Fine penetration tests for fine websites

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Pentest-Report Teleport 2.6.0 05.2018

Cure53, Dr.-Ing. M. Heiderich, M. Wege, MSc. N. Krein, J. Larsson, BSc. J. Hector,
BSc. T.-C. “Filedescriptor” Hong, N. Hippert, Dipl.-Ing. A. Aranguren, Dipl.-Ing. A. Inführ

Index

Introduction

“Teleport is a modern SSH system for managing privileged access across clusters of Linux servers. It is designed for information security best practices, including short-lived certificates and role-based access control.”

From https://gravitational.com/teleport

This report documents the findings of a second-run, source-assisted penetration test against the Teleport software maintained by Gravitational. Specifically in scope were all three editions of Gravitational, namely Open-Source, Professional and Enterprise. The assessment was carried out by Cure53 in May 2018 and involved a team of eight Cure53 testers. They invested an accumulated thirty-two days of effort into the completion of this project. The testers discovered a total of five security-relevant issues which are discussed in greater detail in the following sections of this report.

The test-methodology employed for the project was an augmented white-box approach. This means that full access to all editions of the source code along with developer-built binaries of the software was given to Cure53. The software was deployed in several configurations, from simple local builds to running more complete setups on virtual machines. A significant number of tests was conducted against the mentioned

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

deployments. With the test comprising a second run of a wider Cure53-Teleport collaboration, it has been decided to focus on penetration testing and mostly use the source code to assist and verify such tests. In other words, only the latest source additions were individually reviewed in detail. Noteworthy efforts went into checking the software for possible web-specific problems, especially in the latest extensions.

The assessment was executed in a swift manner and suffered from only a minor delay at the very final stage. Each of the spotted issues was directly communicated to the Teleport team via the dedicated messaging channels and appropriately filed to the Gravitational Github bug tracker. All changes were published in private repositories and their binary counterparts were released as soon as possible. The effective communication allowed for almost immediate fixing and timely verification of the reported issues. Moreover, the efficient exchanges between all involved parties led to a rather complete coverage of components and additionally translated to fully verified modifications being deployed at the end of the test cycle.

The test strongly focused on the newly added *PAM* integration and the refactored TLS code in the Teleport Authentication Server, along with the Recording Proxy Mode, the Certificate Rotation, and the recent addition of the Licensing Module. Additional effort was invested into looking for changes relative to the first-run test, together with a brief engagement of investigating the open source modules which have not been covered by the tests previously. The components were checked for file and log leakage but found to be properly handling all cases. Authentication flows from simple password and 2FA to SSO were checked but none of the usual problems could be identified. Role endpoints were also analyzed for ACL-related issues and input validation omissions enabling file system traversal. APIs were checked for split injections, revealing several issues. Recording Keys were found to be properly anonymized and thus far unleakable.

Of the five discoveries made during this test, one is considered to be a security vulnerability issue, while the other four were classified as general weaknesses. Many approaches to find vulnerabilities were taken and a multitude of false flag issues had to be dismissed before the conclusion of the project. The following pages briefly describe the general scope of the test, moving on to a detailed explanation of the individual issues, including Proof-of-Concept instructions and their corresponding mitigation strategies advice. Due to the timely involvement of the Gravitational development team, most of the issues have been closed already. Finally, a general verdict about the state of security of the software is given in the *Conclusions* section.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Scope

- **Teleport's upcoming 2.6.0 Release**
  - Git tag:
    - https://github.com/gravitational/teleport/releases/tag/v2.6.0-beta.2
  - Teleport binaries (the beta-3 binaries with the fixes are starred in Slack):
    - https://s3.amazonaws.com/clientbuilds.gravitational.io/teleport/2.6.0-beta.2/teleport-v2.6.0-beta.2-darwin-amd64-bin.tar.gz
    - https://s3.amazonaws.com/clientbuilds.gravitational.io/teleport/2.6.0-beta.2/teleport-v2.6.0-beta.2-linux-amd64-bin.tar.gz
  - Enterprise binaries (the beta-3 binaries with the fixes are starred in Slack):
    - https://s3.amazonaws.com/clientbuilds.gravitational.io/teleport/2.6.0-beta.2/teleport-ent-v2.6.0-beta.2-darwin-amd64-bin.tar.gz
    - https://s3.amazonaws.com/clientbuilds.gravitational.io/teleport/2.6.0-beta.2/teleport-ent-v2.6.0-beta.2-linux-amd64-bin.tar.gz
- **Scope Items**
  - **Module 1:** Gravitational Teleport (Open Source Edition)
    - https://github.com/gravitational/teleport.git
  - **Module 2:** Gravitational Teleport (Enterprise Edition)
    - https://github.com/orgs/gravitational/teams/cure53 (Group membership)
    - https://github.com/gravitational/teleport.e.git (Private repository)
  - **Module 3:** Gravitational Usage Reporting & Licensing
    - https://github.com/gravitational/reporting
    - https://github.com/gravitational/license
- **Module 4:** Extraneous Gravitational Components
  - https://github.com/gravitational/configure
  - https://github.com/gravitational/form
  - https://github.com/gravitational/kingpin
  - https://github.com/gravitational/roundtrip
  - https://github.com/gravitational/trace
  - https://github.com/gravitational/ttlmap

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Identified Vulnerabilities

The following sections list both vulnerabilities and implementation issues spotted during the testing period. Note that findings are listed in a chronological order rather than by their degree of severity and impact. The aforementioned severity rank is simply given in brackets following the title heading for each vulnerability. Each vulnerability is additionally given a unique identifier (e.g. *TLP-02-001*) for the purpose of facilitating any future follow-up correspondence.

TLP-02-001 Enterprise: *Role names* allow directory traversal *(High)*

When creating a new *role* through the Web UI, it is possible to trigger a directory traversal vulnerability. This flaw allows to create a file named *params* anywhere on the filesystem. Using the *name* field for the *new role,* an arbitrary path can be specified using a dot-dot-slash sequence (*../*). The Proof-of-Concept (PoC) below shows how this can be abused to create a file in */etc/cron.daily*. Although this does not directly lead to code execution because the created *params* file is not executable (this depends on the target system), the problem quite nicely demonstrates that file creation is a dangerous matter. Further and more creative attack ideas might ensue and be successful.

PoC Request:

```
POST /v1/enterprise/resources HTTP/1.1
Host: work:3080
[...]
Cookie:
grv_csrf=c77567d9270f9b01071c19ba0e9bde461455a5f223c4d7edda27805a86544004;
session=7b2275736572223a2274657374222c22736964223a22636237363134323231303461393930
306139653433326663313233376439326562227d
Connection: close

{"kind":"role","content":"#\n# Example resource for a role\n#\nkind:
role\nversion: v3\nmetadata:\n  # insert the name of your role here:\n  name:
role_name/../../../../../../../etc/cron.daily\nspec:\n  [...] deny: {}\n"}
```

The request shown above will create */etc/cron.daily/params* as root. However, the control over the contents of the file is limited due to the conversion from *YAML* to *JSON*. Therefore, it did not seem possible to insert whitespace characters such as newlines into the generated file.

Missing input validation is the root cause of this issue. Shown below is the code responsible for handling the request to create a *new role.* It can be seen that the *name* value of the *role* is used without prior validation.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Affected File:
teleport-2.6.0-beta.2/lib/services/local/access.go

Affected Code:
```
func (s *AccessService) UpsertRole(role services.Role, ttl time.Duration) error
{
	data, err := services.GetRoleMarshaler().MarshalRole(role)
[...]
	err = s.UpsertVal([]string{"roles", role.GetName()}, "params",
[]byte(data), ttl)
[...]
```

Affected File:
teleport-2.6.0-beta.2/lib/backend/dir/impl.go

Affected Code:
```
func (bk *Backend) UpsertVal(bucket []string, key string, val []byte, ttl
time.Duration) error {
	// create the directory:
	dirPath := path.Join(bk.RootDir, path.Join(bucket...))
	err := os.MkdirAll(dirPath, defaultDirMode)
	if err != nil {
		return trace.Wrap(err)
	}
	filename := path.Join(dirPath, key)
	f, err := os.OpenFile(filename, os.O_WRONLY|os.O_CREATE, defaultFileMode)
[...]
}
```

It is recommended to restrict the allowed characters for *role names* to *[a-zA-Z0-9_]*, which effectively prevents the issue described above. Furthermore, incorrect usage of *UpsertVal* appears to be a more general issue (see TLP-02-002). Therefore, it would make sense to consistently disallow directory traversal sequences in *UpsertVal*'s *dirpath* variable.

Fix Note: This issue was fixed and verified during the testing phase.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Miscellaneous Issues

This section covers those noteworthy findings that did not lead to an exploit but might aid an attacker in achieving their malicious goals in the future. Most of these results are vulnerable code snippets that did not provide an easy way to be called. Conclusively, while a vulnerability is present, an exploit might not always be possible.

TLP-02-002 General: Username registration allows directory traversal *(Medium)*

Checking for further directory traversal vulnerabilities caused by the misuse of *UpsertVal* led to the discovery of another issue. This flaw has been given a rather miscellaneous rating. Notably, it was noticed that simply abusing *tctl* to create *new users* seems to already be sufficient in allowing to create *user directories* outside of the usually designated path under */var/lib/teleport/backend/web/users*. Since being able to use *tctl* already requires administrative privileges on the server, this flaw has been ascribed with a "*Medium*" severity ranking. Nevertheless, this still demonstrates another input validation issue that is also caused by the missing checks inside *UpsertVal*. The following PoC quickly demonstrates this behavior.

Example Command:

```
$ tctl users add "../../../../../../../../foobar" --roles admin
```

Contents of *stdout*-Log:

```
INFO [AUTH]      [AUTH] Created user: User(name=../../../../../../../../foobar,
roles=[admin], identities=[]) auth/new_web_user.go:319
```

Created Directory:

```
$ sudo ls -la /foobar/
total 20
drwxr-x---  3 root root 4096 May  8 14:25 .
drwxr-xr-x 28 root root 4096 May  8 14:25 ..
-rw-------  1 root root  336 May  8 14:25 params
-rw-------  1 root root   60 May  8 14:25 pwd
drwxr-x---  2 root root 4096 May  8 14:25 sessions
```

The recommendation here goes along with TLP-02-001 in that it is vital to fix *UpsertVal* and make sure that no directory traversal is possible.

Fix Note: This issue was fixed and verified during the testing phase.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

TLP-02-003 Auth: Rate limiting too aggressive on *Login* *(Info)*

The currently employed rate limiting strategy for failed login attempts is to lock an account for twenty minutes when failed attempts exceed five times. An attacker can simply keep logging in a user with incorrect credentials and the victim's account will be persistently locked out.

It is recommended to employ different rate limiting strategies for different scenarios. For login attempts in which the password is incorrect, the limiting should apply to the source IP address. For login attempts in which the password is correct but the OTP is incorrect, the limiting can be applied to the account.

TLP-02-004 General: Split API injections through *WebAPI* endpoints *(Medium)*

An unauthenticated attacker is able to trigger a split API injection by sending a request, such as a U2F sign request, through the WebAPI. The vulnerable request handlers require a *username* parameter which is used to construct the API endpoint path. Through the use of dot-dot-slash (../), it is possible to reach any endpoint that accepts POST requests. However, due to the very limited control over the parameters, it is extremely difficult to exploit these issues.

Affected File:
teleport-2.6.0-beta.2/lib/auth/clt.go

Affected Code:

```
func (c *Client) GetU2FSignRequest(user string, password []byte)
(*u2f.SignRequest, error) {
    out, err := c.PostJSON(
        c.Endpoint("u2f", "users", user, "sign"),
        signInReq{
            Password: string(password),
        },
    )
[...]
func (c *Client) AuthenticateWebUser(req AuthenticateUserRequest)
(services.WebSession, error) {
      out, err := c.PostJSON(
            c.Endpoint("users", req.Username, "web", "authenticate"),
            req,
      )
[...]
func (c *Client) AuthenticateSSHUser(req AuthenticateSSHRequest)
(*SSHLoginResponse, error) {
      out, err := c.PostJSON(
            c.Endpoint("users", req.Username, "ssh", "authenticate"),
            req,
```

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Highlighted above are the vulnerable code snippets which show that the *user* parameter is passed straight to the *Endpoint* function without prior validation. The *Endpoint* function simply takes all arguments and joins them together, using a forward slash (/) for separation.

Two Proof-of-Concept (PoC) requests below demonstrate the issue. Note that one results in a request being sent to the *upsertPassword* endpoint and the other resolves to the *generateServerKeys* endpoint. However, these attack vectors are mitigated by the ACL checks in place because the *Proxy* role does not have the correct permissions to talk to these endpoints. The latter can be derived from the error messages displayed below.

PoC1 Request:

```
POST /v1/webapi/u2f/signrequest HTTP/1.1
Host: TBD
[...]

{"user": "foo/../../../users/new_user/web/password/?s=", "pass":"Trololo123!?"}
```

PoC1 Error Message:

```
INFO             [RBAC] create access to user [namespace default] denied for
roles Proxy,default-implicit-role: no allow rule matched services/role.go:1640
```

PoC2 Request:

```
POST /v1/webapi/ssh/certs HTTP/1.1
Host: TBD
[...]

{ "user": "../server/credentials?", "password": "pass", "token": "tok",
"pub_key": "a2V5IHRvIHNpZ24=", "ttl": 1000000000 }
```

PoC2 Response:

```
{"message":"username    mismatch    \"dfa71d88-0b0e-48c6-b7ef-ae252d3d9dde.work\"
and \".work\""}
```

It is recommended to fix this issue in a manner similar to the tasks warranted for TLP-02-001. In other words, the username character set should be restricted to *[a-zA-Z0-9_]*.

Fix Note: This issue was fixed and verified during the testing phase.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

TLP-02-005 Auth: OTP seed bound to *signup* token *(Info)*

It was found that the OTP seed is generated from the *signup* token. This means that if an attacker has access to a *signup* URL, s/he can jot down the OTP seed. Then, as soon as a *user* registers their account via the *signup* token, the OTP will be known to the attacker due to the fact of sharing the same OTP seed. The difference between an attacker using the *signup* token to register themselves and only noting the OTP seed is that the sequence is completely invisible to the user.

It is recommended to use a different OTP seed on each page request instead of issuing them per *signup* token.

Fix Note: This issue was fixed and verified during the testing phase.

Conclusions

The results of this second-run Cure53 security assessment of the latest release of the Teleport software by Gravitational are once again very positive. With the first Cure53-Gravitational collaboration already yielding good results, the fact that this time the findings are few and are between across the board is very much praiseworthy. The nine Cure53 testers tasked with the completion of this project over the course of thirty-two days in May 2018 identified consistent levels of high quality code.

The total number of successful attack vectors is considered to be very low, especially given the overall complexity of the product. Only one issue was considered to be security-relevant and therefore marked with “*High*” risk marker, even though it was deemed hard to exploit. The web-related aspects of the software were found to be mostly impenetrable on this second-run, as only two minor issues could be spotted directly on the scope. None of these issues led to a loss of credentials or leakage of relevant information.

It should be mentioned that the "pro" *anonymisation/reporting* feature was only recently added to the project. Nevertheless, Cure53 decided to take a brief look at the source code in the initial phase of this security assessment. The testers managed to trace the execution path in order to see how the feature actually worked, especially addressing the question of specific incoming and outgoing data. In addition, attempts were made at leaking the generated *uuid* used as the HMAC key. Cure53 failed to achieve the latter. After a brief discussion about the key generation process and safety of the HMAC usage in general, for instance in connection to caching in */var/lib*, it has been agreed that the existing mitigations and solutions were sufficient for the envisioned security premise.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

The overall feeling of coherence caused by good design decisions, the achieved good test coverage, as well as the choices of relying on a modern implementation language, translated to the conviction that Teleport should be seen as a mature software package. The product can subsequently be recommended for general deployment in security-critical environments.

As on the first run, the team is repeatedly impressed by the professionalism and dedication of the development team, which were noted alongside the uncomplicated and pragmatic handling of all related issues. Various tasks ranging from repository access and license management to almost instantaneous fixing and updating of the reported issues were performed to the highest standard. Overall the Teleport software continues to make a good and solid impression, so the Cure53's verdict is expectedly conclusive in terms of the Teleport compound being production-ready.

Cure53 would like to thank Sasha Klizhentas, Alexey Kontsevoy, Russell Jones and the rest of the team over at Gravitational for their excellent project coordination, support and assistance, both before and during this assignment.