B5
Security Assessment

Project No. 378.2101
Report
FINAL

for

Agilebits Inc dba 1Password
4711 Yonge St., 10th Floor
Toronto, ON M2N 6K8
Canada

Document Versions and Changes

Version	Author	Date	Comment
0.1	Bruno Kirschner	2022-03-18	Initial draft
0.2	Bruno Kirschner	2022-03-24	Additions
0.3	Micaela Ranea-Sánchez	2022-03-25	Additions
0.4	Micaela Ranea-Sánchez	2022-03-29	Additions
0.5	Andreas Lindh	2022-03-30	Technical review
0.6	Micaela Ranea-Sánchez	2022-03-30	Clarifications
0.7	Nico Lindner	2022-03-31	Editorial review
0.8	Nico Lindner	2022-04-27	Incorporating feedback
0.9	Bruno Kirschner	2022-05-03	Retest of finding `378.2101.002`
0.10	Micaela Ranea-Sánchez	2022-05-09	Retest of finding `378.2101.003`
0.11	Nico Lindner	2022-05-10	Review
0.12	Micaela Ranea-Sánchez	2022-05-16	Fix typos
1.0	Nico Lindner	2022-05-17	Final version based on v0.12

Table of Contents

Terms and Definitions

Term	Definition
2FA	Two-Factor Authentication
API	Application Programming Interface
AWS	Amazon Web Services
BSI	Bundesamt für Sicherheit in der Informationstechnik
CSV	Comma Separated Value
DoS	Denial-of-Service
EC2	Amazon Elastic Compute Cloud
GUI	Graphical User Interface
ID	Identification
IETF	Internet Engineering Task Force
JSON	JavaScript Object Notation
MFA	Multi-Factor Authentication
n/a	not applicable / not available
PFS	Perfect-Forward Secrecy
TLS	Transport Layer Security
URL	Uniform Resource Locator
UUID	Universally Unique Identifier

1 Executive Summary

AgileBits requested consultation by Recurity Labs to perform a time-boxed security assessment of the Web application version of their password manager, also referred to as *"B5"*. The background of this assessment is provided by the regular review policy followed by AgileBits. In conformance with this policy, the application has been reviewed by multiple other security consultancy companies in the course of the last years.

Specifically, Recurity Labs was tasked to perform an application-level security audit of the B5 application with a general focus on the user-role model of family and business accounts, paying special attention to unconfirmed and guest accounts. Due to the size and complexity of the application and the time-boxed nature of this assessment, it must be noted that Recurity Labs was not able to perform an exhaustive review of the application in its entirety. More details on the features reviewed can be obtained from section 2.3.

Summarizing the present assessment, the analysis revealed that the application can be considered to be in a good condition from an IT security perspective. No *critical* impact findings have been discovered, and the security posture of the code base seems to be strong. Nevertheless, four issues have been identified in the solution. Most noteworthy are the possibility for unconfirmed members of a family to fetch metadata from business-internal vaults, as described in detail in section 3.2, and secrets which remain in the `sessionStorage` after an aborted log-in attempt, as described in section 3.4. The TLS configuration of the AWS servers in place for the testing environment were found to support cipher suites not providing Perfect Forward Secrecy (PFS), and use the `SHA1` hashing algorithm, as described in section 3.1. In addition, the Group API was found to allow users to fetch CSV files from the server, containing potentially dangerous values, which are neither sanitized nor encoded, as described in section 3.3.

All findings, including detailed reproduction steps and recommendations, can be found in section 3.

Apart from the findings described above, analyzing the provided documentation and source code, it is clear to Recurity Labs that AgileBits could benefit from improving the Access Control implementation of *B5*, as described in section 4.1. This and other recommendations are listed in section 4.

Feedback provided by AgileBits

AgileBits provided feedback on the findings via Email on 2022-04-14 and requested the incorporation into this report. The feedback has been appended to the respective findings' sections in chapter 3 and commented upon by Recurity Labs, where applicable.

Retest Status (May 2022)

After the initial assessment, AgileBits provided fixes for two (`378.2101.002` & `378.2102.003`) of the four identified issues. Recurity Labs confirmed their mitigation as part of an intermediate retest cycle, denoted in this document as *Retest Status (May 2022)*. Further details are provided as additions to the relevant subsections. Findings not in-scope of the retest activities (`378.2101.001` & `378.2102.004`) have been marked as *Open*.

1.1 Table of Findings

The following table summarizes the findings Recurity Labs made during the assessment. Each finding is briefly described by its title, its type as well as the effort and impact of a successful exploitation. Technical details for the individual findings are provided in the respective sections of chapter 3 of this document. Details regarding Recurity Labs' rating scheme are provided in section 5.

ID	Description	Chapter	Effort	Impact	Retest
378.2101.001	Weak TLS Configuration	3.1	High	Low	Open
378.2101.002	Unconfirmed Family Member Access Other Members Metadata	3.2	High	Medium	Closed
378.2101.003	Potential CSV Injection	3.3	High	Low	Closed
378.2101.004	Secrets in SessionStorage Stay Accessible After Login Abortion	3.4	High	High	Open

2 Project Background

AgileBits requested Recurity Labs to perform a security review of the B5 Web application, as part of their regular review policy. As this is the first time the application is assessed by Recurity Labs, Recurity Labs was tasked to perform an application-level security audit of the whole application with special focus on certain features, as listed in 2.3.

2.1 Team

The security assessment has been conducted between March 14th and March 30th in 2022 by Micaela Ranea-Sánchez and Bruno Kirschner of Recurity Labs. Support was provided by Rick van Galen and a dedicated team of developers of AgileBits, and Florian Grunert of Recurity Labs as responsible project manager.

Retest Status (May 2022)

This intermediate retest cycle has been conducted between May 06 to May 09 in 2022 by Micaela Ranea-Sánchez and Bruno Kirschner of Recurity Labs.

2.2 Analyzed System

The tests were performed against the *B5* test environment available at the domain `b5test.com`. The following subdomains were available:

- `api.b5test.com` for the API
- `app.b5test.com` to serve the static assets of the application
- `share.b5test.com` for the sharing service

User accounts were created by Recurity Labs by utilizing the following email addresses, provided in the present report to aid AgileBits in their cleanup tasks:

- `bruno@recurity-labs.com` to `bruno5@recurity-labs.com`
- `mica@recurity-labs.com` to `mica5@recurity-labs.com`

The assessment utilized the following build and source code excerpt as determined by the commit number provided by AgileBits:

```
6c29ebeb08f1fac94ab25543ca3d6cc629da44d4
```

The following output from the tool `scc`[1] provides a high-level overview of the source code made available:

```
-------------------------------------------------------------------------------
Language                 Files       Lines     Blanks    Comments      Code
-------------------------------------------------------------------------------
Go                       13696     4273775     403269      577415   3293091
TypeScript                2232      334926      33879       12902    288145
Go Template                889       86742       7060         936     78746
Markdown                   723       76817      20374           0     56443
SVG                        470         767         29           1       737
Sass                       413       38582       6544         431     31607
License                    398       28723       5091           0     23632
JSON                       340       79328        133           0     79195
YAML                       267       34363       1818        1120     31425
SQL                        249        9490       1758         437      7295
gitignore                  188        2386        427         304      1655
Assembly                   179       22934       4291           0     18643
Makefile                   130        6691       1279         410      5002
JavaScript                 103        7390        557        1314      5519
```

1 https://github.com/boyter/scc

Shell	85	10101	849	945	8307
Plain Text	67	40698	2845	0	37853
BASH	25	885	121	95	669
CSV	19	278	10	0	268
Protocol Buffers	19	2775	476	288	2011
Dockerfile	18	381	72	26	283
TypeScript Typings	12	641	84	81	476
Gherkin Specificati…	9	451	92	2	357
XML	9	12164	54	227	11883
Systemd	8	181	16	0	165
TOML	8	350	45	56	249
C	7	313	64	65	184
HTML	7	448	22	8	418
Gemfile	6	18	6	0	12
CSS	4	29	4	2	23
Docker ignore	4	23	0	0	23
Python	4	667	107	37	523
Bazel	3	129	10	0	119
Happy	2	4350	295	0	4055
Powershell	2	24	6	16	2
Vim Script	2	2	0	0	2
C Header	1	485	39	335	111
Total	20598	5078307	491726	597453	3989128

Additional project-related documentation was not made available, but instead, AgileBits provided *testing notes*[2] to simplify the test and account setup, and to narrow the focus of the assessment, as well as a short tutorial video on how to manipulate requests during the course of an active user session.

A general overview of the Web Applications API, based on the OpenAPI[3] standard, was generated by Recurity Labs utilizing scripts available as part of the source code.

A Slack channel was provided by AgileBits to ensure an efficient communication between the consultants and the development team.

Retest Status (May 2022)

The retest was based on an updated source code excerpt labeled as `b5-release-1238`, which did not include a specific commit ID. This update was provided by AgileBits and was expected to include fixes for the findings with IDs `378.2101.002` (see section 3.2) and `378.2101.003` (see section 3.2).

2 File name testing_notes.md, SHA256 86c92478320e6be7338a038a8e609d28524d1a3cf6f037553b039d2b7c7f66a5
3 https://www.openapis.org/

2.3 Procedures

The audit was performed in the timeframe of March 14th to 30th, 2022 in a total of 20 person-days.

A kick-off meeting was held remotely on March 3rd in 2022 to ensure a smooth beginning of the assessment. The following participants were present:

- Rick Van Galen, Aidan Woods, Neal Fennimore & Mohamed Mostafa of AgileBits
- Bruno Kirschner, Florian Grunert & Micaela Ranea-Sánchez of Recurity Labs

The following topics have been discussed:

- Scope and priorities
- Description of new material added to the shared vault
- Brief introduction to setting-up Duo MFA on a business account

The actual security assessment followed a mixed approach, where both a source code review and dynamic testing were performed, with the main objective to uncover weaknesses and vulnerabilities. As a general guideline, OWASP's *Top 10 Web Application Security Risks*[4] was utilized to identify common vulnerabilities, and focus was placed upon, but not limited to, the following categories:

- Access control
- General injection
- Hard-coded credentials
- Insecure random number generation
- Logical flaws
- Sensitive data exposure
- Session management

This allowed to identify an issue, relating to the generation of potentially insecure CSV files by the API (see section 3.3) as well as some general concerns regarding the documentation and reviewability of the present access control implementation (see section 4.1).

For completeness, a brief port scan was performed on the hosts in-scope. The only ports identified as *open* were `80`, utilized for backwards compatibility to redirect older clients, and `443`, to expose the solution over TLS. The supported TLS cipher suites were analyzed as well and were found to support cipher suites not providing Perfect Forward Secrecy (PFS), and to utilize the deprecated `SHA1` hashing algorithm (see section 3.1).

A large number of API endpoints are present in the solution, and assessing all of them in a time-boxed audit (such as this) is impossible. Therefore, after analyzing the existing endpoints, and the bigger context of the solution, Recurity Labs prioritized the following APIs:

- Account
- Billing
- User
- Vault

4 https://owasp.org/www-project-top-ten/

Additionally, and although the whole solution was placed in-scope of the present assessment, AgileBits kindly provided a set of *testing notes*. These notes placed the focus upon certain areas, and allowed to prioritize them in a way that is compatible with the needs of AgileBits. The following sections provide an overview of the testing efforts in regards to these areas, and are organized in descending order or priority, in accordance to the documentation provided by AgileBits.

Retest Status (May 2022)

Additionally, and as requested by AgileBits, Recurity Labs conducted an intermediate retest cycle to verify the implemented fixes for the findings `378.2101.002` (see section 3.2) and `378.2101.003` (see section 3.3).

2.3.1 Guest and Unconfirmed Users

The following excerpt from the *testing notes*, as provided by AgileBits, defined the scope in regards to this topic:

```
While we have had B5 tested before, they have usually focused on testers using business
accounts (with all features, focused at businesses) with regular users.

In this test, we'd like to ask you to focus on the following user types:

* Guest users
* Unconfirmed users

These users should generally be limited in their privileges, but in their implementation
they are like regular users with some privileges stripped away. Can you access data with
these type of users you shouldn't be able to access?
```

The implementation of endpoint handlers was reviewed in the search for hints regarding missing access validations. The review was then complemented with dynamic tests to verify potential insecure behavior. One instance was found, where unconfirmed members of a family can fetch metadata from business internal vaults (see section 3.2).

2.3.2 Personal and Family Accounts

The following excerpt from the provided *testing notes* defined the scope:

```
In addition to business accounts, we support individual and family accounts. Are there
security aspects to individual and especially family accounts that don't make sense for
the individual and family use cases?
```

The implementation of endpoint handlers was reviewed in the search for hints regarding missing access validations. The review was then complemented with dynamic tests to verify potential insecure behavior. No vulnerabilities have been identified.

2.3.3 Session Persistance

The following excerpt from the provided *testing notes* defined the scope:

```
B5 builds up complicated things in memory: an account unlock key and a session key to
authenticate to the server. That means that if you refresh or redirect B5 to another
page, that stuff is gone. You'll need to unlock your account.

However, when you use Duo 2 factor authentication, we use a mechanism that uses session
storage to retain a record of your session. B5 redirects to Duo, Duo redirects to B5,
and your session is restored.

Does this mechanism leave any trace of the session on the local machine? If so, we'd be
highly interested in learning about it.
```

In order to investigate the present topic, Recurity Labs utilized the Firefox browser in version `98.0 (64-bit)`, a self-created B5 business account, and a self-managed Duo[5] account. The properties saved to the `sessionStorage` browser storage were examined, and three main areas of concern were identified:

- browser crash
- browser cache
- tab duplication

Dynamic tests were performed to observe the behavior of the stored properties in these circumstances. No traces of the B5 properties were found in these instances, nor was it possible to force the browser to persist these anywhere in the underlying system.

Nevertheless, it was observed that, in the case that the Duo request times out, for example, and the *Back* button is utilized by the user, these properties remain in the `sessionStorage`, which potentially exposes them. In communication with AgileBits, it was specified that the functionality in-scope is planned to be redesigned. For completeness, and to aid AgileBits, the finding described in section 3.4 was added to this report.

2.3.4 Billing

The following excerpt from the provided *testing notes* defined the scope:

```
We have traditionally been quite lenient in our enforcing of how users are paying us.
What are some ways users can avoid paying for 1Password accounts?
```

The implementation of endpoint handlers was reviewed in the search for hints regarding ways to bypass payment. The review was then complemented with dynamic tests to verify potentially insecure behavior. No vulnerabilities have been identified.

5 https://duo.com/

2.3.5 Item Sharing

The following excerpt from the provided *testing notes* defined the scope:

```
This is a recent feature that allows you to share (copies of) vault items with people
that don't have a 1Password account. You can share items by a link, where the link
contains a "share secret" that is used to decrypt the item on the receiving end.

To share an item via a link, simply navigate to a vault item and click the share icon
there. Select "Share…". There you'll get a window to generate a link. This can be sent
to others, or shared with specific email addresses. When shared with specific email
addresses, the user needs to complete email validation - unless they're signed into
1Password with that email address. Finally, new entries for item sharing are included in
the Activity Log.

Architecturally item sharing uses a separately deployed service (i.e. share.b5test.com).
This is in scope of this test.

For item sharing, we'd like to know if the access control and activity log around this
can be circumvented. We're also interested in findings that would allow us to learn
about the contents or keys of what was shared.
```

Due to the time-boxed nature of the assessment, it was not possible to obtain sufficient coverage of the *Item Sharing* functionality. In the invested review time, however, no vulnerabilities have been identified.

2.3.6 Vault Invites

The following excerpt from the provided *testing notes* defined the scope:

```
When inviting users, it's possible to invite users directly into a vault. This is quite
a recent feature in B5, as it previously only allowed invitations and adding to vaults
as separate actions.

To use the new vault invite flow in an account, as an administrator navigate to the
account Settings (right side bar) -> Beta (top bar) -> Enable Beta features and Save
Settings.

This enables a UI that allows you to associate new invites to your account with a vault,
so that new users are immediately added to them upon confirmation. To see the invite
flow, navigate to an existing vault, click "Manage" under People, and click "Invite
People". This feature is only enabled if your user has the "Invite people" permission.

This feature makes use of the following new or updated routes:

* /api/v1/invitevaults
* /api/v1/invite
* /api/v2/people/confirm

Can this be used or manipulated in such a way that it allows people to be added to
different vaults?
```

The related endpoint handlers were reviewed in the search for hints regarding missing cross-vault access validations. The review was then completed with dynamic tests to verify potential insecure behavior. No vulnerabilities have been identified.

3 Findings in Detail

This section provides technical details on the findings that have been made during the security assessment. Each finding is described by its title, its type, effort and impact of exploitation. For details regarding Recurity Labs' rating scheme, please refer to section 5 of this document.

3.1 Weak TLS Configuration

Overview

ID	378.2101.001	
Type	Configuration	
Effort/Impact	High	Low
Location	AWS TLS-Configuration	
Retest	Open	

Details

The AWS EC2 instances used to host `start.b5test.com`, `app.b5test.com` and `my.b5test.com` were found to support cipher suites including the outdated hash function `SHA1`. The `SHA1` algorithm is no longer considered state-of-the-art and is officially deprecated since January 2020. More specific information regarding this deprecation are available as part of the official recommendations of the IETF[6] and the BSI[7].

The BSI recommendations furthermore discourage the usage of any cipher suites utilizing the `RSA` encryption algorithm for the key establishment. Mainly because, in contrast to other key establishment schemes (typically based upn Diffie-Hellman variants), `RSA` does not provide Perfect Forward Secrecy (PFS) in that particular use-case. PFS is a recommended property and should be enforced, as it ensures that disclosures of long-term key material do not result in a loss of confidentiality for any previously recorded encrypted communications.

The above concerns the following cipher suites offered for protocol version `TLSv1.2`:

- `SHA1` as hash algorithm
  - `TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA`
  - `TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA`
  - `TLS_RSA_WITH_AES_128_CBC_SHA`
  - `TLS_RSA_WITH_AES_256_CBC_SHA`
- `RSA` as key exchange algorithm
  - `TLS_RSA_WITH_AES_128_GCM_SHA256`
  - `TLS_RSA_WITH_AES_128_CBC_SHA256`
  - `TLS_RSA_WITH_AES_128_CBC_SHA`
  - `TLS_RSA_WITH_AES_256_GCM_SHA384`
  - `TLS_RSA_WITH_AES_256_CBC_SHA256`
  - `TLS_RSA_WITH_AES_256_CBC_SHA`

6 "Deprecating MD5 and SHA-1 signature hashes in TLSv1.2", https://datatracker.ietf.org/doc/html/draf t-ietf-tls-md5-sha1-deprecate-09

7 BSI TR-02102-2 "Cryptographic Mechanisms: Recommendations and Key Lenghts: Use of Transport Layer Security (TLS)" Version: 2021-1, https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-2.pdf

Please note that this issue was rated with a *Low* impact rather than *Informational*, as not all endpoints served by the underlying Web server expect their incoming requests to include an encrypted body. Therefore, any issue related to the transport encryption might still allow an attacker to exfiltrate some information.

Reproduction Steps

The TLS protocols and cipher suites supported by any Web server can be listed using the `nmap`[8] scanning tool, as detailed below. The following listing has been extracted from the results of running this tool, reflecting the active server configuration during the course of the security assessment. Highlights have been added by Recurity Labs to mark protocols or cipher suites considered insecure or outdated.

`app.b5test.com`

```
> nmap -sV app.b5test.com --script ssl-enum-ciphers
[...]
PORT    STATE SERVICE  VERSION
80/tcp  open  http     Amazon CloudFront httpd
443/tcp open  ssl/http Amazon CloudFront httpd
| ssl-enum-ciphers:
|   TLSv1.2:
|     ciphers:
|       TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (ecdh_x25519)
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (ecdh_x25519)
|       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (ecdh_x25519)
|       TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (ecdh_x25519)
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (ecdh_x25519)
|       TLS_RSA_WITH_AES_128_GCM_SHA256 (rsa 2048)
|       TLS_RSA_WITH_AES_256_GCM_SHA384 (rsa 2048)
|       TLS_RSA_WITH_AES_128_CBC_SHA256 (rsa 2048)
[...]
```

`start.b5test.com`

```
> nmap -sV start.b5test.com --script ssl-enum-ciphers
[...]
PORT    STATE SERVICE   VERSION
80/tcp  open  http      awselb/2.0
443/tcp open  ssl/https
| ssl-enum-ciphers:
|   TLSv1.2:
|     ciphers:
|       TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (secp256r1)
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (secp256r1)
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp256r1)
|       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (secp256r1)
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (secp256r1)
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp256r1)
|       TLS_RSA_WITH_AES_128_GCM_SHA256 (rsa 2048)
|       TLS_RSA_WITH_AES_128_CBC_SHA256 (rsa 2048)
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 2048)
|       TLS_RSA_WITH_AES_256_GCM_SHA384 (rsa 2048)
|       TLS_RSA_WITH_AES_256_CBC_SHA256 (rsa 2048)
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 2048)
```

8 https://nmap.org

Recommendation

Recurity Labs highly recommends to discontinue the usage of any cipher suites utilizing the outdated hash function `SHA1` or the `RSA` encryption algorithm for key establishment. Furthermore, it is recommended to adjust the AWS configuration to enforce the usage of `TLSv1.3` as soon as this feature becomes available.

Further information regarding proper TLS configuration is available as part of the technical guidelines published by the BSI[9]

Feedback provided by AgileBits (2022-04-14)

```
We've reviewed this finding and our TLS configuration. We consider the observation
valid, but have decided not to accept this finding considering the compatibility goals
of the current configuration.
```

Comment by Recurity Labs (2022-04-27)

n/a

Retest Status (May 2022)

Open

The present finding was not in-scope of this retest cycle and is therefore considered *open*.

9 BSI TR-02102-2 "Cryptographic Mechanisms: Recommendations and Key Lenghts: Use of Transport Layer Security (TLS)" Version: 2022-1, https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-2.pdf

3.2 Unconfirmed Family Member Access Other Members Metadata

Overview

ID	378.2101.002	
Type	Observation	
Effort/Impact	High	Medium
Location	/api/v1/vault/$vault_id/managers	
Retest	Closed	

Details

The endpoint used to request the manager meta data for business internal vaults was found to be accessible inside family accounts. Thereby, it seems to be applicable to any vault ID related to each user account associated to the same family account. This also includes any user account invited but not yet confirmed as family members, also called *unconfirmed users*, which should not be able to collect any such information about the family until the account was confirmed through the family manager.

As the resulting response of this endpoint includes information, such as the name and email address of the user account listed as vault manager, and each user inside the family account typically manages at least one personal vault, this potentially allows any unconfirmed family member to extract those meta data.

Reproduction Steps

To reproduce the behaviour described above, it is necessary to create properly structured and encrypted requests. This might be done through the developer version of the `op` binary provided by AgileBits, as shown below:

- Setup a family plan account and with one or more active users and at least one unconfirmed user.
- Add the unconfirmed account to the list of accounts accessible through the `op` binary and follow the provided instructions.

```
> ./op-internal-linux account add --email bruno1@recurity-labs.com --shorthand
bruno1_family
```

- Sign into the unconfirmed family member, which allows `op` to generate properly structured and encrypted requests on the behalf of the active account.

```
> eval (./op-internal-linux signin --account bruno1_family)
```

- Try to access the meta data of a vault of another family member.

```
> ./op-internal-linux request GET "/api/v1/vault/3byaztvdrkip5hhj65ghzdhb5m/managers"
{"managers":[{"avatar":"","email":"bruno3@recurity-
labs.com","firstName":"Recurity_Family_1","lastName":"","name":"Recurity_Family_1","stat
e":"A","type":"R","uuid":"MXOMSSJ545C4HI7S7LOAITO22M"}]}
```

- Try to access the meta data of a vault of another unconfirmed family member.

```
> ./op-internal-linux request GET "/api/v1/vault/eba3sjmstkfbvcuiwisgds6pd4/managers"
{"managers":[{"avatar":"mtlaxwkwpfbpnfiek4kyczstui.png","email":"bruno2@recurity-
labs.com","firstName":"Recurity_Personal_2","lastName":"","name":"Recurity_Personal_2","
state":"P","type":"R","uuid":"JA4BC3V3BRB65LTTCMI6NAYQDE"}]}
```

Recommendation

Recurity Labs recommends to limit the availability of the vault managers endpoint to the expected use-case inside business accounts. If the endpoint itself is also required for family plan related use-cases, it should not be possible for an unconfirmed user to access the meta data of other family members.

Feedback provided by AgileBits (2022-04-14)

```
We have addressed this issue. It is only possible to retrieve vault managers with the
right user status in accounts where this feature is necessary.
```

Comment by Recurity Labs (2022-04-27)

Recurity Labs believe that this solution, if implemented correctly, would resolve this finding. However, to date, no retest of the issue has been performed.

Retest Status (May 2022)

Closed

The latest version of the application no longer allows to utilize this feature in family accounts. Similar restrictions apply to unconfirmed members in accounts attached to a business plan. This could be verified by Recurity Labs both in dynamic tests and through static source code analysis. Therefore, the underlying issue is not present and the finding described above is considered mitigated.

3.3 Potential CSV Injection

Overview

ID	378.2101.003	
Type	Code	
Effort/Impact	High	Low
Location	/api/v1/group/<UUID>/members/csv	
Retest	Closed	

Details

The *B5* API allows users to export group members in the form of a comma-separated values file, also known as `CSV`, via the endpoint `/api/v1/group/<UUID>/members/csv`. When a user has access to this API, providing the UUID of a group will result in the server responding with a JSON payload representing a CSV file, as shown below:

```
{"csv":"UUID,User Name,Role\nAKQXRL5B5JBTRA5FX7R4FXXR64,\"=1+1\",\"A\""}
```

The values contained in this file, however, are neither sanitized nor encoded, resulting in a CSV injection. If the username of an account in the group is crafted to contain a formula, the formula will end up - unmodified - in the resulting CSV file. When this file is then imported by the user into a spreadsheet-handling program, such as Microsoft Excel or LibreOffice, the content of the username field will be interpreted by the software as a formula. These formulas can be used for three key attacks:

- executing code on the computer of the user, by exploiting vulnerabilities in the spreadsheet software
- executing code on the computer of the user, by exploiting the user's tendency to ignore security warnings in spreadsheets downloaded from trusted applications, such as *B5*
- exfiltrating content from the spreadsheet, or other open spreadsheets

It must be emphasized that these attacks do not target the *B5* solution in itself, but rather other pieces of software running in the computer of *B5* users.

In order for an attack to be feasible, an attacker must be able to modify the username of an account in the group, and several pre-conditions must be met, and executed, by the *targeted* user:

1. The user must download a CSV file from *B5*
2. The user must open the downloaded file with spreadsheet-handling software
3. The user must convert the data into columns
4. The user must explicitly accept any warnings presented by the software regarding dangerous external content

A successful exploitation of this issue is outside the scope of *B5* itself, and heavily relies on external factors. As such, the effort and impact ratings of this findings have been adjusted accordingly.

Reproduction Steps

In order to reproduce this finding, it is necessary to create properly structured and encrypted requests, for example by utilizing the `op-internal-linux` binary, as provided by AgileBits.

The following steps can be utilized to reproduce the behavior described in the *Details* section above:

- In a browser, log-in to the application, for example, with the account `mica3@recurity-labs.com`
- In the top right of the page, click on the username and select *My Profile*
- In the top left, click on *Edit Details*
- Set the *Name* field to `=1+1` and click on *Save*
- In a shell, navigate to the location of the `op-internal-linux` binary, and ensure it is set for execution
- Add an account via the following command

```
./op-internal-linux account add --shorthand=mica3
```

- Follow the steps to register the device for the account
- Log-in to the account

```
eval $(./op-internal-linux signin --account mica3)
```

- Perform a `GET` request to the `/api/v3/account?attrs=groups` endpoint to obtain all groups for the account, as detailed below. Observe the text in **bold** for the `group UUID` (output formatted by Recurity Labs for brevity and clarity):

```
$ ./op-internal-linux request GET "/api/v3/account?attrs=groups" --account mica3
{"attrVersion":7,"avatar":"","baseAttachmentURL":"https://f.b5test.com/","baseAvatarURL"
:"https://a.b5test.com/","createdAt":"2022-03-19T00:18:29Z","domain":"my","groups":
[{"activeKeysetUuid":"r7mwbl4g5mavyahyf5k5ie4cxi","attrVersion":1,"createdAt":"2022-03-
19T00:18:32Z","desc":"Can reset user passwords if account recovery is
enabled.","name":"Recovery","permissions":4872,"recoveryKeyset":{
[...]
},"state":"A","type":"R","updatedAt":"2022-03-
19T00:18:32Z","uuid":"vbrkrom3ed6tnlmcgpbw56tvyy"},
{"activeKeysetUuid":"5fusfke2mnxxindvrspymlp5se","attrVersion":1,"createdAt":"2022-03-
19T00:18:32Z","desc":"Access to billing and account
administration.","name":"Owners","permissions":68719476482,"recoveryKeyset":{
[...]
},"state":"A","type":"O","updatedAt":"2022-03-19T00:18:32Z","userMembership":
{"createdAt":"2022-03-
19T00:18:32Z","groupUuid":"bk6z2l5klh3jbtvo4bujirlfta","memberUuid":"AKQXRL5B5JBTRA5FX7R
4FXXR64","role":"A","state":"A","updatedAt":"2022-03-
19T00:18:32Z","version":1},"uuid":"bk6z2l5klh3jbtvo4bujirlfta"},
[...]
```

- Perform a `GET` request to the `/api/v1/group/<UUID>/members/csv` endpoint utilizing the UUID of the `Owners` group from the previous step:

```
$ ./op-internal-linux request GET /api/v1/group/bk6z2l5klh3jbtvo4bujirlfta/members/csv
--account mica3
{"csv":"UUID,User Name,Role\nAKQXRL5B5JBTRA5FX7R4FXXR64,\"=1+1\",\"A\""}
```

- Save the value of the `csv` entry into a CSV file
- Import the CSV file in a spreadsheet-handling software, e.g. LibreOffice

- Observe that the formula has been resolved, as shown in Figure 1

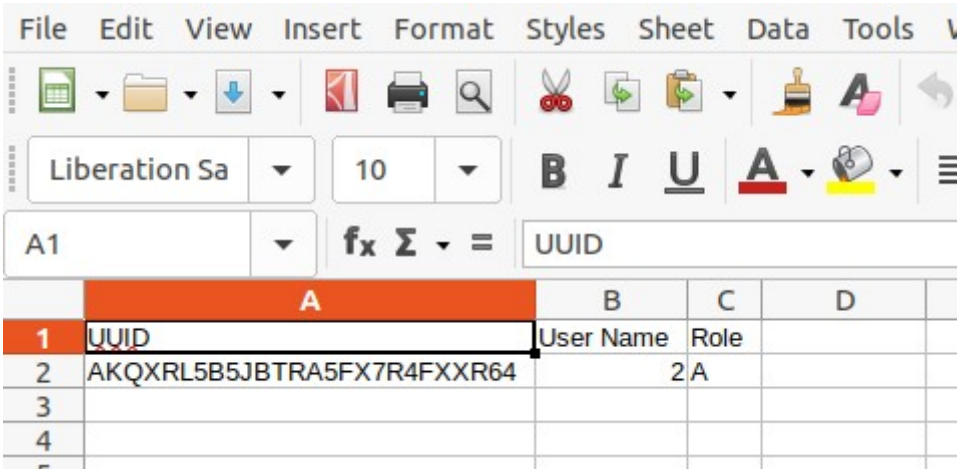

Figure 1 - CSV imported into LibreOffice

Recommendation

In order to prevent CSV injection attacks, none of the exported cells should begin with the following characters:

- Equals to: `=`
- Plus: `+`
- Minus: `-`
- At: `@`
- Tab: `0x09`
- Carriage return: `0x0D`

In addition, field separators (for example `,` or `;`) and quotes (for example `'` or `"`), should be properly encoded or escaped, as these could be used to start new cells, in such a way that dangerous characters are present in the middle of the user input, but in the beginning of a resulting cell.

Feedback provided by AgileBits (2022-04-14)

```
We have addressed this issue. Exported CSV files now contain filtering that mitigates
the described issue.
```

Comment by Recurity Labs (2022-04-27)

Recurity Labs believe that this solution, if implemented correctly, would resolve this finding. However, to date, no retest of the issue has been performed.

Retest Status (May 2022)

Closed

The latest version of the application now adds a single quote to fields that begin with the characters mentioned above, preventing formula injections. In addition, when the allowed field separators are utilized, the application utilizes double quotes to prevent new cells from being formed. This finding can be considered mitigated.

3.4 Secrets in SessionStorage Stay Accessible After Login Abortion

Overview

ID	378.2101.004	
Type	Design	
Effort/Impact	High	High
Location	DUO 2FA Login-Procedure	
Retest	Open	

Remark

The likelihood of a successful exploitation of this vulnerability is considered low as it requires direct access to a browser tab, for example via a shared computer, or another separate issue in the client application, which was previously used in an aborted logging attempt. If this access is not relevant in the attack model of AgileBits, the effort required for a successful exploitation rises drastically, which would require Recurity Labs to update the present rating of this finding accordingly.

Details

During the dynamic assessment of the *Duo*[10] -based MFA login procedure, Recurity Labs discovered that the information stored inside the `sessionStorage` object of the active tab stays accessible after the login procedure is aborted. This is even the case if the user ticked the *"This is a public or shared computer"* checkbox.

This allows an attacker with access to the browser tab initially used during the aborted log-in procedure to restore the session context stored inside the `sessionStorage`. With access to the session context, an attacker is able to circumvent the second factor and access the present user account, as the underlying session is automatically generated after the initial log-in attempt and the *Duo* based MFA authentication step is only meant to unlock it for further usage inside the Web application.

This becomes possible although the stored information is encrypted, as the key required to perform the decryption is part of the redirect URL passed during the login procedure and therefore also available through the history of the same browser tab.

Reproduction Steps

Since it is possible to exploit the issue described above through any computer with shared access, a reproduction can be achieved through the developer tools included in any modern Web browser:

- Target an account, which has *Duo* -based MFA enabled, e.g. the business account related to `bruno@ecurity-labs.com`.
- Successfully finish the password-based authentication step and abort the procedure as soon as the browser is redirected to the *Duo* layer of the authorization.
- Extract the required decryption key from the `state` query parameter present in the initial request targeting *Duo's* OAuth infrastructure, as highlighted in the following request. The related endpoint `/oauth/v1/authorize` is available, e.g. through the browser's history.

10 https://duo.com

```
https://api-1be28a34.duosecurity.com/oauth/v1/authorize?
client_id=DIUNHEBQEFQC4O65B9W5&nonce=eHxl8yH4lBICiW3bvhmCsQ&request=eyJhbGciOiJIUzUxMiIs
InR5cCI6IkpXVCJ9.eyJhdWQiOlsiaHR0cHM6Ly9hcGktMWJlMjhhMzQuZHVvc2VjdXJpdHkuY29tIl0sImNsaWV
udF9pZCI6IkRJVU5IRUJRRUZRQzRPNjVCOVc1IiwiZHVvX3VuYW1lIjoiYnJ1bm9AcmVjdXJpdHktbGFicy5jb20
iLCJleHAiOjE2NDg1NjEwODQsImlzcyI6IkRJVU5IRUJRRUZRQzRPNjVCOVc1IiwicmVkaXJlY3RfdXJpIjoiaHR
0cHM6Ly9teS5iNXRlc3QuY29tL2R1by1zaWduLWluIiwicmVzcG9uc2VfdHlwZSI6ImNvZGUiLCJzY29wZSI6Im9
wZW5pZCIsInN0YXRlIjoiQ3NDd0hnY0ctcFlXZmowRTJieTVOWXlPNUpfVmNnSVp1V3JMVEFxN3IxZyIsInVzZV9
kdW9fY29kZV9hdHRyaWJ1dGUiOnRydWV9.Y9OxxTinh5sKlqWJ6Nt_yJEObEXKc0WXavp5_Yj2rIOGv60IZA-
yBprkaN4os-NpF7zEGVChSz3V_TsXgL__Rw&response_type=code&state=%7B%22alg%22%3A%22A256GCM
%22%2C%22ext%22%3Atrue%2C%22k%22%3A%22DJUoReJ9gxjCSRMgk9Dj9rn0vLZaH90wPrg4G5dewoM%22%2C
%22key_ops%22%3A%5B%22encrypt%22%2C%22decrypt%22%5D%2C%22kty%22%3A%22oct%22%2C%22kid
%22%3A%22jysfx5i75oquwb3tp5xr3pmriu%22%7D
```

- Keep utilizing the same browser tab to return to the `b5test` subdomain used during the initial login attempt - here `my.b5test.com` - and print the content of the `sessionStorage` object through the console developer tool, as shown in Figure 2.
- For further information on how to decrypt the present session, please use the implementations as entry points:
  - **DuoV4**: `client/web-ui/src/lib/sign_in_form/mfa/types/duo_v4/duo_v4.tsx:15`
  - **duoV4verify**: `client/web-ui/src/lib/sign_in_form/mfa/types/duo_v4/duo_v4_verify.ts:13`

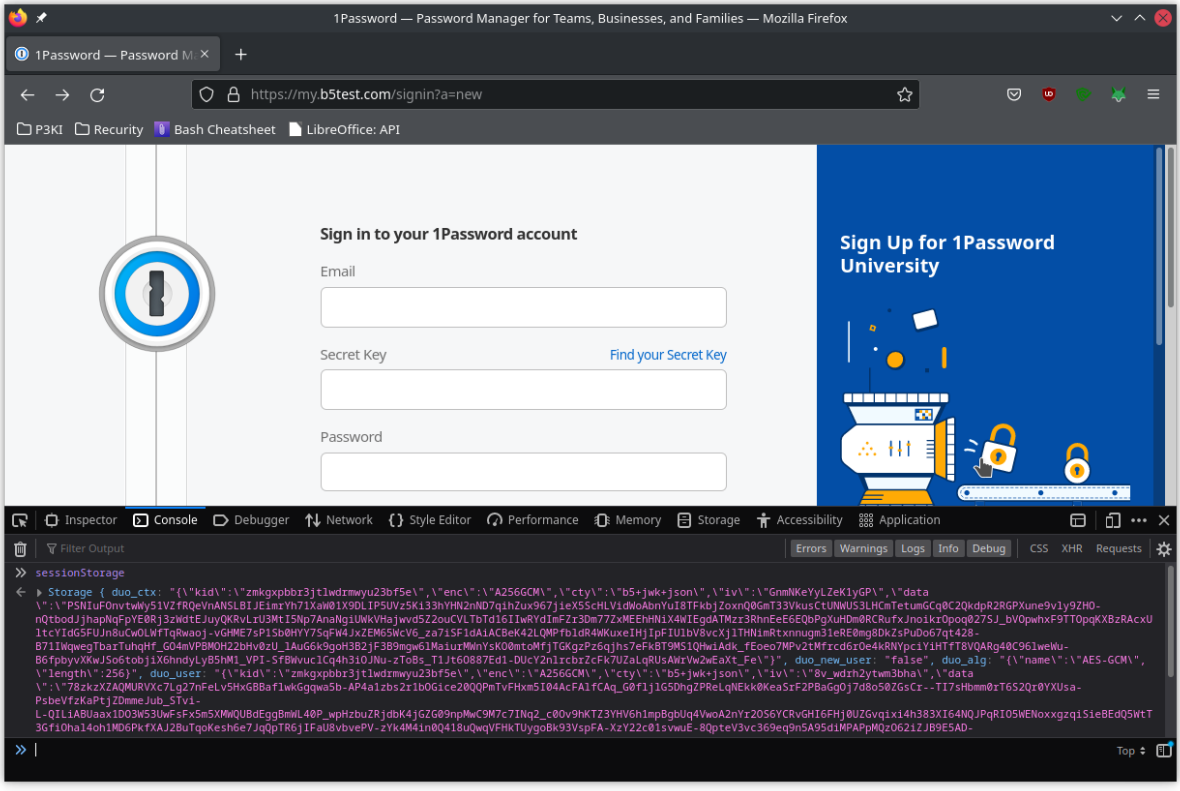

Figure 2 - The session storage still contains the encrypted session context required to bypass the 2FA layer.

Recommendation

It must be noted that, as per communication with AgileBits during the assessment, AgileBits is aware of this issue and is currently designing an improved version of this functionality. However, it was decided to include this finding for the sake of completeness.

Recurity Labs recommends AgileBits to ensure that the session context encryption - utilized to secure the *Duo* -based 2FA authentication step - includes at least one piece of data, which cannot be reconstructed through direct access to the tab used during the login procedure. Furthermore, it should be investigated if it is possible to delete the information stored inside the `sessionStorage` if no active login procedure is present.

Feedback provided by AgileBits (2022-04-14)

```
We are working on a new design of this functionality that will address the noted
limitations. We are looking to implement this in a future version of 1Password.com.
```

Comment by Recurity Labs (2022-04-27)

Once implemented, a retest of the issue is recommended.

Retest Status (May 2022)

Open

The present finding was not in-scope of this retest cycle and is therefore considered *open*.

4 General Recommendations

Recurity Labs wants to point out a number of general observations and recommendations regarding the analyzed system in the following subsections.

4.1 Centralized (Documentation of) Access Control

As initially communicated with and by AgileBits, parts of the assessment focused on the verification of the access rights granted to members of family and business accounts, especially on those access rights granted to unconfirmed and guest users, to prevent them from accessing sensitive accounts or person-related information.

Such access control-related vulnerabilities typically result from a simple mismatch between those features provided to the user - e.g. through the Graphical User Interface (GUI) - and the access verification through the actual backend API, or the discrepancy between information required to power certain user interface related features and those information actually exposed by the underlying backend API.

As such, since access control-related issues are very common in larger Web applications, there are multiple common solutions to increase the maintainability and testability of this application layer. These solutions typically introduce a single source of truth in the form of:

- an extensive and up-to-date documentation, including an access-control matrix
- a centralized access-control middleware layer

None of these are available for the B5 application.

For this reason, Recurity Labs encountered several potential issues during the course of the assessment, which required the consultation with one or multiple employees of AgileBits to determine if the observed behaviour has to be classified as vulnerability or expected behaviour, finally resulting in the issue described in section 3.1.

Hence, Recurity Labs highly recommends to verify if it is possible to either introduce one of the solutions mentioned above - or another way - to improve the current situation. This would not only increase the verifiability in the context of a security assessment, but would likely also facilitate any future API-related development.

4.2 Deny-list for Email Provider Domains

During the course of the assessment, it was identified that certain domains are not allowed in the *team invite* functionality, to prevent any user with an email registered at one of these domains to sign-up for a team with which they do not actually have any association. This list includes domains for some of the biggest public email providers, such as `aol.com`, `gmail.com`, `protonmail.com` etc.

While no weaknesses could be identified with this functionality, such types of deny-lists are usually not recommended, as it is practically impossible to foresee all items that need to be included. For instance, it was identified that the popular domains `gmx.com`, `online.de` and `web.de` (to cite a few examples) are not included in the list. Recurity Labs recommends, as a first measure, to update the list adding all domains corresponding to well-known email providers, including regional ones. In addition, it is recommended to review this functionality altogether to check whether there are alternatives to this approach, which would prevent the need for maintaining a list of unauthorized domains.

5 Explanation of Classification

This section provides a description of Recurity Labs' vulnerability rating scheme. Each finding is rated by its title, its type as well as the effort and impact of a successful exploitation. The meaning of the individual ratings is provided in the following sub-sections.

5.1 Type

The type of the result is explained in the following table:

Rating	Description
Configuration	The finding is a misconfiguration resulting in security issues.
Design	The finding is the result of a design flaw.
Code	The finding is a flaw in the source code of the object in-scope.
Observation	The finding is an observation reported for the sake of completeness.

5.2 Effort

The effort classification represents both knowledge and skills of a potential attacker as well as the availability of tools and technical resources. Here, the maximum of all three requirements is decisive.

Rating	Description
Extensive	The attack is only feasible with extremely high capabilities. The attack can most likely be performed by federal and multinational attackers only.
High	The attack can only be performed effectively by specialists within several months. In single cases, lower efforts are possible.
Medium	The attack can be performed effectively by specialists within several weeks. In single cases, lower efforts are possible.
Low	The attack can be performed by skilled attackers instantly and requires no further arrangements.
Trivial	The attack is already automated or can be performed with standard tools. Further special skills are not required.

5.3 Impact

The impact always depends on the actual object of research and is not comparable to impacts discussed in other documents.

Rating	Description
Critical	The vulnerability is a systemic error or permits compromising the system completely and beyond the scope of the assessment.
High	The vulnerability permits compromising the systems in-scope completely.
Medium	The vulnerability does not exceed security rules but permits the enumeration of other systems or enables to DoS them.
Low	The vulnerability has no direct security consequences but provides information that can be used for or aid in subsequent attacks.
Informational	The vulnerability provides data about internal processes within the system in-scope or can be used to obtain further information about the system.