frequently. If you use an old code, registration may not work or the OTPs may be rejected.

4. Activate the Salt mCodeXpress token (see page 27).
5. On the **Self Service** console, click **Register Token**.
6. When the registration is complete, tap **Begin** on the device. These codes will never be displayed again, so make sure that registration was successful before dismissing this screen.

   Note: To re-register the mCodeXpress app, see Reset the Salt mCodeXpress PIN on page 33.

You can now use Salt mCodeXpress to create a one-time passcode (OTP) to let you access your organisation's systems.

5.1.4 Register a Salt mCodeXpress token on behalf of a user (User Support console)

If a user cannot use the Self Service console, an operator can register tokens on the user's behalf.

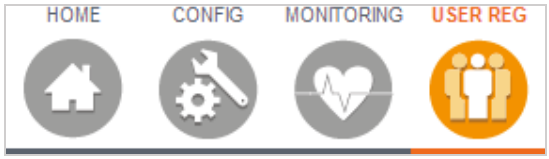
1. Ask the user to follow these steps on their device:
   a. Install the Salt mCodeXpress app (see page 27).
   b. Activate the Salt mCodeXpress token (see page 27).
2. Log in to the **User Support** console.
3. Click **Activate**, then click **Assign mCodeXpress**.
4. Search for the user.
5. Enter the 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**.

5.1.5 Register an mCodeXpress token on behalf of a user (Administration console)

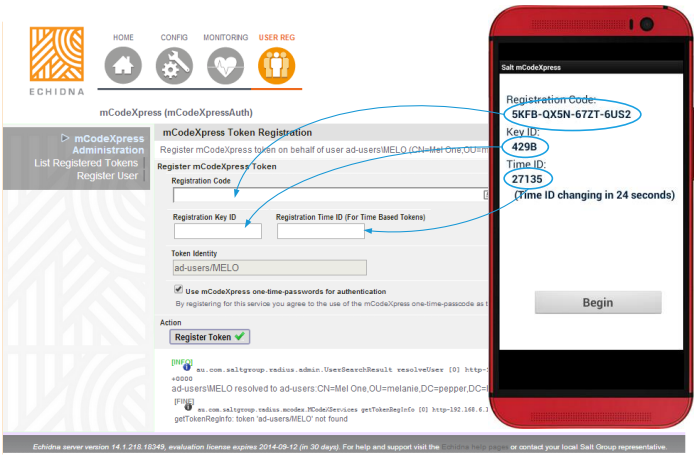
Salt Group recommends that users register their own Salt mCodeXpress tokens using the Self Service console (see page 28). Alternatively, user support staff can register the token on behalf of the user, using the User Support console (see page 29).

If neither of these consoles is ready for use, an Echidna administrator can register a Salt mCodeXpress token on behalf of a user.

1. In the Administration console, click the **USER REG** icon at the top of the page:



2. Click the **mCodeXpress** link directly under the **USER REG** button.
3. In the list on the left, click **Register User**.
4. Enter the user's details:
   - **RADIUS client**: The client that the user is authenticating from. For a simple installation, use **localhost**.
   - **Domain**: The user store that contains this user's record.
   - **Username**
5. Click **Find User**. If the user is found in the user store, the **Register mCodeXpress Token** page appears.
6. Ask the user to Activate the Salt mCodeXpress token (see page 27).
7. Enter the following information from the app on the device:
   - **Registration code**: This code is not case-sensitive.
   - **Key ID**: This code is not case-sensitive.
   - **Time ID**: This code changes frequently. If you use an old code, registration may not work or the OTPs may be rejected.



8. Ensure that the **Use mCodeXpress one-time-passwords for authentication** box is checked.
9. Click **Register Token** or **Re-register token**. The log messages at the bottom of the page show that the token is registered.
10. To check that the token was registered correctly, click **List Registered Tokens** on the left.

The mCodeXpress app on the device is now registered as a token for the user.

5.1.6 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 ID**s 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 resolved 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`

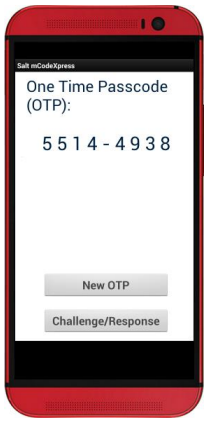
If this process does not resolve the problem, reset and re-register the user's mCodeXpress token.

5.1.7 Get a OTP from Salt mCodeXpress

Salt mCodeXpress generates one-time passcodes (OTPs). The device does not need network access for these steps.

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

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



2. Enter the OTP exactly as it appears on the device.

5.1.8 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.

1. On the device, open the **Salt mCodeXpress** app.
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.
4. The Salt mCodeXpress app is now unregistered.
5. To register again, follow the steps in Register a Salt mCodeXpress token using the Self Service console on page 28.

5.2 Set up OATH tokens

Echidna can work with the following open standard OATH hardware tokens:

- OATH HOTP tokens (see RFC 4426)
- OATH TOTP tokens (see RFC 6238)
- OATH OCRA tokens (see RFC 6287)

This section includes the following:

5.2.1 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.

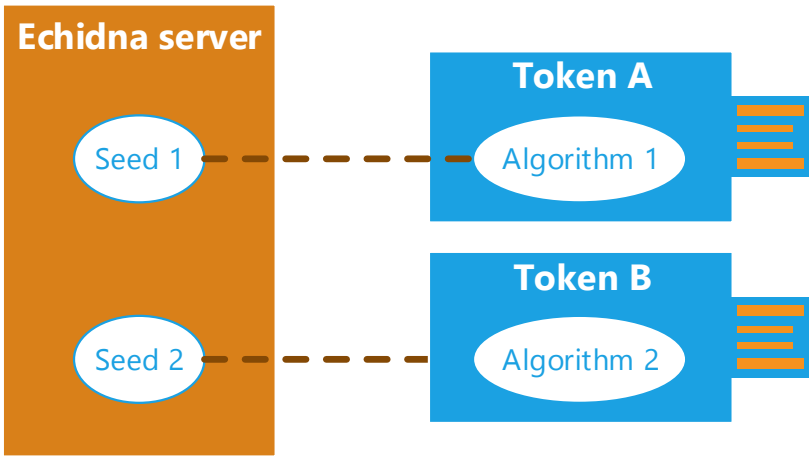


Figure 4: Echidna uses the token seeds to validate OTPs produced by OATH hardware tokens

The seed material should 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. Preferably, the token vendor protects the seed material with Echidna's public key.

5.2.2 How OATH tokens are set up

OATH tokens are set up with the following steps:

1. An administrator uses the Administration console to do these tasks:
   a. Set up an OATH token type (see on page 36).
   b. Export the public key from Echidna (see page 37).
2. The organisation acquires the OATH tokens.
3. The token manufacturer sends the purchased tokens to the correct people within the organisation.
   For example, an organisation might use the following roles":
   a. The **seed files** are sent to the person responsible for importing them into Echidna.
      This person must have the **GRANT 2FA-Token Administration** role in the User Support console.
   b. The **tokens** are sent to the person responsible for giving them to staff.
      This person must have the **GRANT 2FA-User Registration** role in the User Support console.
   For information about these roles, see Give access to the User Support console on page 22.
4. The person with the GRANT 2FA-Token Administration role imports the OATH seed material. See Import OATH seed files into Echidna on page 38.
5. The person with the GRANT 2FA-User Registration role gives the tokens to users. See Validate an OATH token on page 39.

5.2.3 Set up an OATH token type

The Setup wizard automatically sets up token types for Salt mCodeXpress and for temporary passwords. However it does not automatically set up OATH authentication. This section describes how to set up an OATH token type.

1. In the Administration Console, click CONFIG, then click **Authenticators**.
2. In the **Authenticators** section on the left, click **new**.
3. Enter a **name** for the new authenticator and select one of the OATH authenticator **types**.
4. Select an existing Authentication Process that should reference the new authenticator. Salt Group recommends using the default **authenticationProc** process for OATH tokens.
5. In the **Action** section, click **Next**.
   The new authenticator is created with default values.
6. Change some or all of the default values.
7. At the bottom of the page, click **Update**. Echidna immediately uses the new configuration.
8. Save the configuration (see page 16).

5.2.4 Export the public key from Echidna

To protect the token seed files while they are being sent from the vendor, the administrator might need to use the Echidna public key.

1. In the administration console, click **CONFIG**, then click **Crypto**.
2. On the left, find the **Password Protectors** section, then click **serverIdentity**.
3. If the key pair has not been created, click the **Generate** link to create it using the key alias and DN specified on the same page.
4. To download the server public key in PGP format, click the **OpenPGP Public Key** link next to the **Public Key Certificate Chain** label.

   To download the server public key in X.509 certificate format, click the number in the **Serial Number** column.
5. Check the content of the downloaded key file (serverIdentity.asc). It should look similar to the following:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: BCPG v1.46

mI0EUtKBrWEEAMcgX93I3RsEAL3Zk/lR8q738pIEMy+6YfwA3Uo9QcAg+GTrsVrR
ysET08AQpN1tM765613YWS1NFX4/7VkC5RVQTjbydLeluGQkvYrxprKo0TSsZIHI
IojPREa8f+SenUTrM578BDbfkCMfQoNMUO4051/vrHXHNmLsdWLaBaCTABEBAAG0
F2NoaXJvbi5nb2xvbXRiYW5rLmxvY2FsiJwEEAECAvYFAlL64P8ACgkQdMxBms6Q
DTd6HQP+P+8ansM3fqHmhHE13G6THUWPoh4zMDzB368YMAhbW3vnI4jm3O9D3w2O
aF33N8apiXkwcC+2RmqjciujcDQRFp8RsfZoDY9R3yfAviXuA1tC+reUUbhWqE/+
KMGdacWnH2wbcQDoztnypcyauQB7wtltVTmDvVZkRLa4s8T1Jlw=
=mFHl
-----END PGP PUBLIC KEY BLOCK-----
```

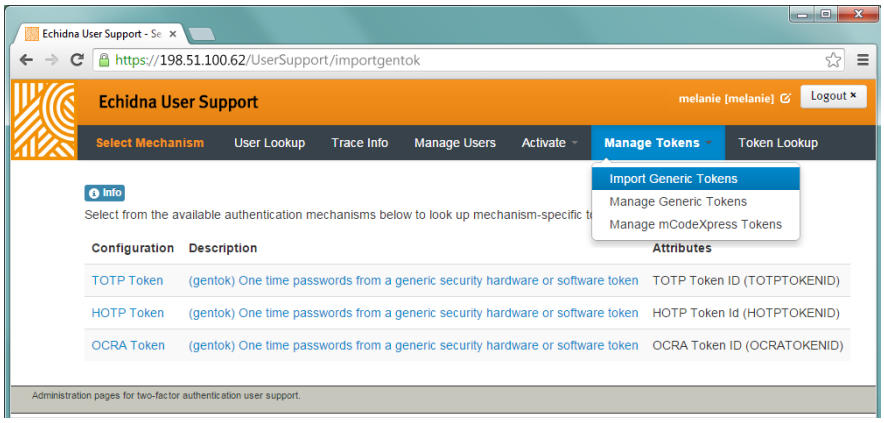
6. Give Echidna's public key to the token vendor.

5.2.5 Import OATH seed files into Echidna

Echidna uses the seed files to calculate the correct OTP or challenge-response for each OATH token.

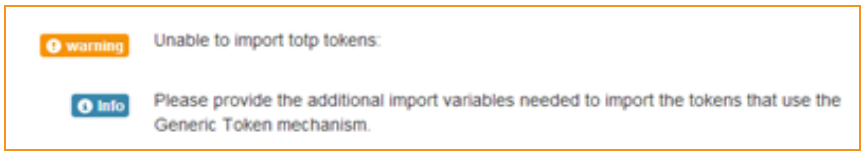
1. Log in to the **User Support** console, then click **Import Generic Tokens**.

   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
WLN3OxWItW4aH9gUePGAJgOaTH6+U0L6929DMgAaPSMuyx3ODqU2X7l5JhZnztwf
TNjkWZIL9cYY8xnkJUeYsYIDrNehZcwz0HJQKiNrqulHlkGdvycomRqNSziziv+2
g2YmWhBZEwTB6UgCxpbFyrpKnM5yEQWUkBt5nO75gQ==
=3DiF
-----END PGP MESSAGE-----
```

8. Echidna checks that the data is valid and that 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.

5.2.6 List all registered OATH tokens

1. Log in to the User Support console.
2. Click M**anage Tokens**, then click **Manage Generic Tokens**.
3. If necessary, choose from the list of OATH token types.

   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 in validating an OTP.
   - **Assigned User**: The user to whom this token is assigned. Click this link to see the user details.

5.2.7 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 a 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 39).
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.

5.2.8 Assign an OATH token to a user

Use this option to activate all tokens that support the OATH HOTP, OATH TOTP, or OATH OCRA standards, except for YubiKey tokens, which have a different token registration process. For information about assigning YubiKey tokens, see the *Administration Reference*.

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.

5.2.9 Resynchronise an OATH token

The time from the clock on the OATH 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. Press and hold down the button on the token, and then enter the display code in that is shown in the **First OTP** box.
4. Continue to hold down the button until a new code appears, and then enter the new code 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`

5.3 Set up Google Authenticator tokens

Echidna can work with the following Google Authenticator tokens:

- HOTP tokens (see RFC 4426)
- TOTP tokens (see RFC 6238)

Token records for registered Google tokens are stored in the Echidna database, but no record exists until an account is registered on the device.

Once registered, Google tokens can be resynchronised, suspended, or cancelled.

This section includes the following:

To follow this section, you will need:

- A user record that is available for testing
- A smartphone or tablet that can install apps from one of the following app stores:
  - iTunes App Store
  - Google Play

  For BlackBerry device, the app can be downloaded using a web browser.

5.3.1 Set up a Google token type

1. In the Administration Console, click CONFIG, then click **Authenticators**.
2. In the **Authenticators** section on the left, click **new**.
3. Enter a **name** for the new authenticator and select one of the OATH authenticator **types** (TOTP/HOTP).
4. Select an existing Authentication Process that should reference the new authenticator. Salt Group recommends using the default **authenticationProc** process for Google tokens.
5. In the **Action** section, click **Next**.

   The new authenticator is created with default values.
6. Change some or all of the default values.
7. At the bottom of the page, click **Update**. Echidna immediately uses the new configuration.
8. Save the configuration (see page 16).

5.3.2 Install the Google Authenticator app

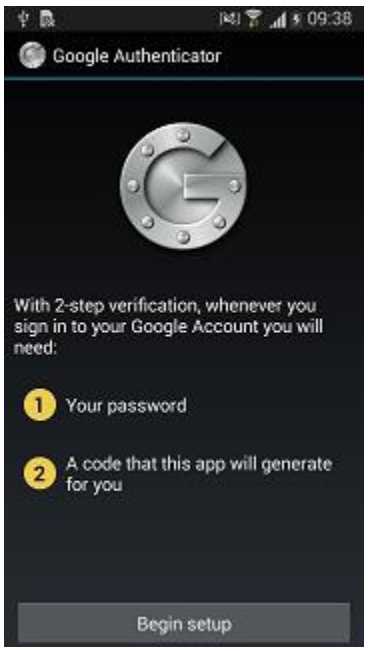
1. On the device, search in the appropriate app store for **Google Authenticator**.
2. Download and install the app.

5.3.3 Register a Google token

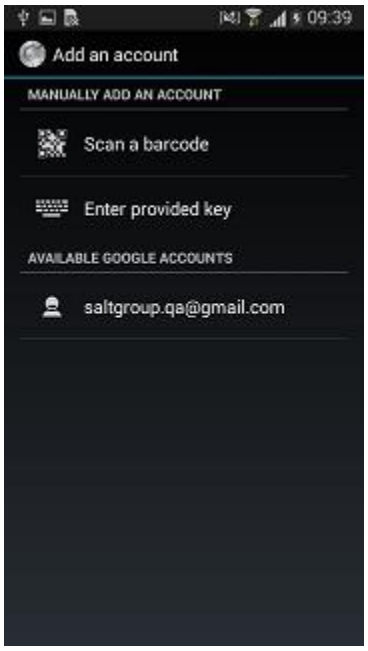
The user can register their own Google Authenticator token, using the **Self Service** console.

Ask the user to follow these steps to register their own token:

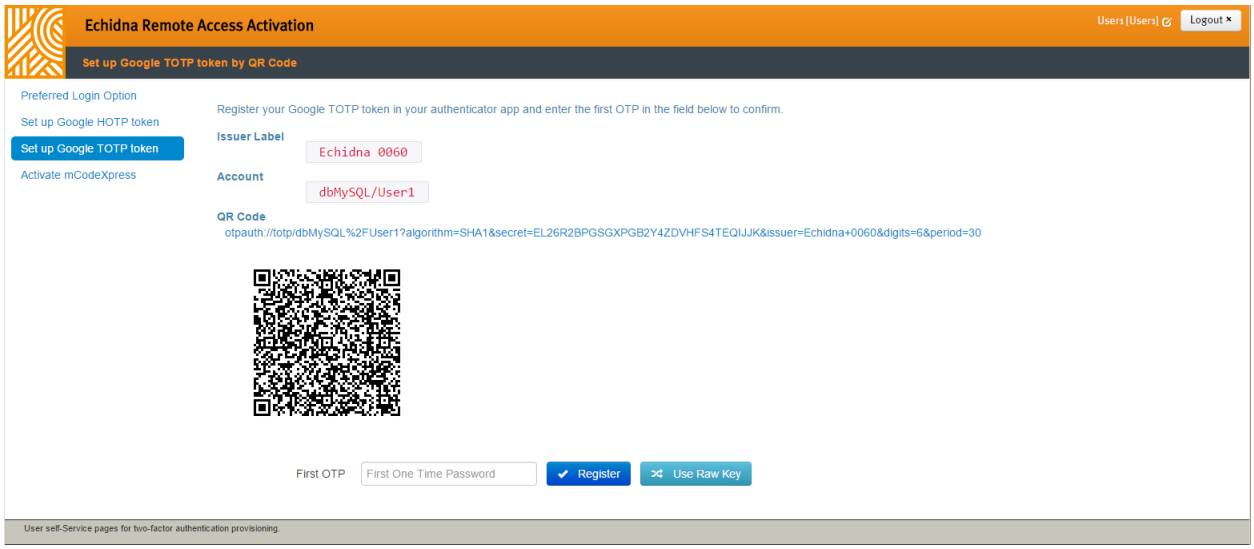
1. In the device, open the Google Authenticator app and touch **Begin Setup**:



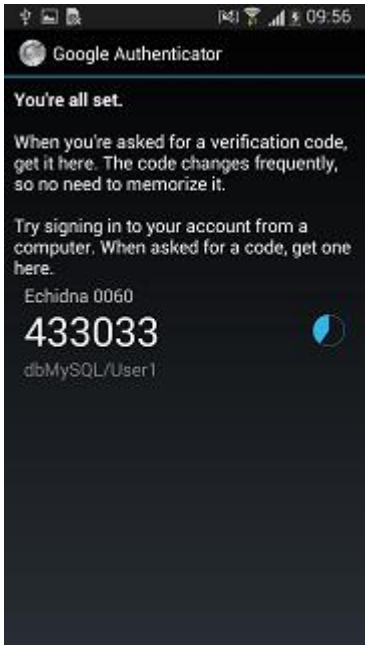
2. An account can be manually added by scanning a QR code or entering the key manually. Touch **Scan a barcode**:



3. In the web browser, go to the **Self Service** console. Ask the administrator where this page is. For a list of supported browsers, see Check the system requirements on page 8.
4. Sign in to the **Self Service** console using your usual domain username and password.
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:

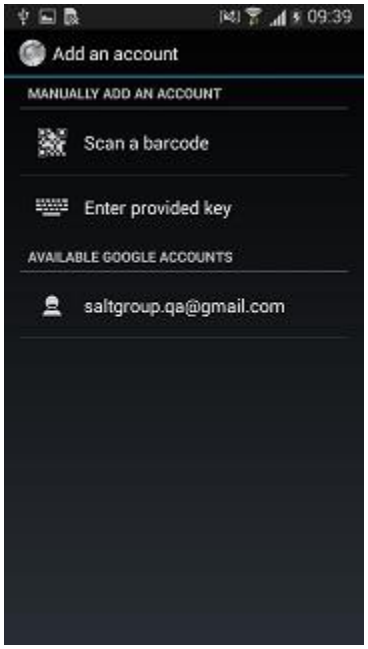


7. Here's the screen after scanning a QR code for a Google TOTP token:

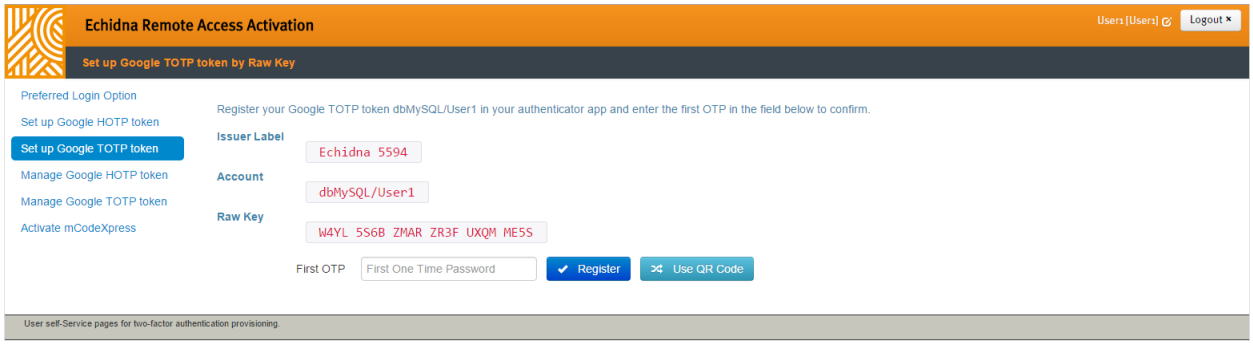


8. Here's the screen after scanning a QR code for a Google HOTP token:

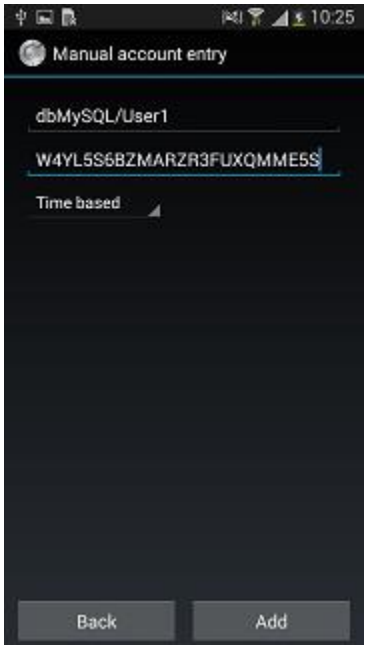
9. Alternatively, a Google token can be registered by entering the raw key manually. On the device, touch **Enter provided key**.



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

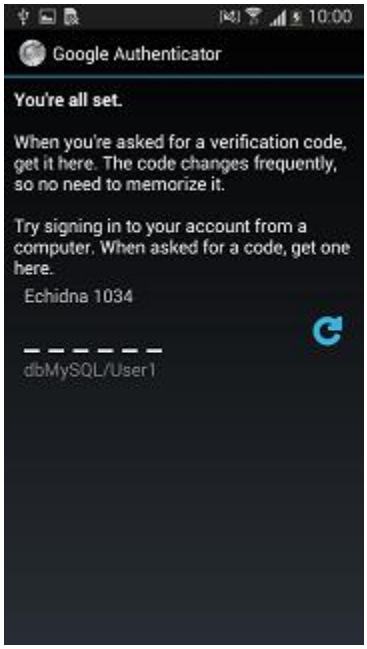


11. Enter the **Account** name and **Raw Key** from the Self Service console to the device. Select **Time based** for TOTP token or **Counter based** for HOTP token. Touch **Add**.

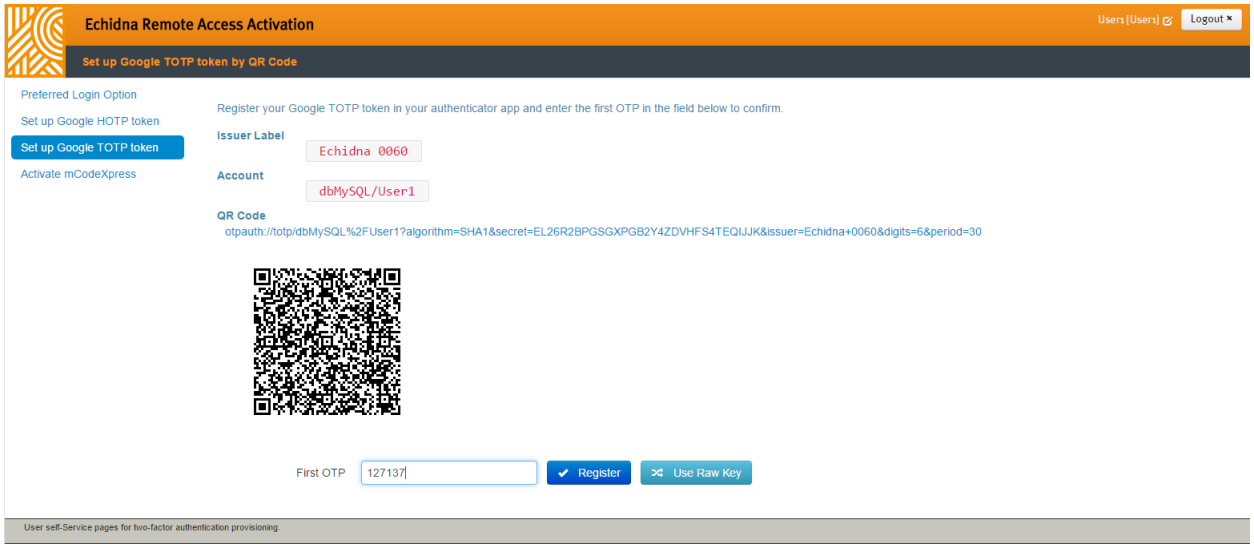


12. Here's the screen after manual raw key entry for a TOTP token:

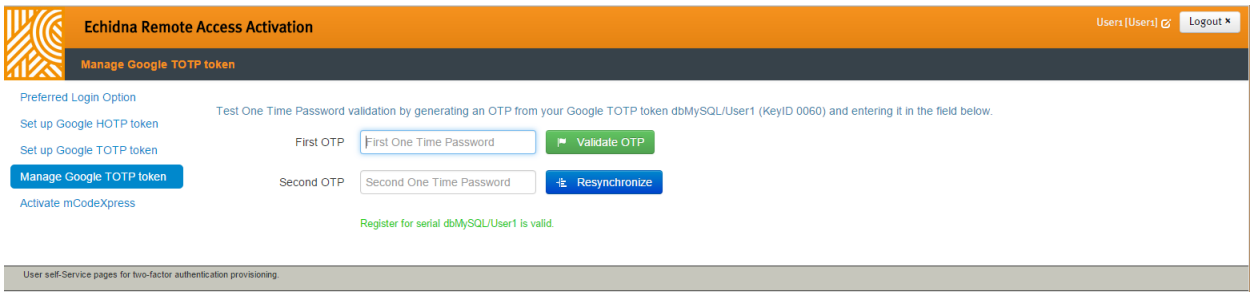
13. Here's the screen after manual raw key entry for a HOTP token:



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



15. Here's the Self Service console after a successful registration:



You can now use the Google Authenticator app to create a one-time passcode (OTP) to let you access your organisation's systems.

5.3.4 List all registered Google tokens

1. Log in to the User Support console.
2. Click M**anage Tokens**, then click **Manage Generic Tokens**.
3. If necessary, choose from the list of Google token types.

   A list of all of the recorded Google tokens appears, showing the following information:
   - **Token/User ID**: The primary identifier of the token. 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 registered into Echidna.
   - **Last Success**: The most recent time that the token was used to successfully validate an OTP.
   - **Last Failure**: The most recent failed attempt to use the token to validate an OTP.
   - **Failure Count**: The number of failed attempts in validating an OTP.
   - **Assigned User**: The user to whom this token is assigned. Click this link to see the user details.

5.3.5 Validate a Google 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.

Using the Self Service Console:

1. Use the Google Authenticator app to generate an OTP.
2. In the Self Service Console, select **Manage Google TOTP/HOTP token**.
3. Enter the OTP in the **First OTP** box, then click **Validate OTP**.
4. Look for the result at the bottom.

Using the User Support Console:

1. Use the Google Authenticator app to generate an OTP.
2. In the User Support Console, List all registered OATH tokens.
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.

5.3.6 Resynchronise a Google token

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

Using the Self Service Console:

1. Log in to **Self Service** console.
2. Select **Manage Google TOTP/HOTP token**.
3. Use the Google Authenticator app to generate an OTP. Enter the OTP in the **First OTP** box.
4. Generate another OTP and enter it in the **Second OTP** box.
5. Click **Resynchronize**.
6. If the synchronisation is successful, a message similar to the following appears at the bottom of the page:

   Resynch for serial dbMySQL/User1 is valid.

Using the User Support Console:

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. Use the Google Authenticator app to generate an OTP. Enter the OTP in the **First OTP** box.
4. Generate another OTP and enter it 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 TOTP Token token dbMySQL/User1`

5.4 Set up temporary password tokens

The Temporary Password authentication method is not intended to be the primary authentication method for a user. Instead, it works as a backup mechanism for the times when other authentication methods are not available.

For example, a temporary password can be assigned by an operator if a user has lost their hardware token.

The password can only be used a fixed number of times, and will expire after a fixed time period. Once the password is used or expired, the user reverts to their original authentication mechanism.

For a full description of temporary passwords, see Chapter 6 in the *Administration Reference*.

This section includes the following:

5.4.1 Assign a temporary password token to a user (User Support console)

1. Log in to the User Support console.
2. Click **Assign Temp Password**. This option might be under the Activate menu, or it might appear on the menu bar itself.
3. Enter the username and domain of the user that needs the temporary password, then click **Search**.

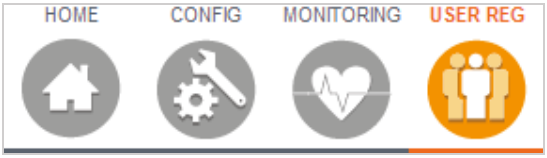
   If the user already has a temporary password, its details are displayed (but not the password itself), and the **Register** and **Deregister** options are available.
4. Click **Register** to assign a new password value (overwriting any existing temporary password).

   The temporary password is created immediately, and it is displayed on the screen.
5. Tell the user their new password immediately.
6. After the user has been told their password, click **Dismiss**. The password will never again appear in the User Support console.

5.4.2 Assign a temporary password token on behalf of a user (Administration console)

The recommended method of assigning a temporary password is to allow an operator to perform this on behalf of the user in the User Support console. If console is not yet available, an Echidna system administrator can assign the password on behalf of the user.

1. In the Administration console, click the **USER REG** icon at the top of the page:



2. Click the **Temp Passwords** link directly under the **USER REG** button.
3. In the list on the left, click **Register User**.
4. Enter the user's details:
   - **RADIUS client**: The client that the user is authenticating from. For a simple installation, use **localhost**.
   - **Domain**: The user store that contains this user's record.
   - **Username**
5. Click **Find User**.

   If the user is found in the user store, the **Temporary Password Registration** page appears.
6. Enter the following details:
   - **Maximum Allowed Logins (1 - 1)**: The number shows the acceptable range. In this example, only 1 can be entered here.

     This range is configured in Max Use Count in the Temporary Password authentication (see the *Administration Reference* for more information.
   - One of the following:
     - **Expiry Timestamp**: The date of expiry, in the format yyyy-MM-dd HH:mm:ss.
     - **Validity Period:** The amount of time for which the password remains valid, in the format HH:MM:SS. The maximum is 24:00:00.

5.4.3 Delete a temporary password

1. Log in to the User Support console.
2. Click **Assign Temp Password**. This option might be under the **Activate** menu, or it might appear on the menu bar itself.
3. Enter the username and domain of the user that needs the temporary password, then click **Search**.

   If the user already has a temporary password, its details are displayed, and the **Deregister** button is available.
4. Click **Deregister** to delete the password and its record immediately.

Chapter 6: Manage tokens

6.1 Suspend, revoke, or delete a token

The Administration console cannot manage tokens. The following steps use the User Support console.

This section applies to all token types except CAPTCHA and Temporary Password.

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. For example, use **Suspend** when transferring a token from one user to another.
- **Revoke**: Suspend a token permanently, and leave the token's record in the database. If a token as 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 correct role (token-admin). **Delete** should be used only when audit records are not required, such as when testing or importing tokens.

  Note: The **token-admin** role is by default granted 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.

6.2 Check the tokens that are registered to a user

To check which tokens are registered to a user, search for the user, then on the **2FA Mechanism** tab look in the **Status** column. The available roles and authentication methods have a green check mark.

Next to the **Value** identifier for the authentication method, there may be buttons like  . These can be used to activate or clear the associated token types.

Chapter 7: Configure the brokering service for third-party authentication servers

Echidna can work as a broker for an existing authentication server, such as RSA Authentication Manager.

In this situation, Echidna provides a single target for all authentication traffic for the enterprise, and it directs authentication requests to the appropriate verification point. Echidna handles some authentication requests itself, and it forwards other requests to the legacy server.

The organisation can retain its legacy authentication tokens at the same time as it introduces new mobile and open-standard OATH tokens. Because the legacy authentication server continues to respond to requests, there is minimal user or technology impact.

Before Echidna, the clients communicate with the legacy authentication server directly. After configuring Echidna to receive all authentication requests, the clients communicate directly with Echidna, and Echidna forwards requests as required.

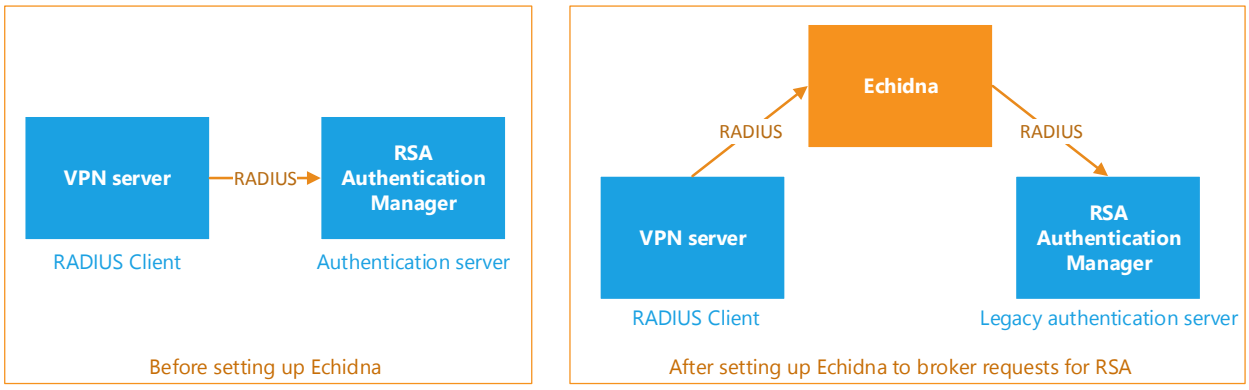


Figure 5: Example of brokering RADIUS requests for RSA Authentication Manager

This example show brokering with a RADIUS authentication server. Brokering also works with an authentication server that communicates via web services.

7.1 Configure Echidna to broker RADIUS authentication requests

1. Note the shared secret for the client. This is required in a later step.
2. In the Administration console, click **CONFIG**, then click **Authenticators**.
3. On the left under **Authenticators**, click **new**, then enter the following details about the new authenticator:
   a. Type a name for this authenticator.
   b. Select **Remote RADIUS** as the authenticator type.
   c. In the **Authentication Processes** section at the bottom, leave **authenticationProc** selected. For information about authentication processes, see the *Administration Reference*.
4. Enter the following information:
   - **User ID Link Attribute**: The user name that will be used for the remote RADIUS request. For example, `${storeName}/${uid}`.
   - **Remote RADIUS Server Host Name**: The hostname of the RADIUS server.
   - **Remote RADIUS Server Auth Port**: The RADIUS authentication port, which is usually 1812 or 1645.
   - **RADIUS Client Shared Secret**: The shared secret from Step 1.
5. Click **Update**, then Save the configuration (see page 16).
6. Verify the new configuration:
   a. As a user, log in to the Self Service console and set **Remote RADIUS** as the preferred login option.
   b. As the same user, attempt to log in to the RADIUS client.
   c. Check that the user is successfully authenticated by the authentication mechanism configured in the remote server.

   Note: If the only client is currently **localhost**, test Echidna by sending a request from the local computer only. Echidna will not accept authentication requests from other clients. If the configuration includes the details of actual clients, Echidna will accept authentication requests from them.

7.2 Configure Echidna to broker web services authentication requests

1. In the Administration console, click **CONFIG**, then click **Authenticators**.
2. On the left under **Authenticators**, click **new**.
3. Type a name, then select **Remote Web Service** as the authenticator type.
4. In the **Auth Attributes** section, add the user attributes required for the authentication mechanism to be enabled. If a user has one of these attributes, they will be permitted to use this authenticator.
5. In the **Remote Server Bind Credential** section, enter the following information:
   - **Username**: The service account name to connect to the remote authentication server.
   - **Password Protector**: Select the preferred password protector.
   - **Password**: The service account password to connect to the remote authentication server.
6. In the **Remote Generate Challenge Service Call** section, enter the following information:
   - **Service URL**: Enter the URL to be used for the service call, without any parameters. For example: `http://<remote server>:<web service port>/webUserAuth/authenticate`
   - **Method**: Specify the HTTP method to be used.
   - **HTTP Headers**: Click on the **Add** button next to **Params** and add the headers for the HTTP request. For example: **Key**=Host, **Value**=<hostname>:<web service port>
   - **HTTP Request Parameters**: Click on the Add button next to **Params** and add the parameters for the HTTP request. For example:
     - **Key**=username, **Value**=${username}
     - **Key**=domain, **Value**=${domain}
     - **Key**=credential, **Value**=${password}
7. In the **Remote Verify OTP Response Service Call** section, enter the following information:
   - **Service URL**
   - **Method**
   - **HTTP Headers**
   - **HTTP Request Parameters**
8. In the **Generate-Challenge Responses** section, specify the patterns to categorise the response from the **Remote Generate Challenge Service Call**.

   For example:
   - **Response Type**=VALID, **Match Type**=REGEX, **Pattern**=.*name=SUCCESS, type=ACCEPT.*
   - **Response Type**=INVALID, **Match Type**=REGEX, **Pattern**=.*name=INVALID_CREDS, type=REJECT.*

9. In the **Verify-Response Responses** section, specify the patterns to categorise the response from the **Remote Verify OTP Response Service Call**.
10. Click **Update**, then Save the configuration (see page 16).
11. Verify the new configuration:
    a. As a user, log in to the Self Service console and set **Remote Web Service** as the preferred login option.
    b. As the same user, attempt to log in to the web service client.
    c. Check that the user is successfully authenticated by the authentication mechanism configured in the remote server.

7.3 Update a client to authenticate to Echidna

When an existing client already authenticates to a legacy authentication server, it must be updated to authenticate to Echidna instead. Echidna will then forward authentication requests to the legacy server.

7.3.1 Update a web services client

To update a client that currently uses web services to communicate with a legacy authentication server, use the information in the *Echidna API Guide*.

7.3.2 Update a RADIUS client

To update a client that currently uses RADIUS to communicate with a legacy authentication server, update the following information:

- **IP address**: The IP address of Echidna
- **Shared secret**: The shared secret configured in Step 6b of Configure Echidna on page 14.

Chapter 8: Update Echidna Appliance

The following steps describe how to migrate an existing Echidna configuration to the new Echidna 15.1 appliance.

8.1 Create backup

1. Create a backup of the existing Echidna configuration and internal Echidna database, if it exists. Refer to the ***Backup and recovery*** section of the *Administration Reference for Echidna_15.1* for instructions. The internal Echidna database is backed up together with the configuration.
2. Create a backup of any external keystore files outside the Echidna configuration folder. This may be checked by opening the config.xml file and checking the *storePath* attributes of the *<keyStoreConfig>* elements.

   Ex.

```
<keyStoreConfig name="utilKs" storePath="utility.jks" type="jceks">
<keyStoreConfig name="trustKs" storePath="file:/usr/lib/jvm/java-7-
   openjdk-amd64/jre/lib/security/cacerts" type="jks">
```

   For the above example, `/usr/lib/jvm/java-7-openjdk-amd64/jre/lib/security/cacerts` needs to be backed up. `utility.jks` doesn't need to be backed up separately because it resides in the Echidna configuration folder.

8.2 Deploy Echidna Appliance

Deploy the new Echidna Appliance and configure the new virtual machine as described in Deploy the virtual appliance.

8.3 Restore configuration

1. Login to the new appliance and stop the Tomcat service.

2. Restore the backed up Echidna configuration and the internal Echidna database, if it exists.

   NOTE: If migrating from Echidna 14.1 SR3 and below, the following command needs to be executed on the Echidna configuration folder to give access to the Tomcat service:

   ```
   > chown -R tomcat7:tomcat7 *
   ```

   Also, in Echidna 14.1 GA, generic OATH tokens are not MACed even if the TEMAC box is checked in the configuration. This will create problems when migrating to Echidna 15.1. To work around this issue, uncheck the TEMAC box in the configuration for the generic OATH token authentication.

3. Restore the backed up external keystore files. If the location of these keystore files changed in the new appliance, the *storePath* in the corresponding *<keyStoreConfig>* elements of the config.xml file also needs to be updated.

4. Start the Tomcat service.

5. Open the Administration console and go to CONFIG > Crypto > serverIdentity. Modify the Key Alias and Subject DN to point to the correct hostname.

6. Refer to Update the SSL certificate for instructions on how to update the SSL certificate with the new hostname.

8.4 Request new license

1. Open the Administration console and go to CONFIG > License. Here you can see that the license type returns to evaluation since the current license file is invalid.
2. Delete any invalid hosts and license file.
3. Create a new license request and add the new license as described in Upgrade from the evaluation license.

8.5 Start Access Points

1. Each of the configured access points was stopped due to the invalidated license. After importing a valid license, go to CONFIG > Connectors and start the configured access points.

The new Echidna appliance is now ready to be used. The new installation may be tested by setting up tokens as described in Set up tokens.