



RedTeam
SECURITY CONSULTING

RedTeam Security
214 4th Street E.
Suite #140
St. Paul, MN 55101

Contact
info@redteamsecure.com
www.redteamsecure.com
612-234-7848

Application Penetration Test

Adeptia

October 14, 2021

Prepared by:

Kyle Brennis

RedTeam Security Corporation
214 4th Street E., Suite #140
St. Paul, MN 55101
P: 612-234-7848 E: info@redteamsecure.com

Date Issued:
December 3, 2021

Report ID: 1039

Scope

Target(s)

The scope of the test included the following in-scope information assets:

acesectest.adeptia.com

Roles tested include: Admin, Business, Partner and View-Only

Control(s)

The in-scope information assets were measured against the following controls:

- Open Web Application Security Project (OWASP)
- Open Web Application API Security Project (OWASP)
- Penetration Testing Execution Standard (PTES)

Timetable

The following testing timetable is shown below:

- Test Start: 10/11/21
- Test End: 10/14/21

Executive Summary

Overview

RedTeam Security has adopted an industry-standard approach toward security assessments. This approach is used in all our assessments and provides our clients with real-world risks that take into account a number of factors ranging from: Skill Level, Motive, Ease of Exploit to Financial and Reputational Damage. Our comprehensive approach ensures that our clients' vulnerabilities are represented by their true real-world likelihood and potential impact to their business.

RedTeam Security conducted an Application Penetration Test against the organization using a methodical and standardized approach. The objective of the assessment was to measure the security posture of the in-scope assets and identify any deviating vulnerabilities by measuring them against industry-adopted controls. For more information about our approach and methodology, please see [Appendix A](#).

Important findings from the assessment were communicated to management either during or following the assessment as appropriate based on the nature and risk level of the finding. All of our findings are explained in detail in the Findings section of this report. These findings are aligned with industry best practices at the time of report generation.

Summary

RedTeam Security conducted an application penetration test for Adeptia of Adeptia Connect with the purpose of assessing the security posture of the company's web application in the test environment. This testing utilizes industry standard methodologies, as well as manual and automated techniques, to identify security vulnerabilities and assess the risk presented by these findings. As a result of testing, nine (9) vulnerabilities were identified. Two (2) findings were rated High-Severity, four (4) were rated Medium-Severity, and three (3) were low-risk.

High Severity Vulnerabilities

The first High Severity vulnerability is an Authorization Bypass that results from a lack of server-side checks to verify that a request is permitted for a user of a given role. The server frequently processes authenticated requests without checking. Some of the examples of this vulnerability that would be the most attractive to a malicious actor could allow an authenticated user to make themselves a sys-admin user, assign themselves to any partner/company, and change passwords and emails for any accounts in the application. If sensitive information existed, it would also allow access to that. The second High Severity vulnerability is directly related to the first, and involves an endpoint that any authenticated user can access in order to change any user's password. This vulnerability is enhanced by the fact that the Access Token used to allow the password change can actually be expired. As an example, a former employee could therefore exploit this to gain access to accounts by using a saved and stored token value.

Medium Severity Vulnerabilities

The first Medium Severity vulnerability is for Insufficient Session Expiration, which is related to the above-mentioned password-change vulnerability. Passwords can be changed while authenticating with an expired Access Token. The second Medium Severity vulnerability, Brute Force Requests Allowed, is also related to the password-change functionality. When submitting a password-change request, the application asks the user to enter the existing password. As the user types, the application submits requests to the server to check if the password is valid, and the server responds "true" or "false." This response can be exploited by an authenticated

user to rapidly guess passwords for other users. The server response will inform them of a correct guess. The third Medium Severity finding stems from a separate avenue for Unverified Password Change. Higher privileged users can use the UI to change the registered email for any other user. This means they can change any user's email to their own and submit a password-reset request, effectively taking over the account and also negatively impacting accountability of otherwise separate users. The final Medium Severity finding is for Weak Passwords. The password policy requires only eight (8) characters minimum. RedTeam recommends a minimum of no less than 15 characters.

Low Severity Vulnerabilities

The first Low Severity vulnerability is for Concurrent Sessions Allowed. The risk here is primarily that if an account was compromised, say, by a password-guessing attack, then the legitimate, logged-in user would not be alerted when an unauthorized third party also logged into their account from a different IP. The remaining two Low Severity vulnerabilities are the result of insecure cryptographic configurations that affect the confidentiality of encrypted traffic.

RedTeam Security would like to encourage Adeptia to review the findings contained within this report and to use the information to develop remediation strategies which can help ensure the security and integrity of corporate assets and information.

Update: Nov. 30, 2021

RedTeam retested all findings to validate remediation efforts. Six (6) of the nine (9) findings were remediated. Both of the High Severity findings, Authorization Bypass, and Unverified Password Change, were remediated.

Two (2) of the four (4) Medium Severity findings were remediated. RedTeam was unable to test the finding regarding an Unverified Password change over email, because the email address of the test accounts could not be changed. As such the status of that finding cannot be changed to remediated. The other Medium Severity finding that was not remediated was Weak Passwords. The application still allows a password with fewer characters than is recommended, though the severity was slightly lowered due to the remediation of another finding. RedTeam was also informed by Adeptia that the password length and complexity was an application requirement, and thus would not be remediated.

Two (2) of the three (3) Low Severity findings were remediated. RedTeam was informed by Adeptia that concurrent logon sessions was an application requirement, and thus would not be remediated.

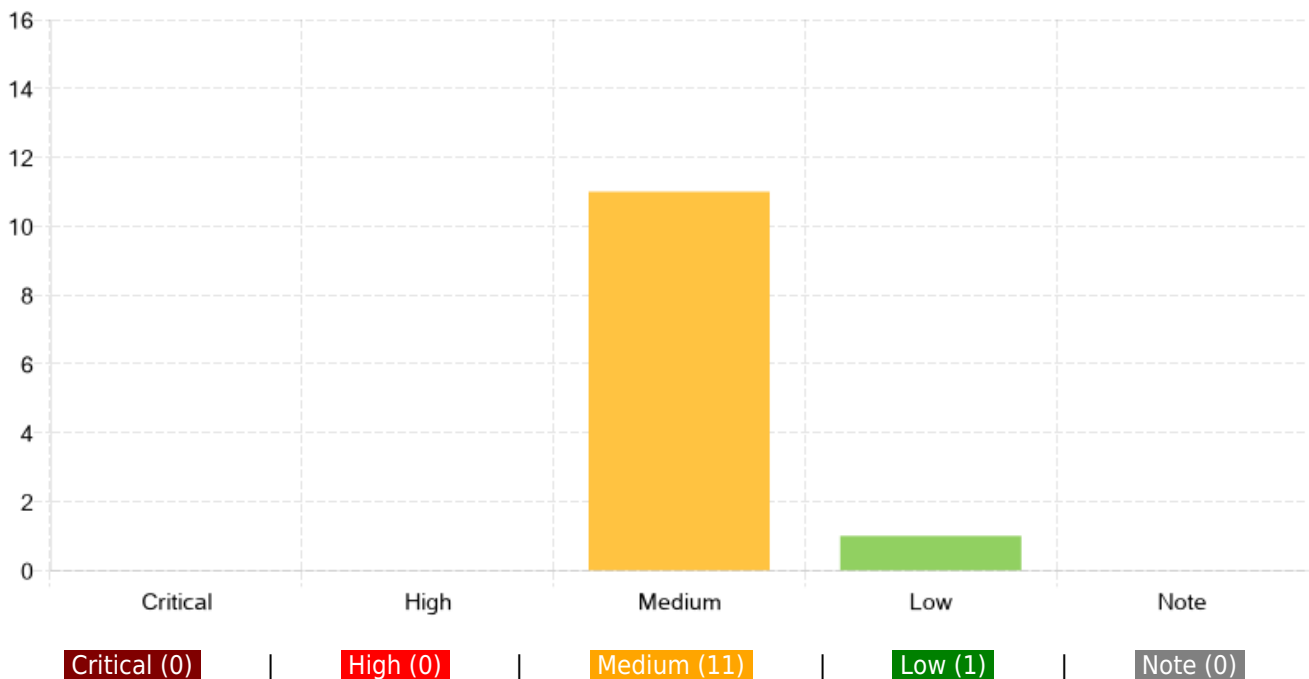
Update: Dec 1, 2021

After reviewing the Medium Severity finding regarding Unverified Password Change, RedTeam found that preventing an email change remediated this finding.

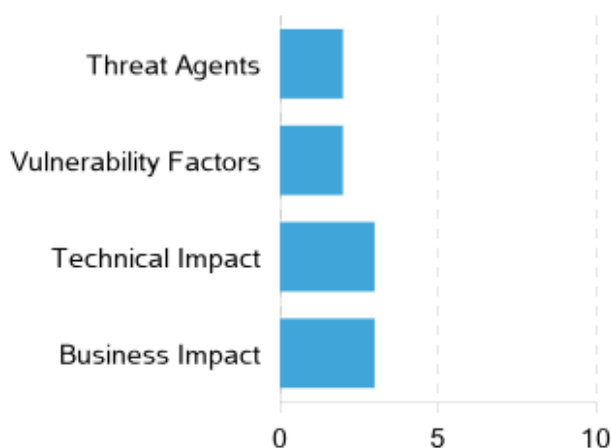
Vulnerability Summary

The charts below are designed to provide a quick snapshot of the assessment. For information regarding risk ratings, please see [Appendix B](#). Otherwise, for vulnerabilities as a result of this assessment, please see the Findings section.

Total Vulnerabilities by Rating



Average by Risk Factor



Average Overall Rating



Vulnerability Summary

Quick View

The table below is designed to provide a quick view of all the identified findings and their respective risk ratings. Please see the following section for a detailed listing of the identified findings.

For information regarding our risk rating methodology, please see [Appendix B](#).

#	Finding Title	Instances	Rating
1.	[REMIATED] Authorization Bypass	11	High (7)
2.	[REMIATED] Unverified Password Change	11	High (6.25)
3.	[REMIATED] Insufficient Session Expiration	11	Medium (5.5)
4.	[REMIATED] Unverified Password Change	11	Medium (4.25)
5.	Weak Password	11	Medium (4)
6.	[REMIATED] Brute Force Requests Allowed	11	Medium (3.5)
7.	[REMIATED] HTTP Strict Transport Security Not Enforced	1	Low (2.75)
8.	[REMIATED] 3DES Ciphers Supported	1	Low (2.25)
9.	Concurrent Login Sessions Allowed	1	Low (1)

Total Instances: 12

Finding(s)

1. Authorization Bypass | High (7)

[THIS FINDING HAS BEEN REMEDIATED]

Description:

An Authorization Bypass occurs when an application fails to prevent a user from taking actions that the user is not authorized to take. There are a wide variety of vectors through which these vulnerabilities can be found and exploited. The discovery, exploitation and impact of Authorization Bypass vulnerabilities is highly dependent upon the functionality of the application.

Impact:

The impact of Authorization Bypass vulnerabilities is highly dependent upon the functionality of the application. Impact can range from gaining access to non-critical functionality, to unauthorized access to administrative functionality and compromise of PII.

Test(s) Conducted:

Testing for Authorization Bypass vulnerabilities involves the identification of permissions limitations for the user role allocated for testing. By identifying the process through which permissions are limited within an application, a tester will then attempt to manipulate these processes to elevate or gain unauthorized privileges within an application. Such manipulation may include changing ID parameters or directly visiting parts of the application intended only for other users.

Finding Comments:

A lower privilege user (for example, a user with the view-only role) can escalate privileges to sys-admin by intercepting and tampering with a profileUpdate request before it is sent to the server. The server checks for Authentication in the form of a valid Access Token, but there is a lack of server-side checks for valid *Authorization* according to the application's role-system. The number of instances is set to 11 because there are 11 users in the testing environment, but in other environments this vulnerability will likely affect any user who can make POST requests to change any information about their profile.

This vulnerability essentially allows any authenticated user of the application to exercise the privileges of a sys-admin user. It is likewise possible for a user to change their own partner-id and company this way. This could be used to compromise users of other companies, take over accounts, or to create DoS conditions for users of the application. If sensitive information exists within any users accounts, then that information could be compromised. Escalating privileges by changing a user's role is one of the first and most obvious targets for an attacker. However, the lack of server-side authorization checks in this case is indicative of a larger potential issue. There are likely more Unauthorized API calls that can be made than those that RedTeam was able to identify during the limited time-span of the test.

RedTeam recommends reviewing the application and ensuring that all of the role-based authorization checks that are occurring in the UI are also completed by the server before it processes requests.

It is also possible to achieve at least some of the same bypass simply by tampering via the browser console.

Update: Nov. 30, 2021

RedTeam retested this finding and observed that manipulating the role id parameter in a request to the previously affected API endpoint, no longer changed the account to an administrator.

Recommendations:

Privileges and permissions within an application must be limited both through client-side controls and server-side validations. It is recommended that these controls are checked for functionality, and that server-side controls are put in place to prevent the accessing of privileges and information outside of allocated and intentional user permissions.

Affected System(s):

acesectest.adeptia.com/*
acesectest.adeptia.com/rest/users/[UUID-endpoint]?profileUpdate=true

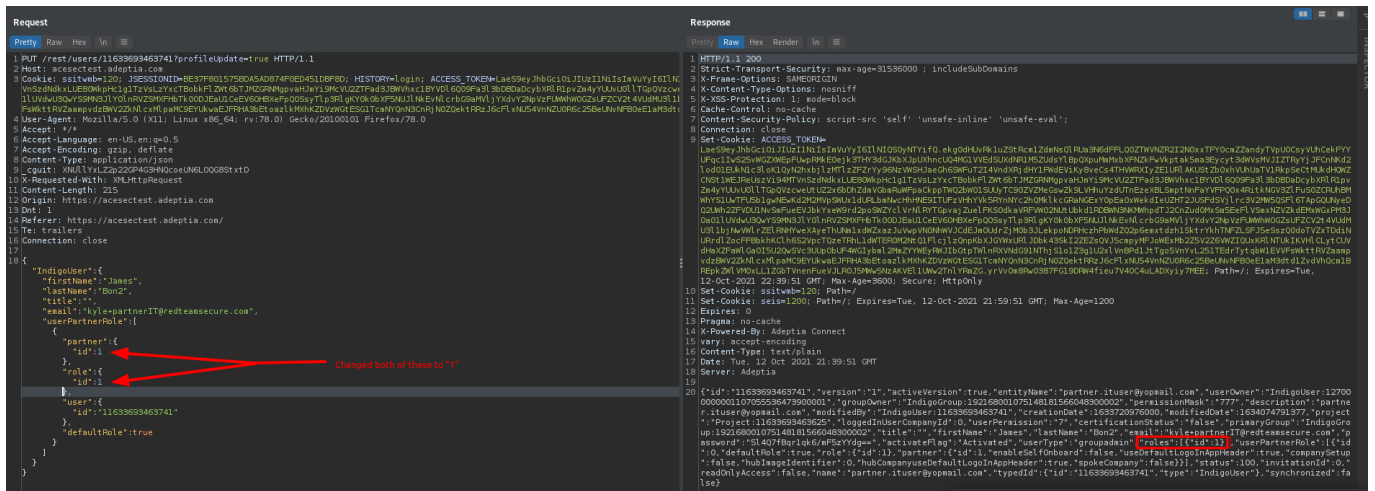
Instance(s):

11

Status:

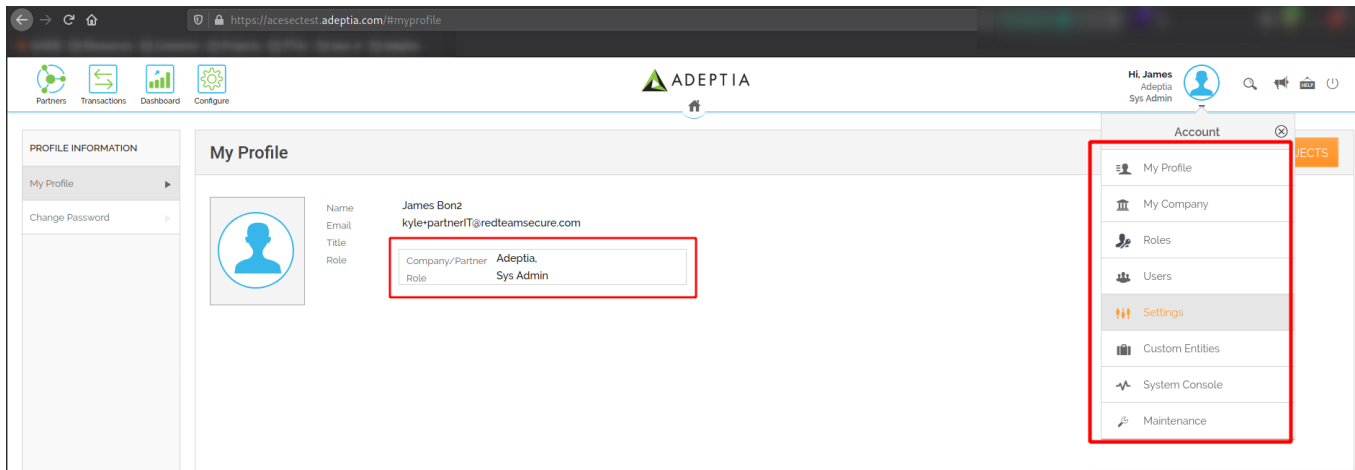
Remediated

Evidence:



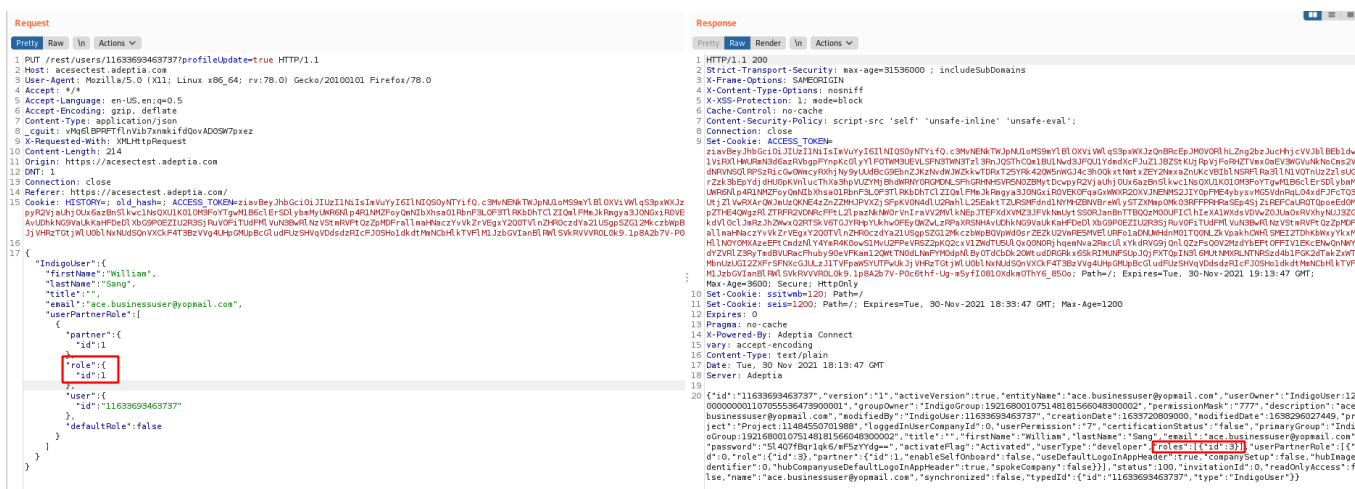
Evidence notes:

The profileUpdate request is intercepted and the partner-id and role-id are tampered before forwarding to the server.



Evidence notes:

The changes are no longer reflected in the UI after refreshing the session and are now reflected in the UI.



Evidence notes:

Update: Nov. 30, 2021

PUT request, which no longer changes the role id.

Severity Calculation:

The process for calculating the finding's severity is derived by assigning a numeric value between 0 and 9 to four (4) criteria separated into Likelihood and Impact. The formula is best represented here: **Likelihood**(Threat Agents + Vulnerability Factors) /2 + **Impact**(Technical Impact + Business Impact) /2 = **Risk Rating**(Likelihood + Impact) /2

High (7) = (Likelihood (7 + 7) /2 = **High (7)** + Impact (7 + 7) /2 = **High (7)**) /2

Reference(s):

- <https://cwe.mitre.org/data/definitions/285.html>
- <https://www.securitymetrics.com/blog/attackers-known-unknown-authorization-bypass>
- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/05-Authorization_Testing/02-Testing_for_Bypassing_Authorization_Schema

CVSS:

(AV:N/AC:M/Au:S/C:I/C:A/N)

[\[Back to Top\]](#)

[THIS FINDING HAS BEEN REMEDIATED]

Finding(s)

2. Unverified Password Change | High (6.25)

[THIS FINDING HAS BEEN REMEDIATED]

Description:

When setting a new password for a user, the application does not require knowledge of the original password or use another form of identity verification outside of the presence of session tokens.

Impact:

If an attacker is able to take control of a valid session, they could easily change the victim's password.

Test(s) Conducted:

Beginning as a logged in user, change that user's password, observing whether the old password is required. The password change process should then be examined for vulnerability to Cross Site Request Forgery. Further testing should attempt to change a different user's password. Successive failures of these tests indicate increasing severity of this finding.

Finding Comments:

The password change functionality can be abused in order to allow a logged-in user to change the password without first entering the existing password. This is due to the following circumstances:

The client-side application initially requests the user to enter the existing password in order for it to be changed. When this request is sent, the server checks the validity of the password and responds with a statement "passwordmatch:true" or "passwordmatch:false", depending on whether the correct password was entered.

By intercepting the server's response and changing a "false" statement to "true", the client-side application, after receiving "true", proceeds to submit a subsequent request to set the new password.

A valid Access Token can also simply be used directly to submit this request, if the structure of the request is already known. This is true even if the Access Token is expired. The server unconditionally processes requests to this endpoint, if they are paired with a valid or expired Access Token.

The risk presented by this vulnerability is enhanced by a number of factors. First, because the Access Token provided does not appear to expire. Second, because an attacker can also manipulate the UUID of the request in order to successfully change another user's password without verification. This aspect of the vulnerability is covered further in the finding "Authorization Bypass." The UUID is difficult to guess for an attacker with a low level of existing access. However, they are not random and enumeration would likely be possible. An attacker who has the ability to sniff network traffic could also observe these numbers in HTTP requests. They could also take advantage of the existing Authorization-Bypass vulnerability to escalate their role to admin, enumerate endpoints for various users, and then revert their role in order to minimize visibility. They could then hold on to the endpoints and the expired access token until such a time as they decide to take over another account.

Because the UUIDs are incremented instead of random, it could also be possible for an attacker to create a DoS event by rapidly submitting password-change requests while targeting all user endpoints in a given range.

Update: Nov. 30, 2021

During retesting RedTeam found that submitting a password change request with an incorrect current password resulted in a 401 error. This prevented the unauthorized change of a password, thus remediating this finding.

Recommendations:

Require knowledge of the original password or use another form of identity verification in order to change a password.

Affected System(s):

acesectest.adeptia.com/#passwordchange

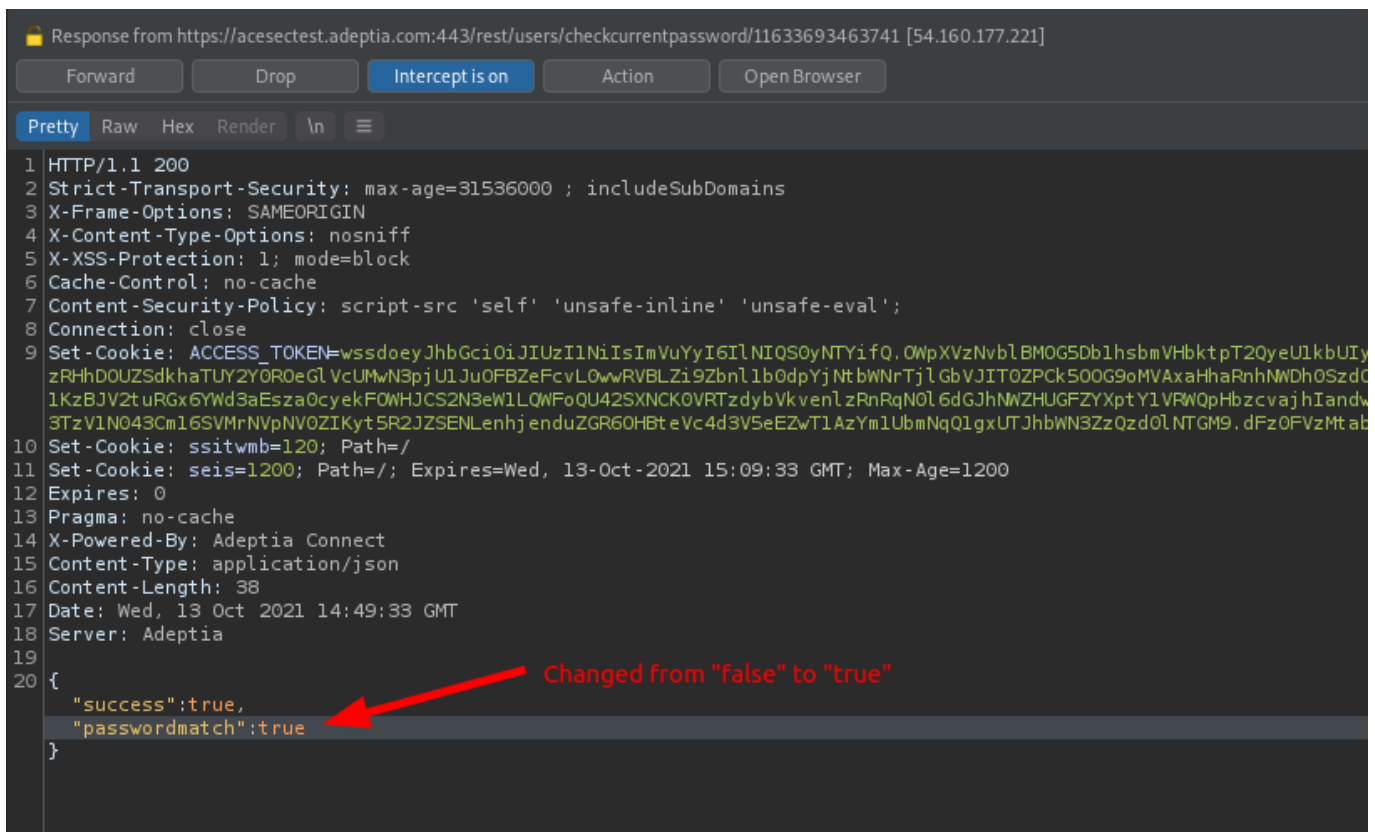
Instance(s):

11

Status:

Remediated

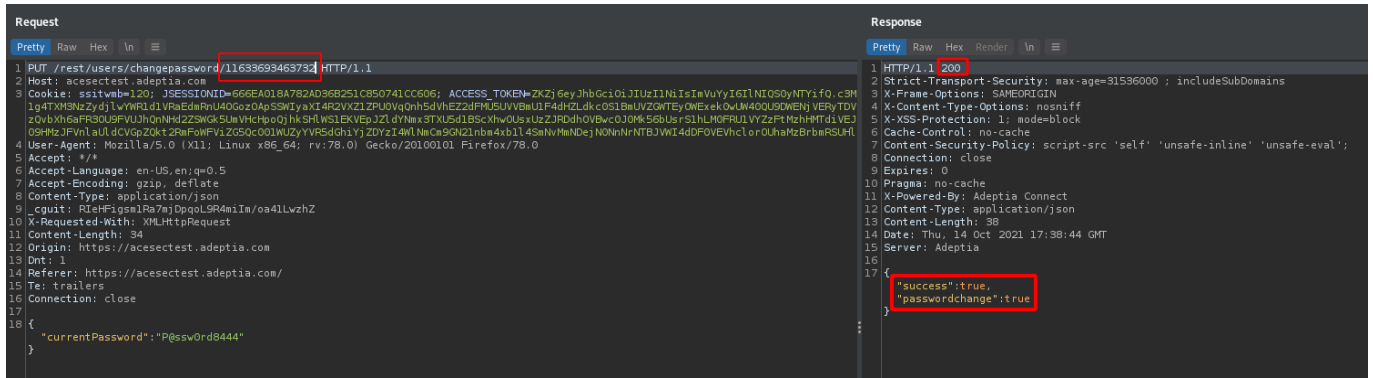
Evidence:



```
Response from https://acesectest.adeptia.com:443/rest/users/checkcurrentpassword/11633693463741 [54.160.177.221]
Forward Drop Intercept is on Action Open Browser
Pretty Raw Hex Render \n
1 HTTP/1.1 200
2 Strict-Transport-Security: max-age=31536000 ; includeSubDomains
3 X-Frame-Options: SAMEORIGIN
4 X-Content-Type-Options: nosniff
5 X-XSS-Protection: 1; mode=block
6 Cache-Control: no-cache
7 Content-Security-Policy: script-src 'self' 'unsafe-inline' 'unsafe-eval';
8 Connection: close
9 Set-Cookie: ACCESS_TOKEN=wssdoeyJhbGciOiJIUzI1NiIsImVuYyI6IlNIQS0yNTYifQ.OWpXVzNvb1BMOG5Db1hsbmVHbktpT2QyeU1kbUIy
zRHhDOUZSdkaTUyZ0R0eGlVcUMwN3pjU1Ju0FBZeFcvL0wRVBLZi9Zbnl1b0dpYjNtbWNRtj1GbVJIT0ZPck50OG9oMVAXaHhRnhNWDh0Szdc
1KzBJV2tuRGx6YWd3aEsza0cyekF0WHJCS2N3eW1LQWFOQU42SxNCK0VRTzdYbVkenl3RnRqN0l6dGJhNWZHUGFZYXptY1VFRW0pHbzcvaWJhIandw
3TzV1N043Cm16SVMrNVpNV0ZIKyt5R2JJZENLenhjemuZGR60HBteVc4d3V5eEZwT1AzYm1UbmNqQ1gxUTJhbWNS3ZzQzd0LNTGM9.dFz0FVzMt ab
10 Set-Cookie: ssitwmb=120; Path=/
11 Set-Cookie: seis=1200; Path=/; Expires=Wed, 13-Oct-2021 15:09:33 GMT; Max-Age=1200
12 Expires: 0
13 Pragma: no-cache
14 X-Powered-By: Adeptia Connect
15 Content-Type: application/json
16 Content-Length: 38
17 Date: Wed, 13 Oct 2021 14:49:33 GMT
18 Server: Adeptia
19
20 {
  "success":true,
  "passwordmatch":true
}
```

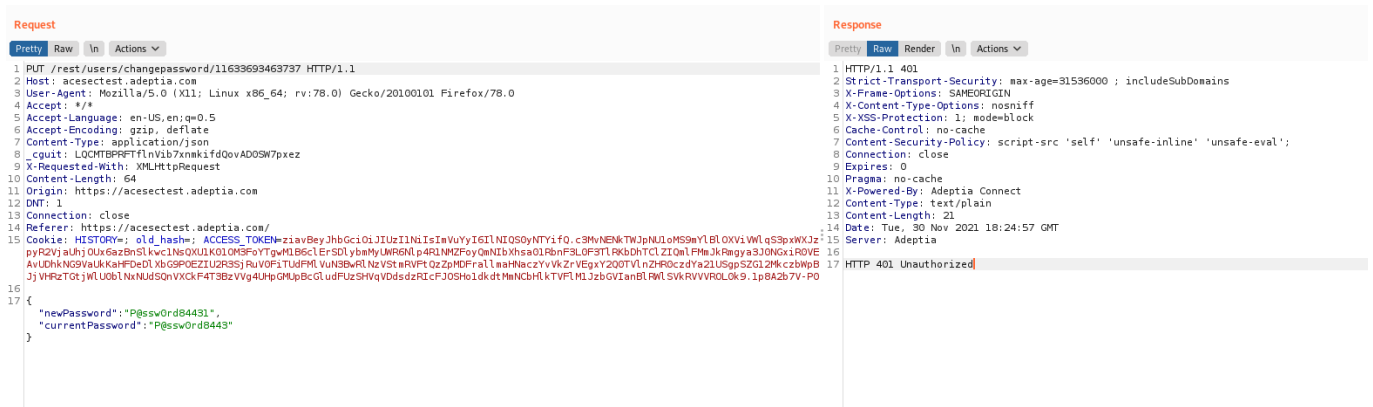
Evidence notes:

The server response is tampered with in order to initiate the subsequent password-change request.



Evidence notes:

The UUID is changed from that of the view-only of the sysadmin user. This results in a successful password change for that sysadmin user.



Evidence notes:

Update: Nov. 30, 2021

Unsuccessful password change, using incorrect current password.

Severity Calculation:

The process for calculating the finding's severity is derived by assigning a numeric value between 0 and 9 to four (4) criteria separated into Likelihood and Impact. The formula is best represented here: **Likelihood**(Threat Agents + Vulnerability Factors) /2 + **Impact**(Technical Impact + Business Impact) /2 = **Risk Rating**(Likelihood + Impact) /2

High (6.25) = (Likelihood (5 + 8) /2 = **High (6.5)** + Impact (7 + 5) /2 = **High (6)**) /2

Reference(s):

- <https://cwe.mitre.org/data/definitions/620.html>
- https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/09-Testing_for_Weak_Password_Change_or_Reset_Functionalities

CVSS:

(AV:N/AC:H/Au:N/C:N/I:P/A:N)

[\[Back to Top\]](#)

[THIS FINDING HAS BEEN REMEDIATED]

Finding(s)

3. Insufficient Session Expiration | Medium (5.5)

[THIS FINDING HAS BEEN REMEDIATED]

Description:

Insufficient Session Expiration occurs in scenarios in which a user session is terminated on the client side, but is still active on the server-side. As a result of the session remaining active it is possible for an attacker, who has compromised session tokens, to continue to take unauthorized actions after it appears to the user, on the client-side, that the session has been terminated.

Impact:

By not immediately terminating a user's session server-side, it is possible for an attacker to perform actions as that user in the window of time where the server has not ended the user's session. The risk of this vulnerability is magnified the longer the timeout window is, as a shorter timeout window reduces the amount of time an attacker has to act. In that window of time, the exposure to other session-based attacks is increased, as reuse of a valid session id to hijack a session requires that the session still be active. The impact of this vulnerability is dependent on how the back-end systems respond to the attacker's traffic, and could range from information disclosure to account takeover.

Test(s) Conducted:

Authenticated traffic is sent within a session, the user logs out, and older traffic is resent in that original session. If the server accepts and responds to the traffic, the session has not been properly terminated on the back-end server.

Finding Comments:

It is possible to exploit a valid, but expired Access Token to successfully submit a password-change request for any user. This is connected to the existing Authorization Bypass vulnerability detailed in a separate finding. In this case, the server is similarly failing to check to see if the token is expired. This could allow, for example, a laid-off employee or anyone else with even brief access to an authenticated session, to potentially take over other users' accounts at an undetermined later time.

During the time frame of the test, RedTeam was unable to identify other API calls that could be successfully made using an expired token. However given the breadth of functionality affected by the lack of server-side authorization checks, it should be considered very likely that there are more instances. At the present time, the number of instances is set to 11 to reflect the existing number of users that have access to a token.

It should be noted that there appear to be more ways to exploit lack of server-side authorization checks in order to extend sessions or bypass session expiration. RedTeam did not have time to conclusively identify more instances during this short engagement.

Update: Nov. 30, 2021

RedTeam retested this finding and found it to be remediated. Once a user has logged out, the access token can

no longer be used to modify the users profile.

Recommendations:

Ensure that the user's session is terminated immediately on both the client and the server upon the user initiating the logout process.

Affected System(s):

acesectest.adeptia.com

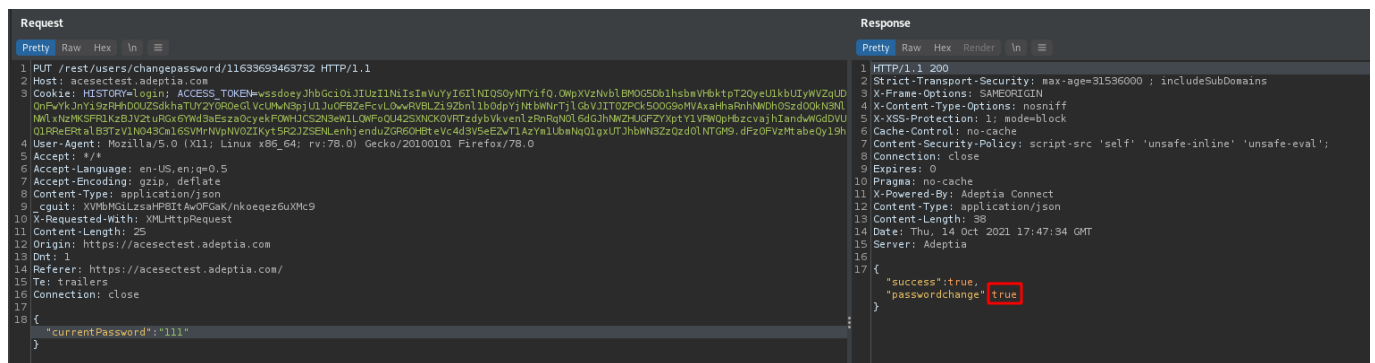
Instance(s):

11

Status:

Remediated

Evidence:



Evidence notes:

This password change request was successfully submitted in order to change the password of a separate, sysadmin user. This was successful even though the Access Token used was multiple days old and the original session had been logged out and a new session had been generated for the user several times in the intervening days.

Request

```
1 PUT /rest/users/11633693463735?profileUpdate=true HTTP/1.1
2 Host: acesectest.adeptia.com
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/json
8 _cguit: BTMBhBDoQ15yUpZ3QJ+CJ09/vo93oYkaKKoAj
9 X-Requested-With: XMLHttpRequest
10 Content-Length: 217
11 Origin: https://acesectest.adeptia.com
12 Connection: close
13 Referer: https://acesectest.adeptia.com/
14 Cookie: HISTORY=login; ssitwmb=120; ACCESS_TOKEN=so04geyJhbGc:
HY2klawoxcmYzWlJqdnJCYUNrRHhwWGLSdHRrKONMMVFnc0JiVET6alpVUWNlF
OaVZOR1FZYVF1UGhmTTYKeS0wb1lGZ3pLRHo4ejJVSnpREZMVG40S0dCUVpz
CbDjtRdc5U3RkQVRtCwC2dnFMRE0yTzdPCngldTUzQUc1RHF00UNHTXZuZUFx
15
16 {
  "IndigoUser":{
    "firstName":"Oliver",
    "lastName":"Liam",
    "title":"Big Boss",
    "email":"ace.sysadmin1@yopmail.com",
    "userPartnerRole":[
      {
        "partner":{
          "id":1
        },
        "role":{
          "id":1
        },
        "user":{
          "id":"11633693463735"
        },
        "defaultRole":true
      }
    ]
  }
}
```

Response

```
1 HTTP 401 Unauthorized
```

Evidence notes:

Update: 30 Nov. 2021

Testing if an access token from a session that had been logged out, can still change attributes of the account.

Request

Pretty
Raw
ln
Actions ▾

```

1 PUT /rest/users/11633693463735?profileUpdate=true HTTP/1.1
2 Host: acesectest.adeptia.com
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/json
8 _cguit: BTMBhBDoQ15yUpZ3QJ+CJ09/vo93oYkaKKoAj
9 X-Requested-With: XMLHttpRequest
10 Content-Length: 65
11 Origin: https://acesectest.adeptia.com
12 Connection: close
13 Referer: https://acesectest.adeptia.com/
14 Cookie: HISTORY=login; ssitwmb=120; ACCESS_TOKEN=so04geyJhbGc:
HY2klawoxcmYzWl JqdnJCYUNrRHhwWGl SdHRrKONMMVFnc0JiVet 6al pVUWNlI
OaVZORlFZYVF1UGhmTTYKeS8wb1l GZ3pLRHo4ej JVSnpPREZMVG40S0dCUVpzl
CbDJtRdc5U3RkQVRt cWc2dnFMRE0yTzdPCngldTUzQUc1RHF00UNHTXZuZUFxc
15
16
17 {
  "newPassword": "P@ssw0rd8443",
  "currentPassword": "P@ssw0rd8443"
}

```

Response

Pretty
Raw
Render
ln
Actions ▾

```

1 HTTP 401 Unauthorized

```

Evidence notes:

Update: Nov. 30, 2021

Testing if the access token from a session that had been logged out, can still change a users password.

Severity Calculation:

The process for calculating the finding's severity is derived by assigning a numeric value between 0 and 9 to four (4) criteria separated into Likelihood and Impact. The formula is best represented here: **Likelihood**(Threat Agents + Vulnerability Factors) /2 + **Impact**(Technical Impact + Business Impact) /2 = **Risk Rating**(Likelihood + Impact) /2

$$\text{Medium (5.5)} = (\text{Likelihood (5 + 5)} / 2) + \text{Impact (6 + 6)} / 2 = \text{High (6)} / 2$$

Reference(s):

<https://cwe.mitre.org/data/definitions/613.html>

https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#session-expiration

CVSS:

(AV:N/AC:H/Au:N/C:N/I:P/A:N)

[\[Back to Top\]](#)

[THIS FINDING HAS BEEN REMEDIATED]

Finding(s)

4. Unverified Password Change | Medium (4.25)

[THIS FINDING HAS BEEN REMEDIATED]

Description:

When setting a new password for a user, the application does not require knowledge of the original password or use another form of identity verification outside of the presence of session tokens.

Impact:

If an attacker is able to take control of a valid session, they could easily change the victim's password.

Test(s) Conducted:

Beginning as a logged in user, change that user's password, observing whether the old password is required. The password change process should then be examined for vulnerability to Cross Site Request Forgery. Further testing should attempt to change a different user's password. Successive failures of these tests indicate increasing severity of this finding.

Finding Comments:

The email change process does not require a user to provide the existing password in order to verify the change. This means that if a malicious actor is able to gain access to an active session (i.e., in-person, physical access to a computer), then they can easily change the registered email and send themselves a password-reset token, completing the account takeover.

Furthermore, it is possible for users with sufficient privileges to likewise change the emails and therefore passwords of almost any user without first proving knowledge of the original password, or proving access to the original email account.

Update: Nov. 30, 2021

During retesting RedTeam was unable to change the email address associated with any of the testing accounts. As such, it was not possible to evaluate the 'forgot password' email change options.

Update: Nov. 30, 2021

The previous update was based on a misunderstanding of the finding. Making the email field read-only remediated this finding.

Recommendations:

Require knowledge of the original password or use another form of identity verification in order to change a password.

Affected System(s):

acesectest.adeptia.com

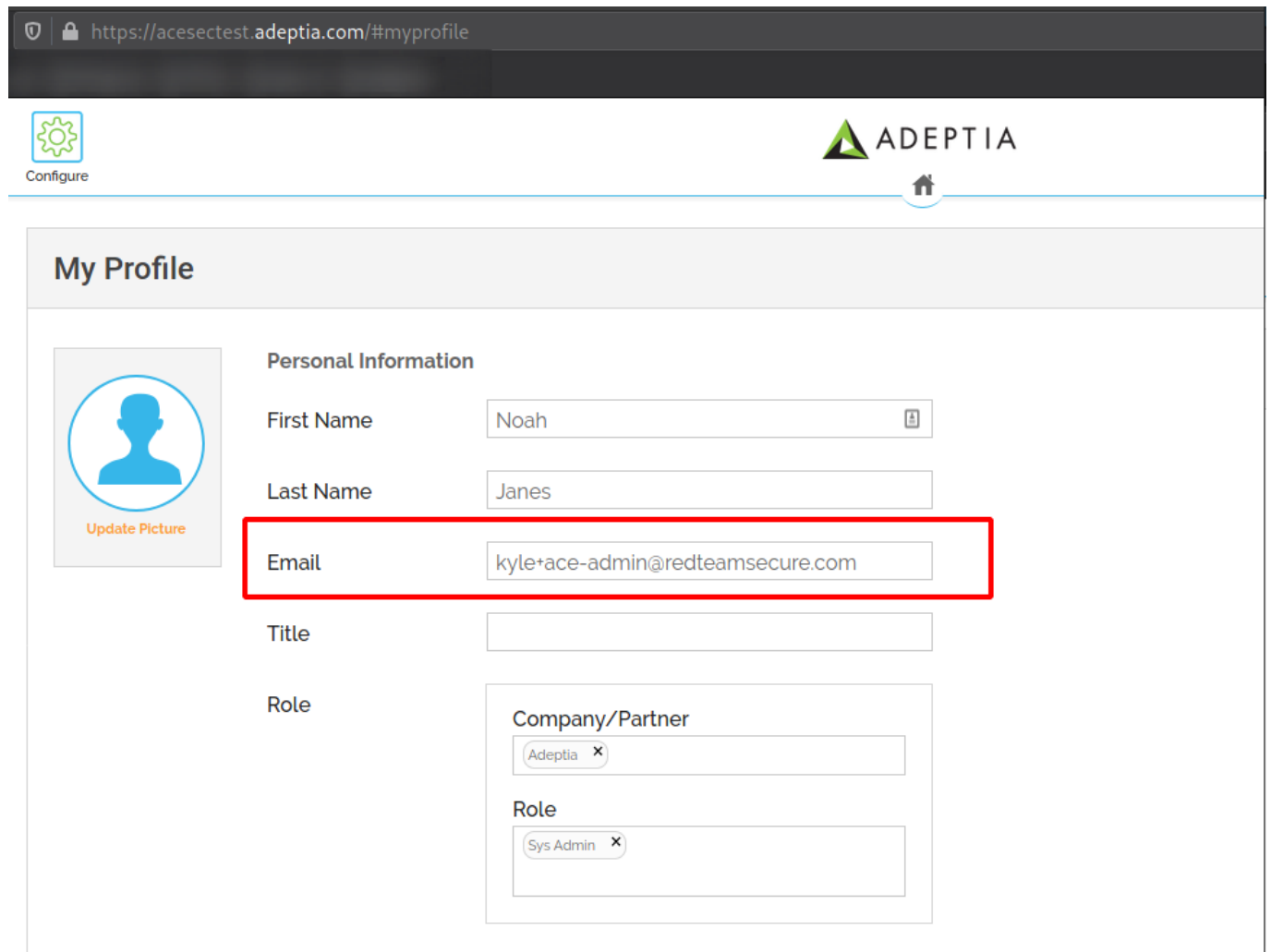
Instance(s):

11

Status:

Remediated

Evidence:



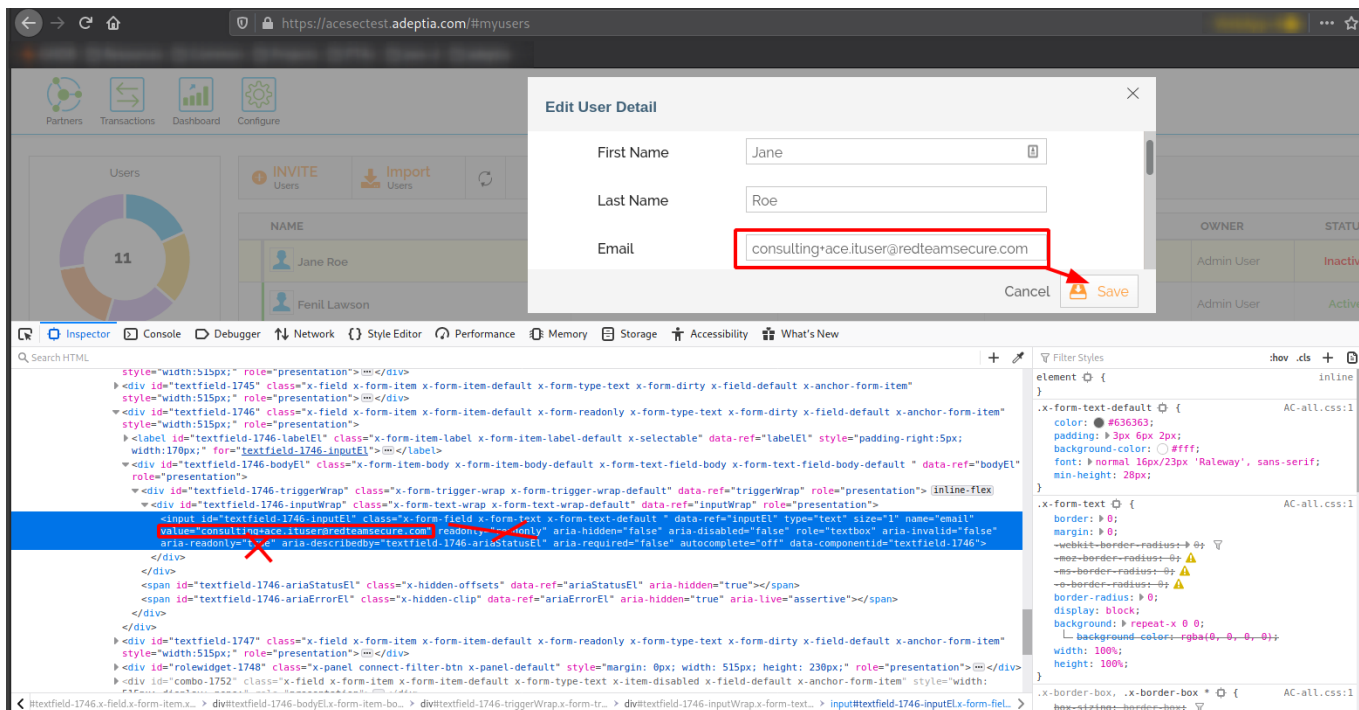
The screenshot shows a web browser window with the URL `https://acesectest.adeptia.com/#myprofile`. The page header includes a 'Configure' button (gear icon) and the ADEPTIA logo with a home icon. The main content area is titled 'My Profile' and contains a 'Personal Information' section. On the left is a profile picture placeholder with an 'Update Picture' link. The form fields are as follows:

Field	Value
First Name	Noah
Last Name	Janes
Email	kyle+ace-admin@redteamsecure.com
Title	
Role	Company/Partner: Adeptia, Role: Sys Admin

The 'Email' field is highlighted with a red rectangular border.

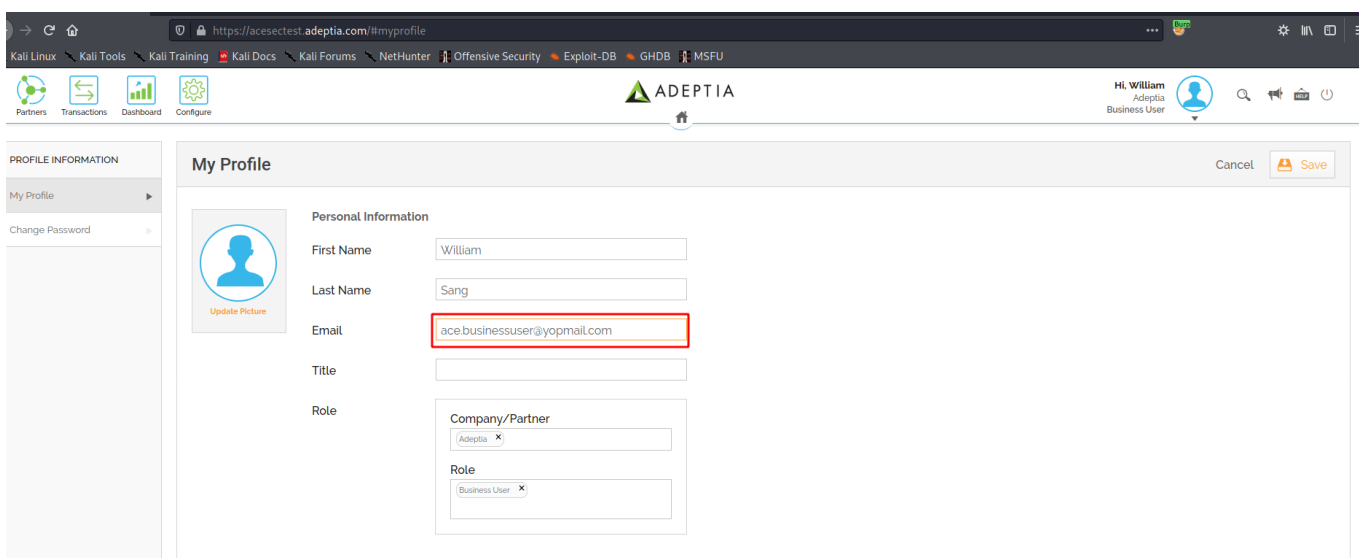
Evidence notes:

An administrative user's email can be changed without first proving ownership of the original email account.



Evidence notes:

The user can also change the emails, and therefore passwords, of other users.



Evidence notes:

Update: Nov. 30, 2021

The field that is no longer changeable, preventing the retesting of the 'forgot password' feature.

Severity Calculation:

The process for calculating the finding's severity is derived by assigning a numeric value between 0 and 9 to four (4) criteria separated into Likelihood and Impact. The formula is best represented here: **Likelihood**(Threat Agents + Vulnerability Factors) / 2 + **Impact**(Technical Impact + Business Impact) / 2 = **Risk Rating**(Likelihood + Impact) / 2

$$\text{Medium (4.25)} = (\text{Likelihood (3 + 4) / 2} = \text{Medium (3.5)} + \text{Impact (6 + 4) / 2} = \text{Medium (5)}) / 2$$

Reference(s):

<https://cwe.mitre.org/data/definitions/620.html>

https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authen

entication_Testing/09-Testing_for_Weak_Password_Change_or_Reset_Functionalities

CVSS:

(AV:N/AC:H/Au:N/C:N/I:P/A:N)

[\[Back to Top\]](#)

[THIS FINDING HAS BEEN REMEDIATED]

Finding(s)

5. Weak Password | Medium (4)

Description:

A password is considered to be weak if it is easily guessable, publicly available (ie: default credentials), or if the complexity is low enough that the password can be revealed by trying all combinations of allowed characters in a short amount of time. Strong passwords should be 15 characters or more, and include 4 character types: upper and lower case alphabetical characters, numbers, and special characters. Additionally, strong passwords should not consist of easily guessable words and phrases.

Impact:

An attacker that can successfully guess a weak password will gain access to systems and services that may contain sensitive information. This can also allow an attacker to further compromise a system by escalating their privileges beyond those of the compromised account.

Test(s) Conducted:

While performing a penetration test, authentication attempts are made in order to determine if default credentials are accepted, or if easily guessable passwords are in use. Additionally, if a password hash is obtained it will be run against a list of common passwords. If a password is successfully guessed or if default credentials are used, the password is considered weak. Password reset functionality is also probed to identify if periodic password changes are required, and if one can reuse the same value for successive password changes.

Finding Comments:

The password policy for users requires only a minimum of eight (8) characters. RedTeam recommends a minimum of 15 characters with enforced complexity in order to minimize the risk that passwords can be guessed. The severity of this finding has been elevated slightly because under certain conditions, it is possible to rapidly submit large volumes of brute-force password guessing attempts to the application.

Update: Nov. 30, 2021

RedTeam was informed by Adeptia that this was an application requirement. The brute-force vulnerability was remediated, which reduced the Vulnerability and Probability factors and the severity of this finding by two points each. This reduced the overall severity one point, to 4 out of 10, which is still Medium severity.

Recommendations:

Enforce a password policy that requires users to choose passwords that cannot be easily guessed. A typical pattern for this would be to require:

- fifteen (15) characters minimum
- at least one (1) uppercase letter
- at least one (1) lowercase letter
- at least one (1) number
- at least one (1) special character

When possible, ensure that passwords do not use words from the dictionary. Password values should be _____

reused no more often than once every ten password changes.

Affected System(s):

acesectest.adeptia.com

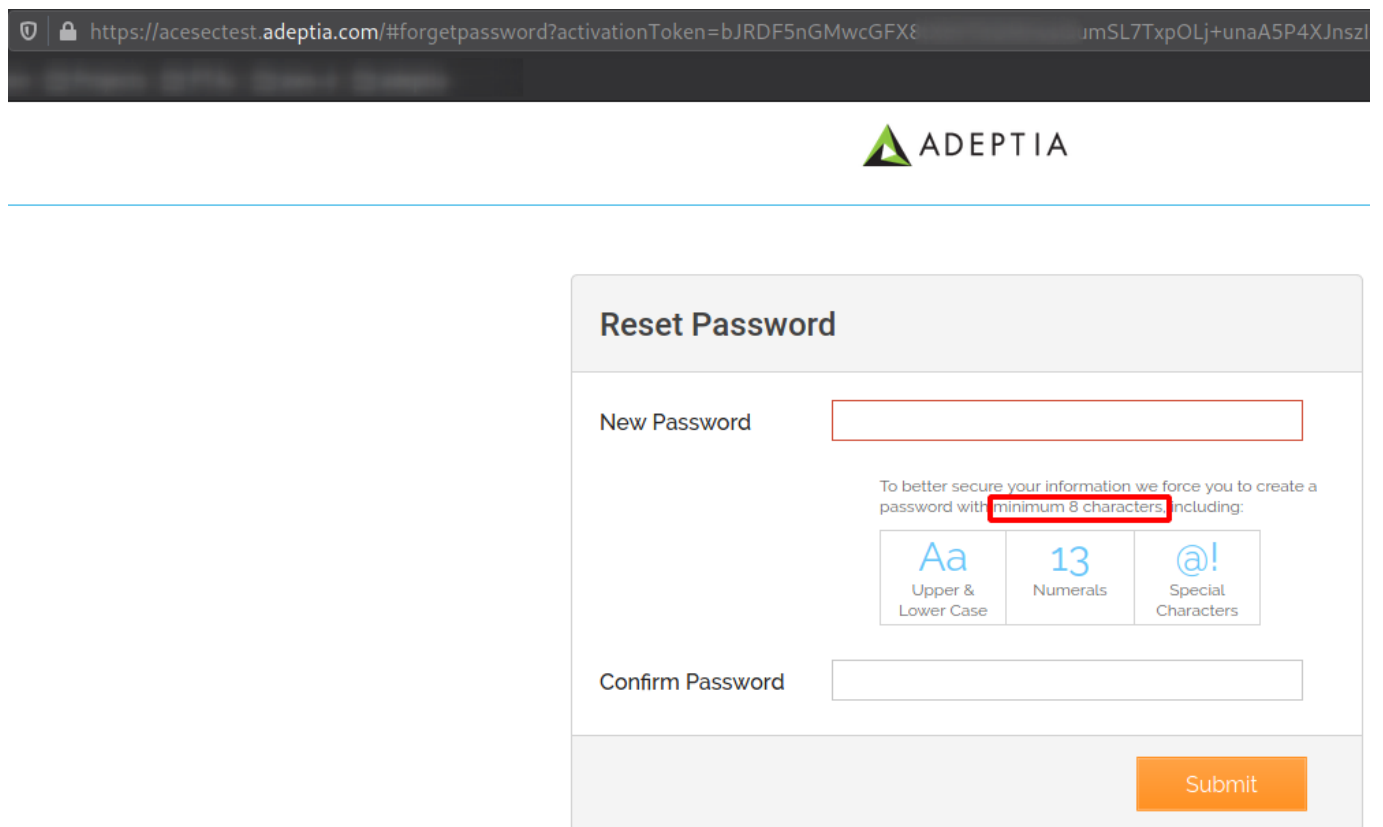
Instance(s):

11

Status:

Not Remediated

Evidence:



Evidence notes:

The password policy is shown.

Severity Calculation:

The process for calculating the finding's severity is derived by assigning a numeric value between 0 and 9 to four (4) criteria separated into Likelihood and Impact. The formula is best represented here: **Likelihood**(Threat Agents + Vulnerability Factors) /2 + **Impact**(Technical Impact + Business Impact) /2 = **Risk Rating**(Likelihood + Impact) /2

Medium (4) = (Likelihood (3 + 3) /2 = **Low (3)** + Impact (5 + 5) /2 = **Medium (5)**) /2

Reference(s):

- <http://www.welivesecurity.com/2016/01/20/weak-passwords-continue-pose-huge-security-threat/>
- http://www.theregister.co.uk/2016/03/02/password_scanning_honeypot_research/
- <https://cwe.mitre.org/data/definitions/521.html>

<https://www.acunetix.com/blog/web-security-zone/common-password-vulnerabilities/>

CVSS:

(AV:A/AC:L/Au:S/C:C/I:C/A:C)

[\[Back to Top\]](#)

Finding(s)

6. Brute Force Requests Allowed | Medium (3.5)

[THIS FINDING HAS BEEN REMEDIATED]

Description:

The application does not throttle requests to critical functionality, either in terms of time between requests or in terms of total failed requests. This means an attacker can make a large number of requests (login attempts, database search, etc.) without limitation. This may allow for password guessing, dumping/scraping of data via repeated searches, or potentially a denial of service if resource-intensive operations are requested.

Impact:

A brute force or dictionary attack against a login system could allow an attacker to compromise accounts and perform actions on behalf of that user. If this user has sufficient permissions (such as an administrator) this could allow for further exploitation of the target system, including but not limited to: data exfiltration, tampering, and deletion, the discovery of sensitive information, placement of software backdoors and/or exploitation of the host operating system. Rapidly-repeated searches against a database can be performed to make an offline copy of data for further analysis or competitive advantage. Given enough time, a full copy of the database could be made. A denial of service attack could be feasible if a resource-intensive task is requested repeatedly, such as generating a large report, scheduling a backup of a database, or requesting pages with logic errors leading to timeouts.

Test(s) Conducted:

Several requests are made to the affected service (login, search, etc.) and the responses are analyzed for signs of a temporary lockout or other such throttling. If no such limit is reached within a reasonable time, then any amount of attempts can be made as quickly as the application is able to respond, allowing for a brute-force approach to be taken.

Finding Comments:

The change-password functionality can be abused by an authenticated user to engage in brute-force password guessing attacks against other users. As the authenticated user types the existing password for verification purposes, the application sends requests to the server to actively check the validity of the password. By changing the User ID in the URI of the request and then submitting a large volume of identical requests containing different passwords, a malicious actor can use the server responses to learn whether a password is valid or not.

The severity of this finding is mitigated because the user must already be authenticated, and because a lower-privileged user will have some difficulty guessing the correct endpoint for a given user. However, these UUIDs are not random and can likely be effectively enumerated by a well-resourced or simply a patient attacker.

Update: Nov. 30, 2021

During retesting RedTeam found that the password check endpoint could no longer be tested for other UUIDs.

Recommendations:

Enable a request limit on a per-service basis and enforce a temporary lockout period for exceeding the limit. Exponential lockout time increases can help make attackers' attempts more costly in terms of time spent waiting for lockouts to expire.

Affected System(s):

acesectest.adeptia.com/rest/users/checkcurrentpassword/[UUID-endpoint]

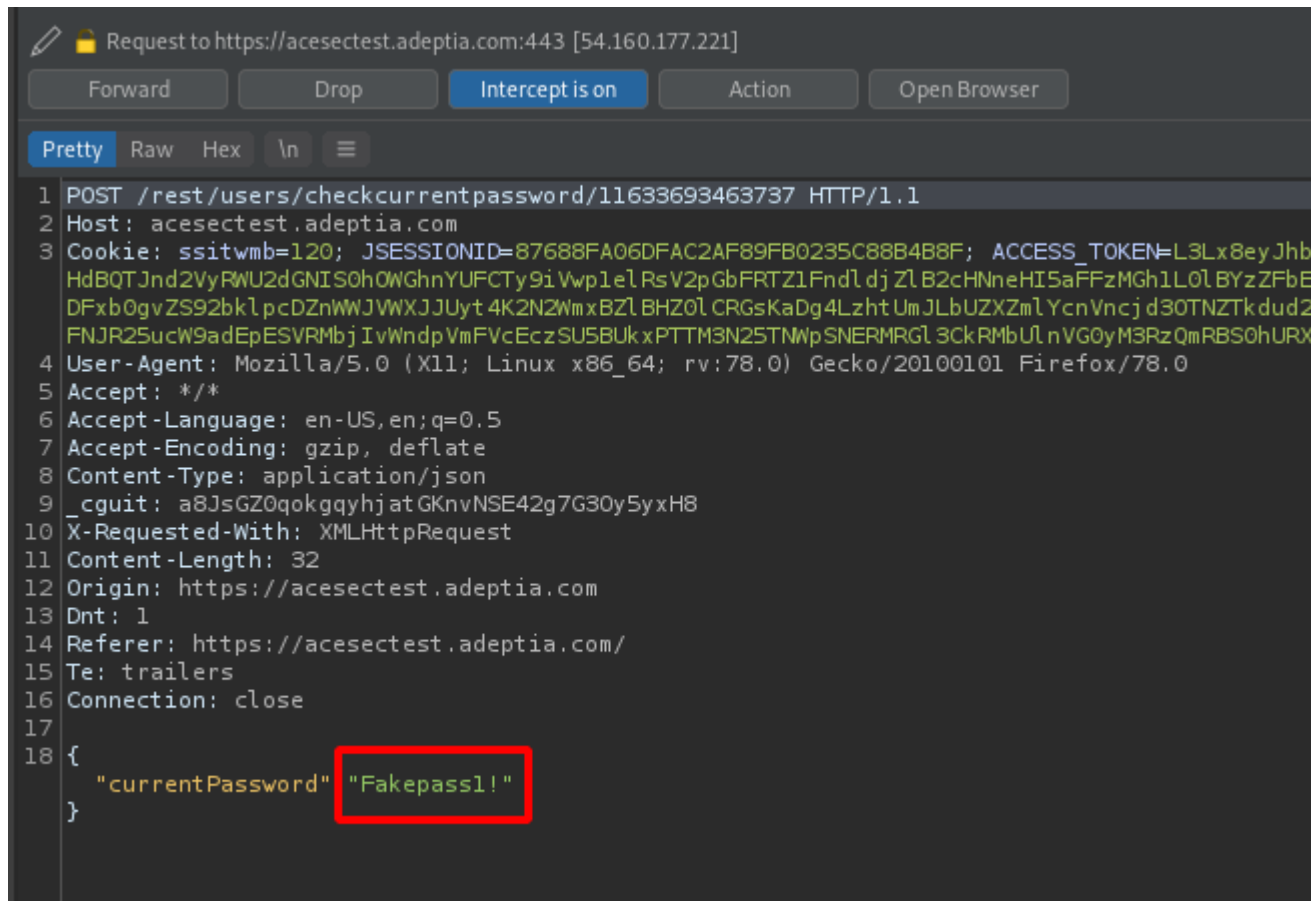
Instance(s):

11

Status:

Remediated

Evidence:



```
Request to https://acesectest.adeptia.com:443 [54.160.177.221]
Forward Drop Intercept is on Action Open Browser
Pretty Raw Hex \n
1 POST /rest/users/checkcurrentpassword/11633693463737 HTTP/1.1
2 Host: acesectest.adeptia.com
3 Cookie: ssitwmb=120; JSESSIONID=87688FA06DFAC2AF89FB0235C88B4B8F; ACCESS_TOKEN=L3Lx8eyJhb
  HdBQTJnd2VyRWU2dGNIS0h0WghnYUfCTy9iVwp1eLrsV2pGbFRTZ1FndldjZlB2cHNneHI5aFFzMGh1L0lBYzZFB
  DFxb0gvZS92bklpcDZnWWJWVXJJUyt4K2N2WmxBZlBHZ0lCRGsKaDg4LzhtUmJLbUZlZm1YcnVncjd30TNZTk
  dud2FNJR25ucW9adEpESVRMbjIvWndpVmFVcEczSU5BUkxPTTM3N25TNWpSNERMRL3CkRMbUlNlVG0yM3RzQmRBS0hURX
4 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
5 Accept: */*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: application/json
9 _cguit: a8JsGZ0qokgqyhjatGKnvNSE42g7G30y5yxH8
10 X-Requested-With: XMLHttpRequest
11 Content-Length: 32
12 Origin: https://acesectest.adeptia.com
13 Dnt: 1
14 Referer: https://acesectest.adeptia.com/
15 Te: trailers
16 Connection: close
17
18 {
  "currentPassword": "Fakepass1!"
}
```

Evidence notes:

This is the request that can be exploited. The UUID can be incremented to target various users, and various passwords can be submitted in a rapid series of requests.

Attack Save Columns

Results Target Positions Payloads Resource Pool Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	"passwordmatch":false	"passwordmatch":true
3412	zenith	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3413	zeosx	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3414	zephyr	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3415	zeppelin	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3416	zeus	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3418	ziggy	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3417	zhongguo	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3419	zimmerman	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3420	zjaaadc	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3421	zmodem	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3422	zombie	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3424	zxcvbnm	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3423	zorro	200	<input type="checkbox"/>	<input type="checkbox"/>	2344	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3425	P@ssw0rd8443	200	<input type="checkbox"/>	<input type="checkbox"/>	2343	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Request Response

Pretty Raw Hex Render \n ☰

```

14 owered-By: Adeptia Connect
15  tent-Type: application/json
16  tent-Length: 37
17  e: Wed, 13 Oct 2021 14:32:43 GMT
18  ver: Adeptia
19
20
  success":true,
  passwordmatch":true

```

Evidence notes:

3425 requests, each containing a different password, were submitted in 1-2 minutes. The final password is correct, and the server response reflects this.

Request

Pretty Raw ln Actions

```

1 PUT /rest/users/checkcurrentpassword/11633693463735 HTTP/1.1
2 Host: acesectest.adeptia.com
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/json
8 _cguit: CUqaBkiJl0xgvmdMYcecwm7fh792g6PTW7br+
9 X-Requested-With: XMLHttpRequest
10 Content-Length: 34
11 Origin: https://acesectest.adeptia.com
12 DNT: 1
13 Connection: close
14 Referer: https://acesectest.adeptia.com/
15 Cookie: ssitwmb=120; JSESSIONID=D0B7115D00C30D55002E81BE674D88
3hDOC9LdWl uUW5CM3BQ0HhVwnhLaUhfUOt aZgozRE1G0FNYMVhOenBycVRodOf
nLiaU1SznMvVFFKsjBJaVpKzJYVTFUZwDaS9iUG9mQUc3dHkKWW1tVdDYWXt
nArcWZidytra3crWTdqYmhhUk9LbnRybOpYQmRvb3JwVDBTswFHeDNmLzNEYk!
16 {
17   "currentPassword": "P@ssw0rd8443"
18 }

```

Response

Pretty Raw Render ln Actions

```

1 HTTP/1.1 404
2 Strict-Transport-Security: max-age=31536000 ;
includeSubDomains
3 X-Frame-Options: SAMEORIGIN
4 X-Content-Type-Options: nosniff
5 X-XSS-Protection: 1; mode=block
6 Cache-Control: no-cache
7 Content-Security-Policy: script-src 'self' 'unsafe-inline'
'unsafe-eval';
8 Connection: close
9 Set-Cookie: ACCESS_TOKEN=
27RF4eyJhbGciOiJIUzI1NiIsInR5cGU6IjE6IiwiaWF0Ij0iLmVudC50yNTYiFQ.c3MvNENkTWJ
pNU1oMS9mYlBlOxViVWlqS3pxWXJzQnBRcEpJM0VORl hLY2xFSVRSUST3a0
dUY0Q4ZnYrU0LZdk9IZUNoMudUa3FLZgpYdG55T3YyQjFwdl1BRVoyejR3U
lVjTGIyRk9XbE84MzL ySGpXSLFXcU9INudYcVo2SHlnbWFRUWfYVHpxd3NI
eVl r0U9VS05OZ2d2CmJ3dUUVvL1VfzFiaOpzR3NLWEH4QVl6cytTMWh60TI
STUxweHZXR2VDvkVkZFUxMG51L09kblpYNGxNczBCam1adXZMQVE5WgZyZ
QKcW1zNEp4RUZMQzBHM0yZTZpUXB4STdvnHPTZHNXdHR6ZTQ4T3NpaUVha
WZkZCt kZ3hDOC9LdWl uUW5CM3BQ0HhVwnhLaUhfUOt aZgozRE1G0FNYMVhO
enBycVRod0NHVU16VDV3V3ZkbUdTTnJEWkhGNm1TWHRlMTRrdLcxZG1kWmd
RczJnZDZvKzZ1TmhiMwVYnL vTzZjCit oUxcvN05wWw12WjVaTTfubERGJM
FhZWhXc3M30VJxTudhL3ZZbkxQNLVFenRscldwVjY0b2V6R05SY2RWOUDJZ
EZIYXZiL3NobmUKVTLtWnZDUFU0S2w2c1piQWNaSFV1Nkh0cFVHMUFzSnpX
cDB6bUZ0U1EzNkMSVHM1aW54U3Y3d24xQmtxWlUvUmNkMTdad3NzaGFvZwo
OajFRQXB3SxhkSEtIWDPTa3VudkYvOXV0ZWE3MwVhMET2S0pZREpINXRqaz
c1amtGUG1PbTljQ0L0Z1JQ03liWfJ6dUk3ZmNQSGRZCnNHRmZoQ2YvWl hvZ
G00QnhiaE01TUhd2YwcjLwUFRNMzRRdFRtdnLiaU1SznMvVFFKsjBJaVpV
KzJYVTFUZwDaS9iUG9mQUc3dHkKWW1tVdDYWXE0WHBLQj hLWWhNTZQeUM
zbl dMc2RvZHN0SEZsN3UvdmI4UmtvcdVjT2l pb1ZUQWhLbGpkaHJBtmZj cT
ZrWVESM1JTZQp0azBj aDE4TWdPTOhLZFJhcGvYlNnb3AwZwZdqVHvYlBFU
25j b0x1VmNuWnRSS3o1WnZQN0t sN1F0bGtZVFZxVGHtChoreStToGH4Ci85
Yk9o0Tl3TKROLzhPRjM4WksZVExvRkd6cUoreFVMSXPSdU05dTvNlZ6enp
Sc3pRWGNcclLSNESCbEcrQVFpUFVrdkZoRHNwME8KSLd5dVFj bklKakNQW
tLZW4xZCs5RGdmWp1SW82QUntc3l30EkvMjduVDBTQkdjaml6RXFVditSN
FBQRm1ZQXQwVUhpU2dSvnl0Aplam5WSEU1SnJ25i9yeXRTZFRVa3gzR09q
TjlxVnArcWZidytra3crWTdqYmhhUk9LbnRybOpYQmRvb3JwVDBTswFHeDN
mLzNEYkNHCjVaaOpTb2wvdfBndXVSUGUycWxld0dNSUStc1UxdW1NcXJqMm
1QcVU3UmnQWERl b1EzaE8xZkw3UEtoVFFwc2xjRkU9. d7jIu_hC2_ulOfEa
zdSTR1qdLhm0BU1ros7bzHVZ-Q; Path=/; Expires=Tue,

```

The password for this UUID is the one in the currentPassword field

but this token is for a different session

Evidence notes:

Update: Nov. 30, 2021 PUT request, showing that the checkcurrentpassword endpoint can no longer be used for other UUIDs

Severity Calculation:

The process for calculating the finding's severity is derived by assigning a numeric value between 0 and 9 to four (4) criteria separated into Likelihood and Impact. The formula is best represented here: **Likelihood**(Threat Agents + Vulnerability Factors) /2 + **Impact**(Technical Impact + Business Impact) /2 = **Risk Rating**(Likelihood + Impact) /2

Medium (3.5) = (Likelihood (4 + 3) /2 = **Medium (3.5)** + Impact (4 + 3) /2 = **Medium (3.5)**) /2

Reference(s):

- https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks
- <https://www.computerweekly.com/answer/Techniques-for-preventing-a-brute-force-login-attack>
- <https://phoenixnap.com/kb/prevent-brute-force-attacks>
- <https://www.imperva.com/learn/application-security/brute-force-attack/>

CVSS:

(AV:N/AC:M/Au:M/C:P/I:P/A:P)

[\[Back to Top\]](#)

[THIS FINDING HAS BEEN REMEDIATED]

Finding(s)

7. HTTP Strict Transport Security Not Enforced | **Low (2.75)**

[THIS FINDING HAS BEEN REMEDIATED]

Description:

Strict transport security is used to ensure that web traffic uses an encrypted channel (HTTPS) to view and access content. The identified web services do not enforce strict transport security, allowing for encrypted communication to be downgraded to unencrypted communication. This requires a user to mistype a URL that will result in an initial HTTP request being made in plain text. Additionally, utilizing the browser's autocomplete function can result in the same request being sent (i.e. typing "face" and the browser automatically finishes facebook.com). This attack requires that a malicious user already has a man-in-the-middle attack on the network to see the request being made, and also intercept traffic.

Impact:

Successful exploitation can allow an attacker to downgrade web server connections to plain text, and view sensitive information as it is being passed to the server.

Test(s) Conducted:

This finding is first identified if we are able to access site content using HTTP, and are not redirected to a secure (HTTPS) site. The second way of validating this finding is to identify if the HTTPS response header "Strict-Transport-Security" is being used.

Finding Comments:

While testing, RedTeam noted the HTTP Strict-Transport-Security header is not being used correctly. The max-age value is set to 0, which instructs the browser to remove the cached HSTS policy immediately. A max-age of two years is recommended. This vulnerability could allow for an attacker to downgrade the encrypted connection to a non-encrypted version (plain HTTP). In order for a malicious actor to exploit this vulnerability, they would need to have a man-in-the-middle position between the server and the client.

Reference:

<https://hstspreload.org/>

Update: Nov. 30, 2021

RedTeam found that HSTS was enforced, and included subdomains, thus remediating this finding.

Recommendations:

Configure the web server in such a way that requests for HTTP content are redirected to HTTPS. This can be done using the "Strict-Transport-Security" header in the response. Additionally, the "preload" option can be used to specify that a web browser should check the preload repository to ensure that traffic is supposed to be encrypted.

Affected System(s):

acesectest.adeptia.com

Instance(s):

1

Status:

Remediated

Evidence:

```
Start 2021-10-11 14:36:55 -->> 54.160.177.221:443 (acesectest.adeptia.com) <<--
rDNS (54.160.177.221): ec2-54-160-177-221.compute-1.amazonaws.com.
Service detected: HTTP

Testing HTTP header response @ "/"

HTTP Status Code      200
HTTP clock skew       0 sec from localtime
Strict Transport Security  HSTS max-age is set to 0. HSTS is disabled, just this domain
Public Key Pinning    --
Server banner         Adeptia
Application banner    --
```

Evidence notes:

The max-age is set to 0.

```
Testing HTTP header response @ "/" Current Password ●●●●●●●●●●●●●●●●
HTTP Status Code      200
HTTP clock skew       -1 sec from localtime
Strict Transport Security  365 days=31536000 s, includeSubDomains ●●●●●●●●●●
Public Key Pinning    --
Server banner         Adeptia
Application banner    --
Cookie(s)             (none issued at "/")
Security headers      X-Frame-Options: SAMEORIGIN
                     X-Content-Type-Options: nosniff
                     Content-Security-Policy: script-src 'self' 'unsafe-inline' 'unsafe-eval'
                     X-XSS-Protection: 1; mode=block
```

Evidence notes:

Update: Nov. 30, 2021

Scan results showing the HSTS settings.

Severity Calculation:

The process for calculating the finding's severity is derived by assigning a numeric value between 0 and 9 to four (4) criteria separated into Likelihood and Impact. The formula is best represented here: **Likelihood**(Threat Agents + Vulnerability Factors) /2 + **Impact**(Technical Impact + Business Impact) /2 = **Risk Rating**(Likelihood + Impact) /2

$$\text{Low (2.75)} = (\text{Likelihood (2 + 4)} / 2) + \text{Impact (2 + 3)} / 2 = \text{Low (3)} + \text{Impact (2 + 3)} / 2 = \text{Low (2.5)} / 2$$

Reference(s):

https://www.owasp.org/index.php/HTTP_Strict_Transport_Security

<http://www.troyhunt.com/2015/06/understanding-http-strict-transport.html>

CVSS:

(AV:A/AC:H/Au:S/C:C/I:C/A:N)

[\[Back to Top\]](#)

[THIS FINDING HAS BEEN REMEDIATED]

Finding(s)

8. 3DES Ciphers Supported | Low (2.25)

[THIS FINDING HAS BEEN REMEDIATED]

Description:

Transport Layer Security (TLS) is the protocol used to make an encrypted connection for client-server communication. This protocol is used in combination with ciphers to encrypt the data. In the case of the affected systems, a 3DES cipher is supported for this communication. These ciphers are limited to a 112-bit key length. It has been academically proven that a key length of less than 128-bits can be reversed by a malicious actor.

Impact:

If a malicious actor is able to gain suitable network position to capture network traffic, it may be possible to decrypt session details and other session information. In the original proof of concept, 785 GB of data was needed in order to exploit this vulnerability.

Test(s) Conducted:

A connection is made to the server using several DES-CBC3-SHA ciphers. If any of the connections using these ciphers are successful, the server is considered vulnerable.

Finding Comments:

The 3DES cipher suite has known weaknesses in its mathematical implementation which could enable an adversary in a privileged network position to decrypt intercepted network traffic and recover information sent to the affected host, including user credentials, session tokens, or potentially sensitive information.

Update: Nov. 30, 2021

RedTeam found that the 3DES cipher was no longer supported, thus remediating the finding.

Recommendations:

Disable the use of "DES-CBC3-SHA" ciphers within the server's configuration. If these ciphers cannot be disabled, the cipher order should be changed to ensure these ciphers are used only as a last resort.

Affected System(s):

acesectest.adeptia.com

Instance(s):

1

Status:

Remediated

Evidence:

```
Start 2021-10-11 14:37:08 -->> 54.160.177.221:443 (acesectest.adeptia.com) <<--
rDNS (54.160.177.221): ec2-54-160-177-221.compute-1.amazonaws.com.
Service detected: HTTP

Testing cipher categories
NULL ciphers (no encryption) not offered (OK)
Anonymous NULL Ciphers (no authentication) not offered (OK)
Export ciphers (w/o ADH+NULL) not offered (OK)
LOW: 64 Bit + DES, RC[2,4] (w/o export) not offered (OK)
Triple DES Ciphers / IDEA offered
```

Evidence notes:
3DES Ciphers are enabled.

```
Testing cipher categories
NULL ciphers (no encryption) not offered (OK)
Anonymous NULL Ciphers (no authentication) not offered (OK)
Export ciphers (w/o ADH+NULL) not offered (OK)
LOW: 64 Bit + DES, RC[2,4], MD5 (w/o export) not offered (OK)
Triple DES Ciphers / IDEA not offered
```

Evidence notes:
Update: Nov. 30, 2021
Scan results showing that the 3DES cipher is not offered.

Severity Calculation:

The process for calculating the finding's severity is derived by assigning a numeric value between 0 and 9 to four (4) criteria separated into Likelihood and Impact. The formula is best represented here: **Likelihood**(Threat Agents + Vulnerability Factors) /2 + **Impact**(Technical Impact + Business Impact) /2 = **Risk Rating**(Likelihood + Impact) /2

Low (2.25) = (Likelihood (3 + 2) /2 = **Low (2.5)** + Impact (2 + 2) /2 = **Low (2)**) /2

Reference(s):

- <https://www.rapid7.com/db/vulnerabilities/ssl-3des-ciphers>
- <https://medium.com/@cbirt/disabling-3des-and-changing-cipher-suites-order-22396cb05828>
- https://cheatsheetseries.owasp.org/cheatsheets/TLS_Cipher_String_Cheat_Sheet.html

CVSS:

(AV:N/AC:H/Au:N/C:P/I:P/A:N)

[\[Back to Top\]](#)

[THIS FINDING HAS BEEN REMEDIATED]

Finding(s)

9. Concurrent Login Sessions Allowed | **Low (1)**

Description:

Users could use one set of authentication credentials to have more than one unique logon session active at the same time. Even though this occurrence may not seem to be related to security it could reduce the granularity of session identifiers or may render logging less useful.

If multiple users can log in to the same account simultaneously, non-repudiation is not possible. This behavior could indicate that a user is attempting to perform malicious activity. Allowing multiple user sessions can allow for a bad actor to control a user's account at the same time as a legitimate user.

Impact:

This vulnerability could enable higher-risk attacks, such as cloning and hijacking sessions. Also, this can allow a bad actor to access the account at the same time as legitimate users with very little way of identifying the account has been compromised.

Test(s) Conducted:

RedTeam Security logs into the application using the same user account and different browsers. Additional checks are made for alerts or logging that may notify a user that their account is being used in another location or by another device.

Finding Comments:

It is possible to have concurrent sessions. This sets up some of the preconditions needed for user session hijack attacks and attacks leveraging stolen credentials if they can occur without detection. If allowing concurrent sessions is determined to have a legitimate business use case, presenting a notice of login activity with IP address to the user's registered email address and within the session would ensure the logged in user is aware of the multiple sessions.

Update: Nov. 30, 2021

RedTeam was informed by Adeptia that this was an application requirement.

Recommendations:

Consider limiting users to only one session per username. Despite many websites allowing concurrent sessions for the convenience of their users, it presents business risks that must be accepted or addressed. Preventing concurrent sessions can help prevent higher-risk exploits, such as session cloning and session hijacking.

If multiple user sessions are required, a message notifying the users that new session has been initiated should be implemented.

Affected System(s):

acesectest.adeptia.com

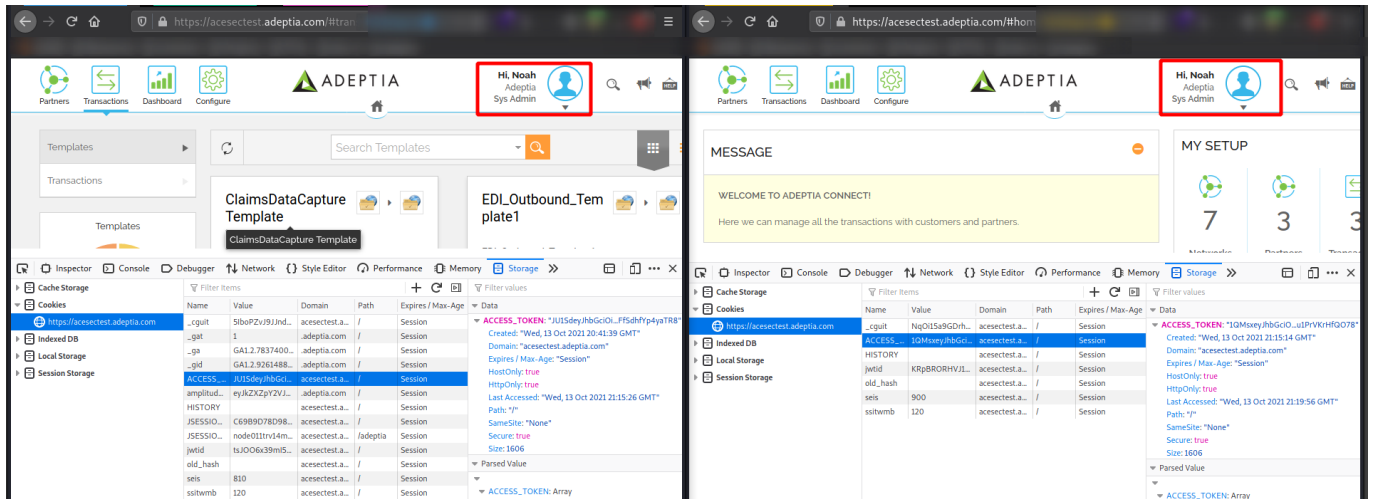
Instance(s):

1

Status:

Not Remediated

Evidence:



Evidence notes:

Two concurrent sessions are active.

Severity Calculation:

The process for calculating the finding's severity is derived by assigning a numeric value between 0 and 9 to four (4) criteria separated into Likelihood and Impact. The formula is best represented here: **Likelihood**(Threat Agents + Vulnerability Factors) / 2 + **Impact**(Technical Impact + Business Impact) / 2 = **Risk Rating**(Likelihood + Impact) / 2

$$\text{Low (1)} = (\text{Likelihood (1 + 1)} / 2) + \text{Impact (1 + 1)} / 2 = \text{Low (1)} / 2$$

Reference(s):

https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#simultaneous-session-logons

<https://www.networkworld.com/article/2227748/why-concurrent-logins-to-a-windows-network-are-a--very--bad-idea.html>

CVSS:

(AV:N/AC:H/Au:M/C:P/I:P/A:P)

[\[Back to Top\]](#)

Appendix A

Approach

RedTeam Security's application penetration test combines the results from industry-leading scanning tools with manual testing to enumerate and validate vulnerabilities, configuration errors, and business logic flaws. In-depth manual network testing enables us to find what scanners often miss.

Web applications are particularly vulnerable to external attack given that they are inherently designed to be accessible to the internet. While automated scanners check for known vulnerabilities, they are incapable of actually reporting on real business risk. Our Web application and API security testing helps you lower your risk of data breach, improve productivity, protect your brand, and maximize your ROI.

RedTeam Security's application penetration test service utilizes a risk-based approach to manually identify critical application-centric vulnerabilities that exist on all in-scope assets.

Using this approach, RedTeam's comprehensive Web Application Penetration Test covers the classes of vulnerabilities in the Open Web Application Security Project (OWASP) Top 10 2017 and beyond:

1. Injection
2. Broken Authentication
3. Sensitive Data Exposure
4. XML External Entities (XXE)
5. Broken Access Control
6. Security Misconfigurations
7. Cross-Site Scripting (XSS)
8. Insecure Deserialization
9. Using Components with known vulnerabilities
10. Insufficient Logging & Monitoring

Automated vs Manual Testing

RedTeam's approach consists of about 80% manual testing and about 20% automated testing - actual results may vary slightly. While automated testing enables efficiency, it is effective in providing efficiency only during the initial phases of a penetration test. At RedTeam Security, it is our belief that an effective and comprehensive test can only be realized through rigorous manual testing techniques.

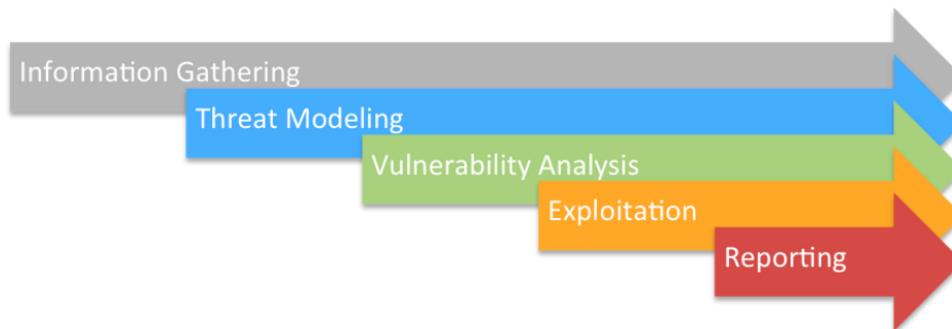
Tools

In order to perform a comprehensive real-world assessment, RedTeam Security utilizes commercial tools, internally developed tools and some of the same tools that hackers use on each and every assessment. Once again, our intent is to assess systems by simulating a real-world attack and we leverage the many tools at our disposal to effectively carry out that task. More information on some of the tools used during an engagement is included in Appendix C.

Appendix A

Methodology

Penetration Testing Methodology



Information Gathering

The information-gathering phase consists of Google search engine reconnaissance, server fingerprinting, application enumeration and more. Information gathering efforts results in a compiled list of metadata and raw output with the goal of obtaining as much information about the applications's makeup as possible. Reconnaissance includes initial application foot printing, metafile leakage review, service enumeration and operating system fingerprinting. The purpose of this step is to collectively map the in-scope application and prepare for threat identification.

During the Information Gathering phase, RedTeam Security will perform the following:

- Use discovery tools to passively uncover information about the application environment
- Identify entry points into the application, such as administration portals or backdoors
- Perform fingerprinting in order to identify the underlying development framework and individual components.
- Send fuzzing requests to be used in the analysis of error codes that may disclose valuable information that could be used to launch a more targeted attack
- Actively scan available services for vulnerabilities and develop a test plan for later phases in the assessment

Threat Modeling

With the information collected from the previous step, security testing transitions to identifying vulnerabilities within the in-scope environment. This typically begins with automated scans but quickly morphs into manual testing techniques using more pointed and direct tools. During the threat-modeling step, assets are identified and categorized into threat categories. These may involve: sensitive documents, trade secrets, financial information, etc.

During this phase, RedTeam Security will perform the following:

- Use open source, commercial and internally developed tools to identify well-known vulnerabilities
- Enumerate the in-scope applications to effectively build a map of each of the features, components, and areas of interest
- Build the applications's threat model using the information gathered in this and previous phases to be

used as a plan of attack for later pahases within the assessment

- Upload preliminary vulnerability information to the customer portal for review

Appendix A

Vulnerability Analysis

The vulnerability analysis step involves the documenting and analysis of vulnerabilities discovered as a result of the previous steps. This includes the analysis of out from the various security tools and manual testing techniques.

During this phase, RedTeam Security will perform the following:

- Compile the list of areas of interest and develop a plan for exploitation
- Search and gather known exploits from various sources
- Analyze the impact and likelihood for each potential exploitable vulnerability
- Select the best method and tools for properly exploiting each of the suspected exploitable vulnerabilities

Exploitation

Unlike a vulnerability assessment, a penetration test takes such a test quite a bit further by way of exploitation. Exploitation involves establishing access to the application or connected components through the bypassing and exploitation of security controls in order to determine their actual real world risk. Throughout this step, we perform several manual tests incapable of being performed through automated means, such as scanners. During a RedTeam Security penetration test, this phase consists of heavy manual testing tactics and is often the most time-intensive phase.

Exploitation may include, but is not limited to credential harvesting, buffer overflows, SQL injection, cross-site scripting, and command injection.

As part of the Exploitation phase, RedTeam Security will perform the following:

- Attempt to manually exploit any identified vulnerabilities in order to determine the level of risk and level of exploitation possible
- Capture and log evidence to provide proof of exploitation (ie: images, screenshots, configs, etc.)
- Notify the client of any Critical findings upon discovery by telephone and email
- Upload validated exploits and their corresponding evidence/information to the project portal for client review
- Perform re-testing, per client request

Reporting

The reporting phase is intended to compile, document and risk rate findings to generate a clear and actionable report, complete with evidence, for the project stakeholders. If the customer requests, a presentation of findings will occur via online meeting.

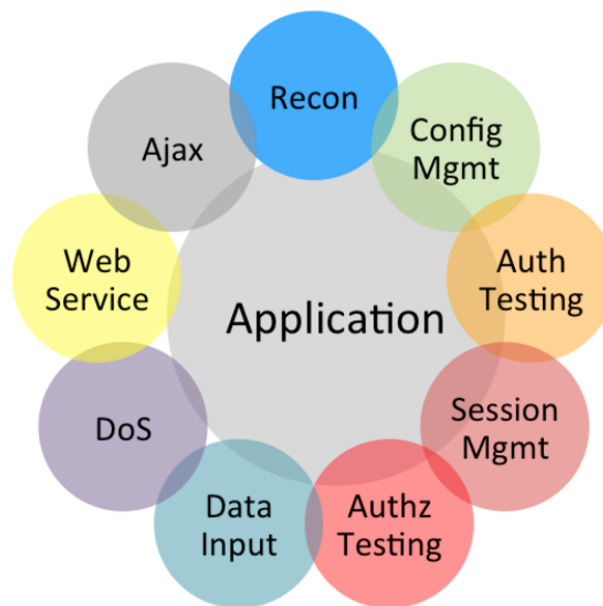
During this phase, RedTeam Security will perform the following:

- Ensure all findings have been uploaded to the project portal for client review
- Create the penetration test report, along with evidence, and conduct an internal review before uploading it to the client portal for review
- Optionally, schedule a meeting with the client to present and discuss the identified vulnerabilities

Appendix A

Comprehensive Methodology

Each web application or api penetration test is conducted consistently using globally accepted and industry standard frameworks. In order to ensure a sound and comprehensive penetration test, RedTeam leverages industry standard frameworks as a foundation for carrying out penetration tests. The underlying framework is based on the Open Web Application Security Project (OWASP).



OWASP is a globally accepted framework designed to enable the execution of effective web application penetration testing consistent with best practices all while ensuring a wholistic and comprehensive evaluation. At RedTeam Security, we consider this framework to be critical to our application and api penetration testing.

Appendix B

Risk Rating Overview

RedTeam Security has adopted an industry-standard approach to assigning risk ratings to vulnerabilities. This approach is used in all our assessments and provides our clients with risk ratings that take into account a number of factors ranging from: Skill Level needed to exploit, Motive, Ease of Exploit, Loss of Integrity to Privacy/Reputational Damage.

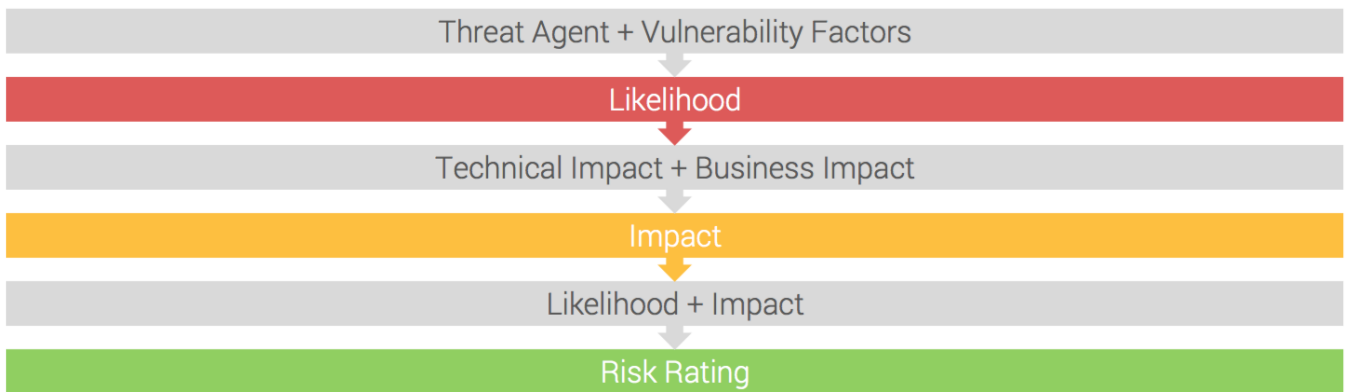
Our comprehensive approach ensures that our clients' vulnerabilities are represented by their true real-world likelihood and potential impact to their business.

Risk Rating Factors



Appendix B

Risk Calculation



Risk Calculation is carried out through a quantitative method. The calculation is an industry standard approach and is widely adopted by many organizations across the globe. Please see the detail below for a walkthrough of the risk calculation process.

Calculation of *Likelihood* is achieved by the equation:

- **AVERAGE(Threat Agent + Vulnerability Factors) = Likelihood**

Calculation of *Impact* is achieved by the equation:

- **AVERAGE(Technical Impact + Business Impact) = Impact**

Calculation of the finding's overall *Risk Rating* is achieved by the following equation:

- **AVERAGE(Likelihood + Impact) = Risk Rating**

Factors Explained

THREAT AGENT FACTORS

Factors in this category aid in establishing the real-world likelihood of exploitation. These factors take into account the knowledge required to exploit the vulnerability and breadth of the threat.

- Skill Level - How technically skilled are the group of agents
- Motive - How motivated are the group of agents
- Opportunity - What resources/opportunity are required to find/exploit
- Size - How large is the group of agents

Appendix B

VULNERABILITY FACTORS

Factors in this category aid in establishing the real-world likelihood of exploitation. Overall, these factors take into account the ease of exploitation and how well known it might be.

- Ease of Discovery - How easy is it to discover this vulnerability
- Ease of Exploit - How easy is it to actually exploit this vulnerability
- Awareness - How well known is this vulnerability
- Intrusion Detection - How likely is this to be exploited

TECHNICAL IMPACT FACTORS

Factors in this category aid in establishing the estimated impact. Overall, these factors account for potential damage to CIA (Confidentiality, Integrity, Availability) with respect to data.

- Loss of Confidentiality - How much data could be disclosed and how sensitive
- Loss of Integrity - How much data could be corrupted/damaged
- Loss of Availability - How much service could be lost and how vital is it
- Loss of Accountability - Are the threat agents' action traceable to an individual

BUSINESS IMPACT FACTORS

Factors in this category aid in establishing the estimated impact. Overall, these factors account for potential damage to the business, such as reputation, finances and privacy.

- Financial Damage - How much financial damage would result
- Reputational Damage - Would an exploit cause reputational damage
- Non-Compliance - How much does exposure does non-compliance introduce
- Privacy Violation - How much personally identifiable information could be disclosed

Appendix C

Tools

Shown below is a list of the most commonly used tools during such an engagement. RedTeam Security consultants utilize commercial, open source and RedTeam-developed tools. Be advised this is not an completed and exhaustive list and not all tools are used on every engagement. RedTeam works to select the best tool for the most accurate results for each client engagement.

Nessus	WebInspect
Metasploit	Wireshark
AppScan	Burp Suite Pro
sqlmap	testssl/ssllscan
netifera	CyberChef
OWASP Zap	Nkito
ffuf	wfuzz
dirbuster	P0F
Custom in house tools	Brutus
Hydra	John the Ripper
hashcat	Maltego