TACACS.net™

Deployment Guide

Using Multi-Factor Authentication with Google Authenticator

Contents

1. Introduction

Authentication is considered to be one of the weakest links in computer security. More than 75% of computer security breaches are due to weak authentication[1]. More than 90% of user-chosen passwords can be found in hacker dictionaries[2], and an 8 character password can be cracked using brute force in less than 2 hours[3]. Passwords are also frequently compromised via phishing, malware, social engineering, or eavesdropping. Multi-factor authentication adds an additional layer of security to user logins. It works by requiring two or more of the following authentication types:

1) Something you know (like a password)
2) Something you have (such as a mobile device, private key, or smart card)
3) Something you are (such as a fingerprint or iris scan)

In some cases, a location (such as a GPS coordinate or IP address) can also be considered a factor, but this is not always recognized. The most common implementation of multi-factor authentication is something you know (password) combined with something you have (smart phone or key fob). Multi-factor authentication is required for organizations that process credit card transactions, health care information, or do business with some state and federal governments. Other regulations and industries also require multi-factor authentication or strongly recommend it as a best practice.

2. Google Authenticator

Google Authenticator is an open source one-time password implementation. Google has released the source code to allow it to be implemented in other software projects. Google has released compatible clients that are available for Android, iPhone/iPad, Windows, and Blackberry devices. It can also be integrated into scripts or programs used for automated device management or provisioning.

3. Counter-based vs. Time-based OTP

HMAC-based one-time passwords, also known as counter-based or event-based one-time passwords, use a counter or event timer to create unique one time passcodes that are incremented each time they are used. Once a passcode is used, it is no longer valid. When the server recognizes a passcode from the authorized list, it removes it and will only accept further passcodes in the list based on a look-ahead window set by the administrator. Time-based one-time passwords do not maintain a list of authorized passcodes, but verify a passcode based on the time it was generated. Because of this, TOTP requires an accurately synched time source on both the server and the client. If the time is not accurately synchronized, the clients will not be able to authenticate. Time-based one-time values only apply for a short period of time (30 seconds). HTOP passwords do not have this restriction. They can remain valid for a much longer time period. This makes HOTP easier to use, but TOTP is generally considered more secure. TACACS.net supports both counter-based and time-based algorithms.

4. System Requirements

Multi-Factor authentication with Google Authenticator requires TACACS.net version 1.3 or better. To see the version of TACACS.net you are running, right-click tacplus.exe and look for the Product version under the Details tab. The current version can be downloaded from www.TACACS.net.

---

[1] Source: Verizon 2013 Data Breach Investigations Report
http://www.verizonenterprise.com/resources/reports/rp_data-breach-investigations-report-2013_en_xg.pdf

[2] Source: 10,000 Top Passwords
http://xato.net/passwords/more-top-worst-passwords/

[3] Source: How to create a 'super password'
http://articles.cnn.com/2010-08-20/tech/super.passwords_1_passwords-character-websites?_s=PM:TECH

5. Configuring TACACS.net

5.1. authentication.xml

TACACS.net allows you to enable multi factor authentication based on User Group membership. This makes it easier to test and troubleshoot and enables the Administrator to choose with groups should use the higher security settings. For best results, you should start a new group to test with to ensure that multi-factor authentication is working properly. After you have confirmed that it is working, you can update existing groups. The following example uses a File group called OTPTest. An additional line for MFAProvider is added under the AuthenticationType, and the value is GoogleOTP.

```
<!--This group is used for OTP testing -->
<UserGroup>
   <Name>OTPTest</Name>
      <AuthenticationType>File</AuthenticationType>
      <MFAProvider>GoogleOTP</MFAProvider>
         <Users>
            <User>
                <Name>otptest</Name>
                <LoginPassword ClearText="mypass" DES=""> </LoginPassword>
                <EnablePassword ClearText="" DES=""></EnablePassword>
                <CHAPPassword ClearText="" DES=""> </CHAPPassword>
                <OutboundPassword ClearText="" DES=""> </OutboundPassword>
            </User>
         </Users>
</UserGroup>
```

Figure 1

5.2. authorization.xml

Since a new User Group has been added, there needs to be a corresponding Authorization Group. The following example will only allow the OTPTest group to run show commands.

```
<!--This group is used for OTP testing -->
<Authorization>
   <UserGroups>
      <UserGroup>OTPTest</UserGroup>
   </UserGroups>
   <ClientGroups>
      <ClientGroup>*</ClientGroup>
   </ClientGroups>
   <Shell>
      <Permit>.*show.*</Permit>
      <Deny>.*</Deny>
   </Shell>
</Authorization>
```

Figure 2

5.3. googleotp.xml

Googleotp.xml is the configuration file for Google Authenticator. It can be found with the other configuration files in Start > TACACS.net> Configuration. In most cases, the default settings should be used, but they can be changed if necessary.

5.3.1. Look-ahead window

The look-ahead window is used in HOTP in order to specify a number of OTP values that are allowed after the next expected OTP value. This number should be set as low as possible, while still ensuring that usability is not impacted. The default setting is 50.

5.3.2. Delay window

The delay window is used in TOTP to set the amount of acceptable time delay between the receiving and transmitting time. RFC 6238 recommends that at most, one time step is allowed as the delay. The time step is 30 seconds and the delay window defaults to 90 which allows one time step before and one time-step beyond the 30 second time step.

5.3.3. Replay Prevention

Replay Prevention is an optional setting that will prevent a time-based OTP value from being used more than once. This can prevent replay and man-in-the-middle attacks, but will restrict the user to no more than one login every 30 seconds. This setting is disabled by default.

5.4. tacplus.xml

5.4.1. OTP Separator

The OTP Separator is a character used to separate the Username and the OTP Value during login. The default is the asterisk (*) character, but this can be modified if needed.

6. Generating Shared Secrets

The Google Authenticator Shared Secret Manager is used to generate and manage Shared Secrets. These Shared Secrets are entered into the client device to activate access.

Use the –c or –t switches to specify counter-based or time-based OTP, respectively. If neither is specified, the generator will default to time-based OTP.

```
>gass generate –t –u otptest –d "used for OTP testing"
```

Figure 3

This command will generate the following output:

```
Your new shared secret is: JM2EOVRZHBIUQSSH
QR Scan URL:
http://chart.apis.google.com/chart?cht=qr&chs=200x200&chl=otpauth%3A%2F%2Ftotp%2Fotpte
st%3Fsecret%3DJM2EOVRZHBIUQSSH
Your current OTP value is: 507567
Your emergency scratch codes are:
77013514
95248749
07536328
05084859
09672508
```

Figure 4

6.1. Username

The username (-u) is used to specify the username that will be used with this Shared Secret.

6.2. Descriptor

A descriptor (-d) is an optional attribute that can be used to add an additional description to who is using this Shared Secret. This is displayed when you use >gass show (see below).

6.3. Current OTP Value

This is the current value of the generated OTP code. In case of Time based OTP, it is the time based value of current OTP. In case of counter based OTP, it is the value of the OTP for counter value of 1. This verifies that the Shared Secret was entered correctly and the user can begin using it immediately.

6.4. Scratch Codes

Scratch codes are used if a user does not have his/her Google Authenticator client available and needs to substitute an OTP value. These codes can be given to the user to write down and put in a safe place in case they are needed. The Administrator may choose not to give these scratch codes to the user if he/she does not want to user to use them.

7. Replicating Shared Secrets

If you are running multiple TACACS+ servers, you will need the Shared Secrets that you have generated on one server to be available to other servers so that it can also authenticate users using Multi-Factor Authentication. This information is in the configuration directory in a file called gass.db. This file needs to be copied to all servers every time a change is made.

8. Authenticating Users

When logging into a router, switch or other TACACS+ enabled device, you will need to append the OTP Value to the username after the separator. For example, if your username were 'otptest', your password were 'mypass', and your OTP Value were '123456', your login would look like this:

```
Username: otptest*123456
Password: mypass
```

Figure 5

When the server sees the above login, it will first verify the username and OTP Value with Google Authenticator. If verified, the username and password will be passed to the authentication database to be authenticated. If either step fails, the user will not be authenticated.

9. Troubleshooting

9.1. Show Shared Secrets

The show command is used to display the currently registered Shared Secrets.

```
>gass show

KBDDIRCRHBMEGM2H
JSmith
Joe Smith, NOC Engineer
[Time based]
Used on 0 client device(s)
Created on 7/28/2013 7:31:40 PM
```

Figure 6

9.2. Delete Shared Secrets

The kill command is used to remove unneeded Shared Secrets. Use '`kill *`' to delete all Shared Secrets.

```
>gass kill 6171819202122232

The Google Authenticator shared secret has been removed and is no longer authorized.

>gass kill *
```

```
All Google Authenticator shared secrets have been removed and are no longer
authorized.
```

Figure 7

9.3. Verify Shared Secrets

Verify is used to check the validity of an OTP value. Use the switch –s to specify the Shared Secret and –o to specify the OTP Value.

```
>gass verify  -s  6171819202122232 -o 123456
 OTP value is valid.
```

Figure 8

9.4. Logging

Google Authenticator OTP events will be logged in Syslog at the Warning level and below (warning, debug, info, etc.). These logs will tell the Administrator if there are errors with Google Authenticator OTP Values.

10. Additional References

- TACACS.net - Free TACACS+ Server for Windows
  http://www.TACACS.net
- Google Authenticator
  http://en.wikipedia.org/wiki/Google_Authenticator
- Google Authenticator client for Android
  http://play.google.com/
- Google Authenticator for iPhone, iPod, and iPad
  http://www.apple.com
- Google Authenticator for Blackberry
  http://m.google.com/authenticator
- Google Authenticator for Windows PCs
  http://www.toms-world.org/blog/google_authenticator
- HMAC-Based One-Time Password Algorithm
  http://tools.ietf.org/html/rfc4226
- Time-Based One-Time Password Algorithm
  http://tools.ietf.org/html/rfc6238