SecureAuth RADIUS v1.0.1.X Installation Guide

Summary / Overview

SecureAuth is a technology leader providing access control to mobile, cloud, web, and network resources, serving over 10 million users worldwide. The SecureAuth IdP all-in-one is a completely scalable solution that manages and enforces access based on existing user entitlements. SecureAuth IdP seamlessly integrates with any device or application that supports RADIUS Authentication to provide strong two-factor authentication using One-Time-Token delivered via SecureAuth OTP application.

Purpose

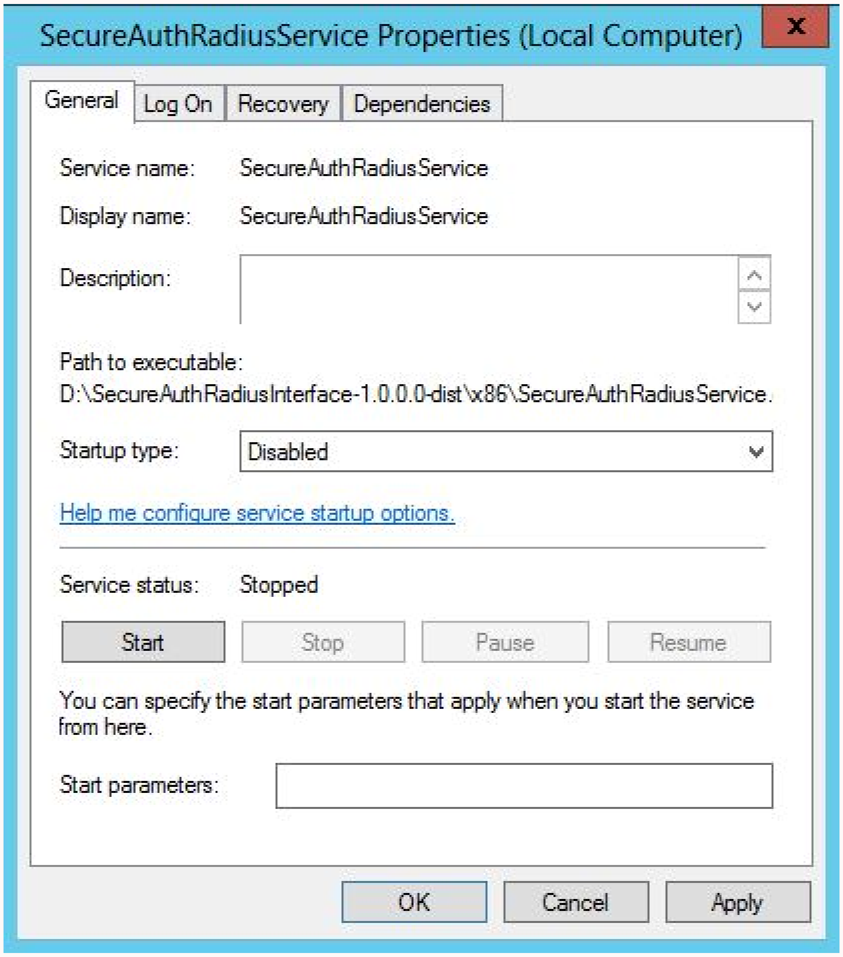
Use this document to install SecureAuth RADIUS v1.0.1.X on the SecureAuth server.

ⓘ This RADIUS service only supports PAP authentication. To use any other protocol, contact **SecureAuth Support**.

System Requirements

- SecureAuth IdP appliance: Windows Server 2003 and above
- JRE 1.7 32-bit
- UDP ports 1812 and 1813 open
- SecureAuth IdP realm 998 configured to deliver One Time Password (OTP) (https://docs.secureauth.com/x/1hXy)

If SecureAuth RADIUS 1.0.0.0 is installed on the SecureAuth IdP server, then please disable the existing Radius service before proceeding with the new installation to upgrade to this version



SecureAuthRadiusService Properties (Local Computer)

Service name: SecureAuthRadiusService

Display name: SecureAuthRadiusService

Description:

Path to executable:
D:\SecureAuthRadiusInterface-1.0.0.0-dist\x86\SecureAuthRadiusService.

Startup type: Disabled

Help me configure service startup options.

Service status: Stopped

You can specify the start parameters that apply when you start the service from here.

Start parameters:

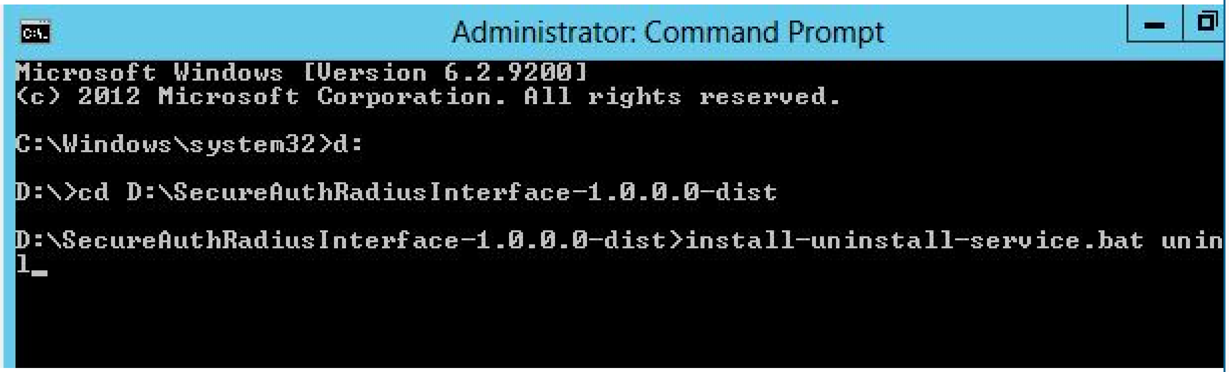
1. Disable the existing RADIUS installation:

Navigate to Service.msc, scroll to the service "SecureAuthRadiusService" – > Right click and Properties --> Startup type = disabled

2. Install and test the RADIUS server v1.0.1.X (**follow the Installation and Configuration steps in the doc below**)

Once completed, uninstall the previous SecureAuth RADIUS version via one of the two options presented below:

Option 1



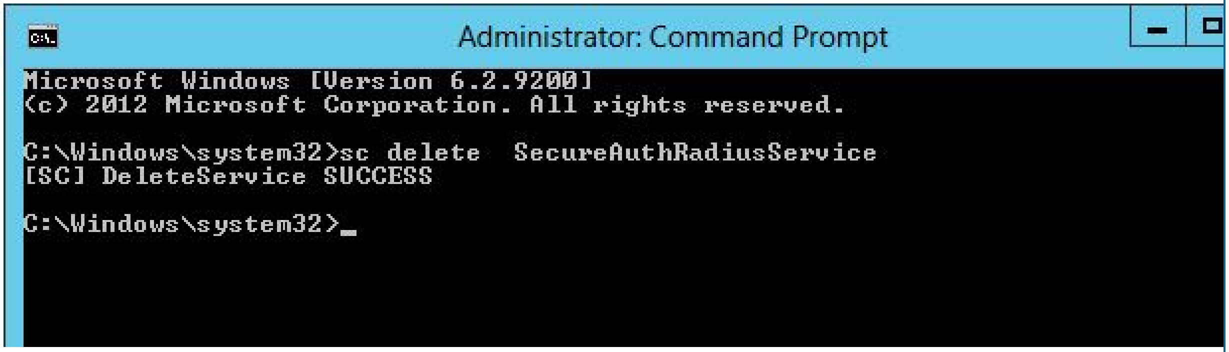
```
Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

C:\Windows\system32>d:

D:\>cd D:\SecureAuthRadiusInterface-1.0.0.0-dist

D:\SecureAuthRadiusInterface-1.0.0.0-dist>install-uninstall-service.bat unin
l_
```

1. Open the command prompt as administrator

2. Navigate to **D:\SecureAuthRadiusInterface-1.0.0.0-dist\**

3. Type **install-install-service.bat uninstall**

Option 2



```
Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

C:\Windows\system32>sc delete  SecureAuthRadiusService
[SC] DeleteService SUCCESS

C:\Windows\system32>_
```

1. Open the command prompt as administrator

2. Type after **sc delete**, **SecureAuthRadiusService**

Installation

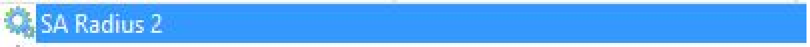
To Download RADIUS

1. Contact support to receive the link to download the **RADIUS** file.
2. Unzip the contents of the zip file **secureauth-radius-1.0.1.x-dist.zip** in the **D Drive** of the SecureAuth Server.
   This creates a folder **secureauth-radius-1.0.1.x-dist**.
3. Open a command prompt with administrator privileges and navigate to **D:\secureauth-radius-1.0.1.x-dist\secureauth-radius-1.0.1.x\bin** directory.
4. Run the **installWindowsService.bat** file.

5. Open **services.msc** and verify **SA Radius 2** service is installed.

6. Open the **SA Radius service** and set the logon rights to the current user.

7. Set the **SA Radius 2** service to **Automatic** and start the service.



To Configure RADIUS Service

1. In the **Windows explorer**, navigate to **D:\secureauth-radius-1.0.1.x-dist\secureauth-radius-1.0.1.x\webapp\WEB-INF\classes** and open **appliance.radius.properties** file in Notepad.

2. In the **radius.secureauth_endpoint** url, set the **IP address** or **FQDN** of your SecureAuth Appliance.

   Example

   radius.secureauth_endpoint:

   Format Example

   ```
   https\://localhost/secureauth998/oathWuserpassvalidate.ashx
   ```

   This is the secure endpoint of the SecureAuth Server that receives authentication requests for validation from the SecureAuth Radius Interface. It can verify the (username, password) and (username, otp) pairs. This needs to be up and running for the service to function correctly. The complete url should be specified here.

   a. **radius.shared_secret**
   This is the shared secret that the SecureAuth Radius Interface shares with its clients. The same needs to be defined on all the radius clients for this service. All clients share the same secret with this radius interface.

   Example radius.shared_secret=testing123

   b. **radius.clients**
   This is the list of client IP addresses, authentication type for each client and the client name.
   Each client definition is enclosed in square brackets ([]). And you may specify multiple client definitions. For multiple client definitions, place each definition within [] and separate each one with a comma.
   Each client definition has the following parts. All of the parts must be supplied:

   - **CLIENT IP ADDRESS:** This parameter specifies a valid IP address for the client
   - **AUTHENTICATION TYPE:** This specifies the authentication type for the client. It has to be one of the value described below.
     - **OTP_ONLY**: This only performs a username, OTP authentication. If successful, it returns an **`ACCESS_ACCEPT, else ACCESS_REJECT`** radius response is returned to the client.
     - **PASSWORD_ONLY:**This only performs username and password authentication. If successful, it returns and **`ACCESS_ACCEPT, else ACCESS_REJECT`** radius response is returned to the client.
     - **PASSWORD_AND_OTP:** This performs a 2-factor authentication for the client authentication requests. The first request performs a username, password authentication. If successful, it returns an **`ACCESS_CHALLENGE`** radius packet to prompt the end user for an OTP.
       The second request performs a second factor authentication for username and otp. If successful, an **`ACCESS_ACCEPT`** radius response is returned, **else** and **`ACCESS_REJECT`** radius response is returned to the client.
     - **OTP_AND_PASSWORD:** This performs a 2-factor authentication for the client authentication requests. The first request performs a username, OTP authentication. If successful, it returns an **`ACCESS_CHALLENGE`** radius packet to prompt the end user for an PASSWORD.
       The second request performs a second factor authentication for username and password. If successful, an **`ACCESS_ACCEPT`** radius response is returned, **else** and ACCESS_REJECT radius response is returned to the client.
     - **OTP_SLASH_PASSWORD:** The password is a single string of your Soft Token and Password separate by a slash "/"
   - **CLIENT NAME:** This currently is a free form alphanumeric string that is passed to the SecureAuth endpoint for additional client context. It is suggested that you use generic client names depending on the particular context like - VPN, VDI.

     Examples configurations for the property radius.clients
     radius.clients=[172.16.0.39, OTP_ONLY, VPN],[172.16.0.21, PASSWORD_AND_OTP, VDI],[172.16.0.1, OTP_ONLY, VPN]

   c. **radius.authport**
   This is the authentication port that the SecureAuth Radius Interface listens to. It defaults to the radius standard of 1812 and can be changed to another free port if needed. The same should be marked in the radius client during its configuration for this radius interface. This needs to be numeric.

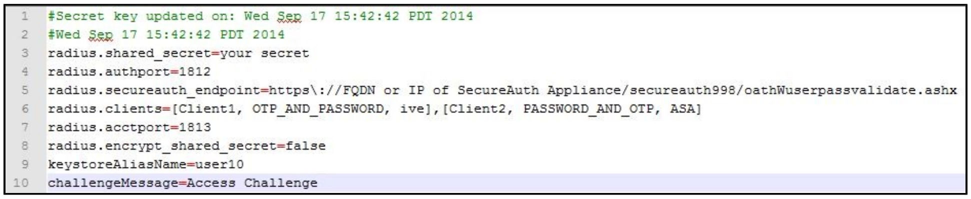
Example radius.authport=1812

d. **radius.acctport**

This is the accounting port that the SecureAuth Radius Interface listens to. It defaults to the radius standard of 1813 and can be changed to another free port if needed. The same should be marked in the radius client during its configurations for this radius interface. This needs to be numeric.

Example radius.acctport=1813

e. **ChallengeMessage:** For Radius authentication type like *OTP_AND_PASSWORD* and *PASSWORD_AND_OTP* where a user is prompted for Access-Challenge, you can Specify what text you want to display along with the Challenge.



```
#Secret key updated on: Wed Sep 17 15:42:42 PDT 2014
#Wed Sep 17 15:42:42 PDT 2014
radius.shared_secret=your secret
radius.authport=1812
radius.secureauth_endpoint=https\://FQDN or IP of SecureAuth Appliance/secureauth998/oathWuserpassvalidate.ashx
radius.clients=[Client1, OTP_AND_PASSWORD, ive],[Client2, PASSWORD_AND_OTP, ASA]
radius.acctport=1813
radius.encrypt_shared_secret=false
keystoreAliasName=user10
challengeMessage=Access Challenge
```

> If SecureAuth RADIUS 1.0.0.0 is installed on the SecureAuth server, then copy the relevant settings from the config (radius.config) file of the previous radius to the config (appliance.radius.properties) file of the new RADIUS installation
>
> **For example:**
>
> 1. radius.shared_secret
>
> 2. radius.clients

Encryption

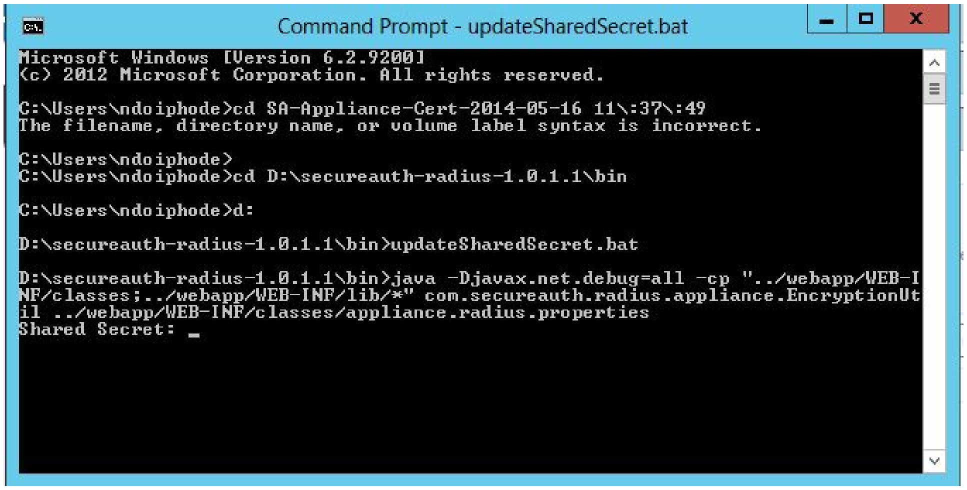
To Encrypt the RADIUS Shared Secret

1. Log into the SecureAuth appliance as a Local Admin
2. In the Radius config file (appliance.radius.properties) Set **radius.encrypt_shared_secret** to **true**.
   This performs encryption of the Radius Shared Secret. By default this value is set to false. To encrypt the shared secret set this value to true.
3. Open **mmc console** and export the certificate from the SecureAuth appliance (machine store) with private key and install it in the user store.

> **Note**
>
> The **Friendly Name** of this certificate or **issued to** name in case friendly name is absent.

4. Enter the friendly name of the certificate in step 2 for **keystoreAliasName** in the Radius config file.

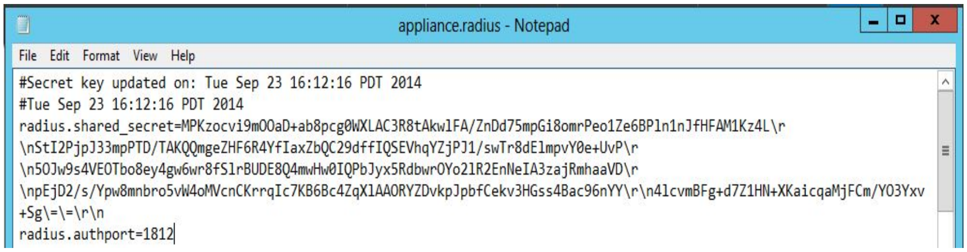
5. Open Command Prompt as administrator, go to `D:\secureauth-radius-1.0.1.x-dist\secureauth-radius-1.0.1.x\bin` folder, and run the **updateSharedSecret.bat** file.



```
Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

C:\Users\ndoiphode>cd SA-Appliance-Cert-2014-05-16 11\:37\:49
The filename, directory name, or volume label syntax is incorrect.

C:\Users\ndoiphode>
C:\Users\ndoiphode>cd D:\secureauth-radius-1.0.1.1\bin

C:\Users\ndoiphode>d:

D:\secureauth-radius-1.0.1.1\bin>updateSharedSecret.bat

D:\secureauth-radius-1.0.1.1\bin>java -Djavax.net.debug=all -cp "../webapp/WEB-I
NF/classes;../webapp/WEB-INF/lib/*" com.secureauth.radius.appliance.EncryptionUt
il ../webapp/WEB-INF/classes/appliance.radius.properties
Shared Secret: _
```

> **Note**
>
> The secret is not visible while you type it as a security feature. You can encrypt the radius secret as many times as you want.

6. After encrypting the secret, the radius secret in the radius config file (appliance.radius.properties) will look like the image below.



```
#Secret key updated on: Tue Sep 23 16:12:16 PDT 2014
#Tue Sep 23 16:12:16 PDT 2014
radius.shared_secret=MPKzocvi9mOOaD+ab8pcg0WXLAC3R8tAkwlFA/ZnDd75mpGi8omrPeo1Ze6BPln1nJfHFAM1Kz4L\r
\nStI2PjpJ33mpPTD/TAKQQmgeZHF6R4YfIaxZbQC29dffIQSEVhqYZjPJ1/swTr8dElmpvY0e+UvP\r
\n5OJw9s4VEOTbo8ey4gw6wr8fSlrBUDE8Q4mwHw0IQPbJyx5RdbwrOYo21R2EnNeIA3zajRmhaaVD\r
\npEjD2/s/Ypw8mnbro5vW4oMVcnCKrrqIc7KB6Bc4ZqX1AAORYZDvkpJpbfCekv3HGss4Bac96nYY\r\n4lcvmBFg+d7Z1HN+XKaicqaMjFCm/YO3Yxv
+Sg\=\=\r\n
radius.authport=1812
```

7. Logon the Radius Service with "Local Admin" credentials and restart the Service.

Uninstallation

To uninstall RADIUS

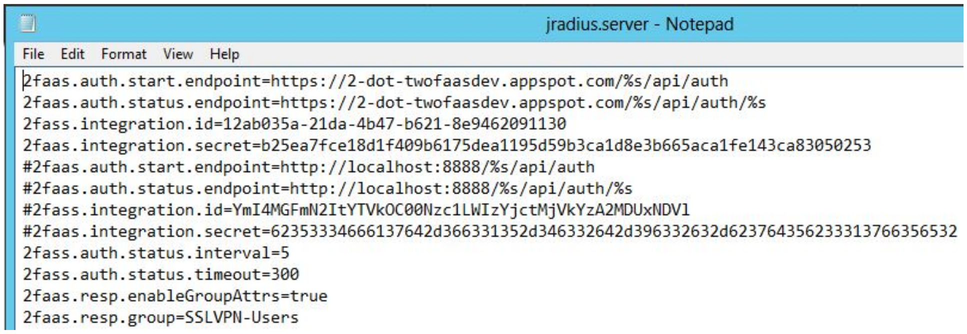
- Open a command prompt with administrator privileges and type in **`sc delete SecureAuthRadiusService2`**

Additional: ***Radius Attribute and User groups***

VPNs like WatchGuard Require Attribute 11 and Usergroup information to be sent back after the authentication. For this, you can enable group attribute and specify the user group in the **"jradius.server.properties"** file.

To make this change navigate to **D:\secureauth-radius-1.0.1.x-dist\secureauth-radius-1.0.1.x\webapp\WEB-INF\classes** and open the **jradius.server.properties** file.

1. Set **2faas.resp.enableGroupAttrs** to **true**.
   This enables the **GroupAttribute**.
2. Use **2faas.resp.group** to specify the User Group name required by Radius client in here. (By default, WatchGuard required **SSLVPN-Users** group).



```
2faas.auth.start.endpoint=https://2-dot-twofaasdev.appspot.com/%s/api/auth
2faas.auth.status.endpoint=https://2-dot-twofaasdev.appspot.com/%s/api/auth/%s
2fass.integration.id=12ab035a-21da-4b47-b621-8e9462091130
2faas.integration.secret=b25ea7fce18d1f409b6175dea1195d59b3ca1d8e3b665aca1fe143ca83050253
#2faas.auth.start.endpoint=http://localhost:8888/%s/api/auth
#2faas.auth.status.endpoint=http://localhost:8888/%s/api/auth/%s
#2fass.integration.id=YmI4MGFmN2ItYTVkOC00Nzc1LWIzYjctMjVkYzA2MDUxNDVl
#2faas.integration.secret=62353334666137642d366331352d346332642d396332632d623764356233313766356532
2fass.auth.status.interval=5
2fass.auth.status.timeout=300
2faas.resp.enableGroupAttrs=true
2faas.resp.group=SSLVPN-Users
```

Logging

The Radius logs can be found at the following location ***C:\Windows\System32\Logfiles\Apache***