Enhancing RADIUS based multifactor-factor authentication systems with RESTful API for self-service enrolment

HUSEYNOV, Emin, SEIGNEUR, Jean-Marc

Abstract
Two-factor authentication can significantly reduce risks of compromised accounts by protecting from the majority of password-leak based attacks. RADIUS protocol is a popular method of providing the second-factor authentication. This paper presents an innovative approach to enrolling users in RADIUS protocol based two-factor authentication. It describes a RESTful API, which complements the standard RADIUS protocol by adding the possibility of self-service second-factor enrollment. It also demonstrates a proof of concept RADIUS appliance and the web-based management interface as well as examples of API integration with a number of products.

Reference
HUSEYNOV, Emin, SEIGNEUR, Jean-Marc. Enhancing RADIUS based multifactor-factor authentication systems with RESTful API for self-service enrolment. Moscow, Russia : IEEE Xplore / AICT, 2017

Available at:
http://archive-ouverte.unige.ch/unige:101282

Disclaimer: layout of this document may differ from the published version.
Enhancing RADIUS based multifactor-factor authentication systems with RESTful API for self-service enrolment

Emin Huseynov
Faculté des Sciences de la Société
University of Geneva
Geneva, Switzerland
emin@huseynov.com

Jean-Marc Seigneur
CUI, ISS & Medi@LAB, Faculté des Sciences de la Société
University of Geneva
Geneva, Switzerland
seigneuj@gmail.com

Abstract— Two-factor authentication can significantly reduce risks of compromised accounts by protecting from the majority of password-leak based attacks. RADIUS protocol is a popular method of providing the second factor authentication. This paper presents an innovative approach to enrolling users in RADIUS protocol based two-factor authentication. It describes a RESTful API, which complements the standard RADIUS protocol by adding the possibility of self-service second factor enrollment. It also demonstrates a proof of concept RADIUS appliance and the web based management interface as well as examples of API integration with a number of products.

Index Terms— digital identification, identity management, authentication, authorization, privacy, context-based authentication, strong security, RESTful services, RADIUS authentication.

I. INTRODUCTION

Majority of modern two-factor authentication solutions use standalone hardware or software tokens that generate one-time passwords (OTP) to be used as the second factor for the authentication process [1]. Many systems use separate and in some cases, isolated, Remote Access Dial In User Service (RADIUS) [12] based systems to serve the authentication of the second factor only. RADIUS is based on simple transaction between a client and server. To ensure the secure communication channel between the client and the server, both are configured with a shared secret key. This secret key is used to cryptographically sign all messages in both directions to ensure integrity. RADIUS server keeps the attributes required to calculate current second factor for each user enrolled and when requests is made, it calculates the second factor and compares it with user provided data. If these values match, the authentication is accepted by sending an “Accept” response.

Although more and more new context-aware authentication means are invented, legacy solutions such as RADIUS can benefit from these new solutions. The solution presented in this paper is based on adding a registration or self-enrollment functionality to standard RADIUS systems by introducing an additional REST Web API library. Representational state transfer (REST) or RESTful Web services are methods of providing command execution and data exchange between computer systems on the network. RESTful web services are based on standard HTTP, or preferably, HTTPS protocol [2].

This paper will review how RADIUS protocol is used in two-factor authentication systems as well as architectural layout of such systems. Then, the Section II will continue with a summary of limitations of using RADIUS as the second-factor provider is reviewed. Section III provides a theoretical model of implementing self-service enrollment for two-factor authentication by complementing the standard RADIUS authentication protocol with RESTful based self-service registration service. Section IV will present a proof of concept implementation of a RADIUS service with examples of integration. Section V will provide a conclusion on the proposed model and its implementation as well as a brief security analysis.

II. REVIEW OF EXISTING IMPLEMENTATIONS

The primary goal for deploying two-factor authentication mechanisms is to increase the security of traditional identity stores such as Lightweight Directory Access Protocol (LDAP) [3], RADIUS or Active Directory (AD) [4]. Even though, both factors (e.g. user password and one-time passwords) can technically be verified using one single store, industry’s best practices recommend to separate first and second factor stores from each other. In Table I we have reviewed a number of products allowing enabling second-factor authentication and discovered that in the majority of cases, RADIUS servers are recommended as the identity stores for second factor only. All reviewed implementations use one-time passwords as the second factor based on time-based one time password (TOTP) [5] algorithm.

<table>
<thead>
<tr>
<th>Product</th>
<th>Primary identity stores</th>
<th>Second-factor-ready identity stores</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMware Horizon 7</td>
<td>Active Directory</td>
<td>RSA SecurID, RADIUS</td>
</tr>
<tr>
<td>Citrix Netscaler 12</td>
<td>LDAP, RADIUS</td>
<td>LDAP, RADIUS</td>
</tr>
<tr>
<td>Fortinet Fortigate VPN</td>
<td>LDAP</td>
<td>RADIUS</td>
</tr>
<tr>
<td>Microsoft Network Policy Server</td>
<td>RADIUS</td>
<td>RADIUS</td>
</tr>
</tbody>
</table>
To illustrate the authentication flow, we will use an example of configuration, where LDAP is used as the primary identity store and RADIUS is used as the secondary identity store. As shown on Figure 1, the user credentials requested by the login server consists of three fields: the username, first password (LDAP user password) and the second password. The authentication flow is happening in the following sequence:

1. User credentials are sent to the identification broker
2. Identity broker sends credentials to the identity stores
   2.1. Identity broker sends the username and the first password to the LDAP Server
   2.2. Identity broker sends the username and the second password to the RADIUS Server
3. Identity broker receives the response from the identity stores
   3.1. Identity broker receives the response from the LDAP Server
   3.2. Identity broker receives the response from the RADIUS Server
4. Identity broker allows or denies access depending on the responses received from LDAP and RADIUS.

Figure 1. An example of authentication flow with LDAP and RADIUS identity stores

All reviewed solutions are relying on pre-existing database of users in both identity stores they rely upon. While it is inevitable for the primary user base as a part of user provisioning process, enabling the second factor by administrators is introducing additional negative user experience.

Use case: Migration from classic to multifactor authentication

Let us review a use case with an existing infrastructure in place, where the goal is to enable second factor authentication. While there is no or minimum change in the primary authentication source, enabling second factor using existing

RADIUS solutions for users will bring the following inconveniences.

Mobile application enrollment

Assuming that a mobile application on a user owned (i.e. personal) smartphone is used as a second factor password generator; an administrator with physical presence of the user must complete the enrollment process. Alternatives to this method (such as sending the secret key by alternative channels, such as email) introduce additional security risks.

Hardware token provisioning

In case a hardware token is used as the second factor password generator, the transition needs to be coordinated with the end users. In addition, it complicates the preparatory phase and the logistics of the process; i.e. a database with secret keys and hardware token assignment needs to be pre-populated before the migration.

User migration and coexistence

Migration from single factor to two-factor authentication for a large user base is not possible with the reviewed RADIUS based solution. This means additional negative user experience as well as more resources for the migration. The main reason for this is that any user is obliged to enter a correct second factor password in order to get access; this also makes the coexistence of both authentication methods impossible: i.e. all users have to be one-factor or two-factor and these two types cannot coexist.

III. THEORETICAL MODEL OF SELF-SERVICE ENROLLMENT FOR RADIUS BASED TWO-FACTOR AUTHENTICATION

To address the issues given in the previous section, I will first describe a theoretical model of a RADIUS appliance that allows users to enroll the second factor without intervention of administrators.

Initial login concept

Prior to everything else, a user without any record in the secondary identity store would normally not be able to login. To overcome this restriction, RADIUS server should allow a number of initial logins with empty or wrong second factor. The system needs to keep a log of such logins to keep the security at the acceptable level. With this method implemented, a user without a second factor record created is able to login and access the system using the first factor only (i.e. username and password). Upon successful login, where possible, the end system detects that the second factor record is missing and prompts or forces the user to enroll to the secondary identity store. In the cases where showing a prompt or informational message is not possible, users shall be informed and instructed to do so using other methods (i.e. a message on the login page, instructions sent by email etc.). As per authentication flow of the initial login shown on Figure 2, the system (RADIUS server), in addition to regular user records database, will also keep history of login attempts of users without user records and allow access on first N attempts. For security purposes, the value of N needs to be kept as low as possible: ideally, N=1:

1 Identity broker shown in this list as well as in Figure 1 is a logical component and depending on the implementation may or may not be implemented as a separate instance.
only to provide users without user records with one single login in order to allow access the self-enrollment procedure.

**Self-service enrollment concept**

After the initial login has been allowed, the users are prompted to enroll for the second factor authentication. This paper proposed 2 methods of second factor: hardware tokens and software tokens. For both methods, the self-enrollment is based on RESTful API, so the user shall launch a web page which will in its turn reach out to the RADIUS appliance and send the current users username (and the serial number in case of the hardware token method). RADIUS appliance will create a user record containing the user name and the TOTP [5] secret hash, which will be randomly generated (for software tokens) or retrieved from hardware token hashes database based on the serial number provided. In case of software tokens, the generated hash will be returned back via RESTful API and displayed in a QR code format to the user in order to complete the software token enrolment. A structural diagram of this process is shown on Figure 3.

**IV. PROOF OF CONCEPT**

**TOTPRadius**

The proof concept of self-enrollment RESTful API for two-factor authentication has been implemented for Citrix Netscaler and Citrix Storefront integration. Usually, two-factor authentication on the Netscaler is integrated based on Radius where usernames are the same as usernames in Active Directory or LDAP. When integrated with Citrix Storefront, Netscaler acts as a reverse proxy and Storefront serves as a web server. Therefore the API calls between Netscaler and Storefront can be run directly over HTTPS. The implementation of the API is based on a virtual appliance code-named TOTPRadius that operates as a standard RADIUS server, but instead of standard static passwords verifies TOTP based one-time passwords. The network diagram of the solution is shown on Figure 4.
Self-enrollment API

This proof of concept is only accepting enrollment of software tokens (such as Google Authenticator [6] or Token2 Mobile OTP [7]). The RESTful API of TOTPRadius looks like shown below:

https://[FQDN]/createuser?api=[api_key]&user=[username]

Where:
- **FQDN** – is the hostname or IP address of the RADIUS server
- **api_key** – the API key to serve as an additional protection factor. The API key should match the key recorded in the server settings
- **user** – username of the current user

Upon successful execution of the API call, the response contains the following information:
- **text** – response text notifying about successful execution of the API call
- **hash** – the secret hash key generated and assigned to the user
- **hashqr** – hash key in the format of base64 data of the PNG image containing the QR code for creating TOTP profile on the mobile application

Implementation of Proof of Concept

The proof of concept described in this paper has been implemented and is available to be tested as a part of Token2 project. The RADIUS server (TOTPRadius) [8] is a virtual appliance running on Ubuntu Linux 17.0 and based on FreeRadius 3.0. Netscaler/Storefront integration based on the self-enrollment API is also presented by Token2.

V. SECURITY ANALYSIS AND CONCLUSION

As the security of RADIUS protocol has been already reviewed in a number of papers [9, 10], this security analysis is covering only the new components of the solution, namely, the self-enrollment part and the initial login procedure.

API call initiation

Depending on the implementation, the web application initiating the API call may present an additional security risk, however, in our proof of concept, the API is initiated from within Citrix Storefront, which is protected by the authentication mechanisms, thus it is not applicable. In theory, this could be a potential vector of attack; therefore, when implementing, it the following is considered to be the safe technique to run API calls to the RADIUS server:
- The username of the current user should be securely identified
- The API call should only be executed on the server side, and under no circumstances the API Key should be visible to the end-users.

API Endpoint security

As described earlier, the API is intended to be reachable over HTTPS. This ensures the data is only accessible from the client and service endpoints. It is also important that the SSL certificate is setup correctly, and the initiator script is correctly verifying the validity of the certificate [11]. Additionally, an API key is being sent and compared with the key recorded in appliance settings. Also, the API endpoint may be configured to respond only to a set of IP addresses or subnets.

Conclusion

Enabling a self-service feature for users to manage their own devices would significantly save on training and support time. The RESTful API concept introduces in this paper is an attempt to enable self-service enrollment keeping the whole concept of RADIUS based two-factor authentication, as it is a de-facto industry standard. The proof of concept used as an example, Citrix Netscaler and StoreFront based integration, proves that the concept can be integrated with minimum effort and maximize positive user experience.

REFERENCES


