Teleport Security Whitepaper

Practical Analysis of and Hardening Against Compromised IdP Scenarios

By Francesco Lacerenza

DOYENSEC

CONTENT

ABSTRACT

As organizations increasingly adopt Single Sign-On (SSO) providers as the central point of authentication, reliance on Identity Providers (IdPs) has become paramount in securing access to infrastructures and sensitive data. However, recent incidents have highlighted a growing threat landscape targeting these identity ecosystems, posing significant risks to the security posture of both businesses and individuals.

This paper presents an overview of the current state of the third-party identity management security landscape. Furthermore, protection mechanisms and detection strategies that organizations can adopt to strengthen their infrastructure security are addressed in relation to Teleport. In particular, emphasis is placed on the importance of implementing an additional layer of security, over centralized identity providers, to reduce the impact of a compromised SSO provider.

Finally, the results of our technical review are discussed to draw conclusions on the evolving IdP security landscape and the role of hardening solutions, like Teleport, in the analyzed scenarios. If you are interested in actionable settings for your Teleport cluster that would enhance its protection, you can consult our Hardening Checklist.

Keywords
Single Sign-On, Identity Providers, Service Providers, Teleport Hardening

INTRODUCTION

Overview of the Identity Security Landscape

Recently, the security industry has been experiencing an unprecedented rise in cyber threats targeting Identity Providers (IdPs). As organizations increasingly rely on centralized identities for user authentication and access control on a wide range of business-critical services, the compromise of an identity provider poses increasingly severe risks to the building blocks of IT infrastructures.

When a trusted Single Sign-On (SSO) provider is compromised, the impact scales with the breadth and depth of the integration. The range of potential attacks heavily depends on the level of compromise and which Service Providers (SPs) are trusting the compromised identity provider to authenticate and authorize internal actions.

In order to better understand the different threat scenarios and analyze the complexity of such scenarios, it is important to distinguish between two levels of compromises.

1. IdP Compromise

In this scenario, we consider all cases in which a vulnerability or misconfiguration allows an attacker to obtain access to the IdP. This level of compromise affects all organizations using the identity solution and consequently, they are considered as the worst case scenarios, because of the breadth of the affected audience.

Under these circumstances, there is no difference between self-hosted and cloud solutions. In fact, self-hosted providers are not inherently more secure and in fact, may lack modern security features, or they may have vulnerabilities unknown to the vendor (zero days). For example, on October 14, 2023, Doyensec discovered a vulnerability in the popular crewjam/saml library, which was made public, via coordinated disclosure, in CVE-2023-45683. This issue allows the registration of a malicious service provider on the IdPs implementing the cited library, as the base for the functional primitives. The vulnerability described in the CVE is an injection issue which leads to Cross-Site-Scripting (XSS) in the identity provider context, during the redirection at the end of a SAML SSO Flow [1].

Similarly, cloud identity providers may introduce so-called "vulnerabilities of the cloud". According to standard cloud shared responsibility models, vulnerabilities affecting the cloud service itself are responsibilities of the cloud provider. As an example, on October 20, 2023 Okta released details [2] about an incident of unauthorized access to their support system.

2. IdP Instance and Account Compromise

Unlike the previous scenario, the compromise of a single identity provider instance involves a threat model with factors related to misconfigurations, credential leakage, and social engineering attacks against employees. These are threats that target a specific company rather than the IdP solution itself.

The SSO provider is chosen as an entry point by attackers specifically because of its trusted highly privileged position managing authentication within the target business.

Common patterns employed by attackers include: social engineering, broad-based or spear phishing campaigns, bribing employees for 2FA codes, prompt-bombing, credential stuffing, session hijacking, password spraying, access tokens leakage and counting [3] [4].

Reducing Impacts from an IdP Compromise

The recent breaches experienced throughout the industry highlight the importance of in-depth security measures and continuous monitoring to promptly detect a compromised IdP and whether an attacker is abusing it to move laterally in the organization.

No SSO provider should be assumed to be and remain secure. Consequently, the services linked to a central source of authentication should introduce mechanisms to defend the service against the possibility of a compromised IdP.

The "Defense-in-depth" approach explored within this technical review consists of implementing an additional layer of security over the identity provider in place. The extra layer needs to be present on the service providers trusting the IdP as the source of authentication for users.

A concrete example is given by services like Duo Security, usable to add an extra MFA check for critical infrastructure actions, after the normal identity verification at the IdP. The list of included actions can be extended from accessing AWS APIs and Kubernetes clusters to production databases. In conjunction, WebAuthn technology can be used to make the additional layer as user-friendly as possible, since users don't have to remember additional passwords.

Those approaches double the identity verification steps by requiring two systems as opposed to the classic single IdP-centric way. The security benefit lies in these two systems needing to be breached simultaneously, just to compromise an account and authorize actions in the service provider (SP).

This Research and the Role of Teleport

As per its mission, Teleport moves away from static credentials towards ephemeral certificates backed by biometrics and hardware identities, and stops attacker pivots with the Zero Trust design. By doing so, Teleport aims at securing access to the connected IT infrastructure.

It plays a crucial role in managing access via protocols such as SSH, RDP, HTTPS, Kubernetes APIs, and a variety of SQL and NoSQL databases. Verifying the identity of the user is fundamental in order to ensure legitimate authorization of actions within the infrastructure.

Currently, Teleport users can also log in to servers, Kubernetes clusters, databases, web applications, and Windows desktops through their organization's Single Sign-On (SSO) provider. A wide range of providers are supported, mainly focusing on standard OIDC and SAML authentication flows, with a few custom exceptions like Entra ID and GitHub [5].

Considering how crucial infrastructure security and identity verification are, it becomes paramount for Teleport to provide users with the correct mechanisms to prevent and detect threats originating from a compromised SSO provider trusted by the Teleport cluster.

Presented in this paper are the results obtained during the evaluation of various threats against Teleport (on-premise) from a compromised identity provider. The focus was the identification of attack patterns and logging capability deficiencies within a defined set of threat scenarios. Different levels of compromise and protection mechanisms were defined to study how the in-depth security layer in a Teleport cluster could affect the outcomes of a successful SSO provider compromise, against the protected infrastructure. This work was performed using release v15.0.0-dev of Teleport Enterprise.

Finally, we will discuss the results of the technical review and draw conclusions on the evolving IdP security landscape and the role of hardening solutions like Teleport, in the analyzed scenarios.

Background and Methodology

As previously mentioned, this paper aims at evaluating the potential threats and attack patterns targeting an on-premise Teleport cluster from a compromised identity provider.

Given its multi-functional and identity-driven nature, a Teleport cluster receiving non-trustworthy identities from a trusted source (IdP) may lead to numerous outcomes in term of impacts to confidentiality, integrity and availability (CIA). Consequently, this work required an approach that would take into account the different levels of protections provided by the tool, against the type of compromises seen in real-world scenarios.

In order to set the context, a brief summary on how Teleport handles SSO users and providers is presented.

Teleport and Identity Providers

Teleport clusters support logging in to resources through multiple Single Sign-On (SSO) providers. Users can register the Teleport cluster as an application within their SSO providers. When a user signs in to Teleport, the SSO provider will execute its own authentication flow, then send an HTTP request to the cluster to indicate that authentication has completed. Teleport issues short-lived certificates to users completing SSO authentication flows. As per Teleport logic, SSO users are stored as temporary users on the Auth Service backend.

Multiple providers can be used to authenticate the same SSO user, but not users registered locally. In the context of Teleport, a local user refers to a user identity that exists in the scope of a cluster and is managed directly via Teleport, rather than a third-party identity provider.

Teleport internally relies on the concept of authentication connectors to represent SSO providers. These components are configuration resources controlling how SSO users are logged in to Teleport and which roles should be assigned to the temporary user resource, according to fine-grained RBAC policies.

Multiple connector types are supported according to the documentation:

- **saml** - The SAML connector type uses the SAML protocol to authenticate users and query their group membership

- **oidc** - The OIDC connector type uses the OpenID Connect protocol to authenticate users and query their group membership

- **github** - The GitHub connector uses GitHub SSO to authenticate users and query their group membership

Since specific IdPs may require or benefit from changes to Teleport's SSO flow, the *spec.provider* property of the connector definition can be used to enable provider-specific changes. The following values are supported to match the different identity providers: **adfs** (SAML), **netiq** (OIDC), **ping** (SAML and OIDC), **okta** (OIDC) [6].

Independently from the different IdP options, rules and policies, the main objective of the research was testing the capabilities of Teleport in mitigating an SSO provider compromise. In particular, having attackers with different levels of privileges in the IdP should not influence the capability of a Teleport cluster to detect and protect against malicious activities.

Performing dynamic testing made us quickly realize that it would be necessary to analyze the IdP permission system related to CRUD (create, read, update, and delete) operations performed on the attributes used to map roles in Teleport. Teleport's documentation specifies which of these attributes should be used in each IdP as the source of roles during the mapping. As an example, for Okta the suggested attribute is the *okta* group. The research was also focused on how admins can manipulate those attributes and who normally has access to edit them.

Separately, IdP-specific analysis was required to understand the safety of the suggested username mappings in the listed IdP connectors, during auto-provisioning of users with non-existent usernames.

Evaluation Strategy

Four main areas were considered during the analysis:

- **Level of Compromise**. Based on real-world scenarios and case-study specifics, four levels of compromise were identified. These allowed us to cover the range of potential impacts, from a compromise involving anywhere from least-privileged to fully-privileged capabilities, on both the SSO provider and the Teleport cluster instance

- **Threats against the Teleport cluster**. Definition of the valuable threats, from the attacker perspective, for each described level. In this case, threats were identified by aggregating the functional areas of a Teleport cluster instance. By doing so, it was possible to analyze the potential outcomes based on the targeted feature within a specific level

- **Available protection mechanisms in Teleport**. Enumeration and evaluation of the opt-in features in Teleport that could be used to harden the infrastructure against the studied scenarios

- **Threat detection in Teleport**. Review the available range of events in Teleport, according to the studied scenarios, to determine if it is practical to observe and track malicious actions

By combining a multilayered analysis on these areas, we attempted to determine how Teleport can mitigate a potential IdP compromise.

Level of Compromise

The level of compromise defines the assumptions made on the attacker's capabilities within the identity provider.

The different compromise scenarios are ordered top-down, based on the resulting impact. Each level includes all the threats above, however repetitions exist when the exploitation conditions are different.

Fully Compromised IdP

Description: Attackers have full control over the IdP system and they can manipulate it in any possible way. Consequently, the contents of the authentication details sent via verified SSO requests are attacker-controlled. This is the case when the IdP itself was fully compromised via a set of vulnerabilities or misconfigurations.

Likelihood: *Low*. A deep understanding of the system and/or a set of novel vulnerabilities may be required to achieve the described level of control over the IdP instance. On-premise IdP instances are less exposed, but they are generally more prone to such attacks, since the security of the system is fully delegated to the customer.

Impact: *Critical*. As stated, the attacker has no limits in the usage of the compromised IdP.

Privileged IdP Account Compromise

Description: Attackers obtained access to an administrative account in the IdP. The potential actions depend on the IdP type and on the SSO users configuration in Teleport. Moreover, multiple levels of admin accounts could be present depending on the SSO provider. This case was created to explore how different IdPs and connector types affect the exploitability of the user impersonation. In particular, each SSO provider is configurable with specific attributes related to objects at the IdP level (e.g., in Okta, it is suggested to use groups). This scenario allows us to evaluate how even a limited admin account compromise of the IdP, would permit or deny the exploitation of roles mapping, needed to access arbitrary privileges within Teleport.

In particular, the following list of identity providers were considered during the evaluation:

- *Azure Active Directory (AD)*
- *Entra ID*
- *Google Workspace*
- *GitHub*
- *OIDC*
- *Okta*

Likelihood: *Medium*. Compromising an IdP admin user usually involves social engineering techniques, hence it is related to the efficiency of the human factor and the policies of a specific company managing the IT operations. Given this scenario refers to admin accounts, we would expect particular care is taken with these credentials.

Impact: *High*. Depending on the level of the compromised IdP admin user, the outcomes may vary. In general, it was observed that a great level of control over the impersonated users and roles in Teleport could be possible, from middle-management in the IdP, under certain conditions.

Unprivileged IdP Account Compromise

The level is divided in two sub-cases covering administrative and generic Teleport users mapped to the compromised IdP account.

Unprivileged IdP Account Compromise - Privileged Teleport Account Compromise

Description: Attackers obtained access to an unprivileged account at the IdP level.

The account is mapped as an administrative account in Teleport, having access to a full-set or subset of admin actions (see admin actions referenced in RFD 131 - Administrative Actions MFA [11]). Attackers are able to exploit the mapping and pivot to the Teleport cluster, to further increase the impact, within the company.

Likelihood: *Low*. Social engineering or phishing could involve a restricted target audience having access to administrative actions in Teleport.

Impact: *High*. Teleport admins usually have access to a wide range of potentially malicious actions usable to pivot into the infrastructure and increase the impact.

Unprivileged IdP Account Compromise - Unprivileged Teleport Account Compromise

Description: Attackers obtained access to an unprivileged account at the IdP level.

The account is mapped as an unprivileged account in Teleport that does not have access to admin actions. Attackers are able to exploit the mapping and pivot to the Teleport cluster, to further increase the impact within the company.

Likelihood: High. Social engineering and phishing could be involved in a wide target audience having generic access to Teleport. Basic engineering and DevOps personnel are included in that group.

Impact: Medium. While administrative actions are not present, the compromised users could still have access to business critical resources, usable to pivot and perform dangerous actions, within the infrastructure.

Threats Against the Teleport Cluster

Within each level of compromise used as the attacker context, a list of threats were defined to aggregate the potential groups of actions performable within the cluster, on a core Teleport features basis.

Users Impersonation

Impact: The possibility to impersonate any existing user / create new users within the Teleport Cluster via SSO authentication. This could be achieved by either compromising an existing SSO connector or by configuring a new connector with a compromised account, having SSO connector creation capability [12].

In this case, attackers could:

- impersonate any existing SSO user
- create new temporary users in a specific SSO role-mapping group by leveraging automatic provisioning

It should be highlighted that the impersonation case was also contextually adapted to the specific IdPs studied during the evaluation and their peculiarities.

Downgrade of the Cluster Security Level

Impact: Possibility to disable security mechanisms in Teleport. We assume the attacker has access to any SSO user in the cluster.

In this case, attackers could:

- Grant themselves more access by editing roles after impersonating an admin
- Disable security features
- Move between multiple impersonated accounts to obtain a certain role and identity
- Lock out legitimate local admins
- Bypass security features by leveraging legitimate features
- Perform many other similar malicious actions

Access Resources in the Cluster

Impact: The possibility to access internal resources mapped in Teleport. In this case, attackers could connect to servers, databases, K8s clusters or other resources.

Cross-Cluster Accessibility

Impact: The possibility to pivot into clusters trusting the one linked to the compromised IdP. The possibility to escalate the access level within the compromised cluster and bypass protections by adding a root cluster.

In this case, attackers could:

- Access leaf clusters
- Edit leaf clusters
- Configure the compromised cluster as a leaf cluster, and grant the highest permissions in it, to a malicious root cluster, controlled by the attacker
- Perform many other similar malicious actions

Manipulate and Access Cluster Applications

Impact: The possibility to obtain / modify access to Teleport applications. Note that Credential Service Provider (CSP) Authenticator apps might also be present, making the pivoting through Teleport very dangerous.

In this case, attackers could:

- Authenticate on apps, as well as CSP authenticators
- Edit apps to redirect their traffic and steal information
- Create new applications
- Perform many other similar malicious actions

Access Teleport IdP Authenticated Service Providers

Impact: Since Teleport can be configured as an IdP to authenticate Teleport users in third-party applications, other SPs could be present and the attackers could access them from a Teleport authenticated session.

In this case, attackers could:

- Move laterally as any Teleport user inside the registered SPs trusting Teleport as an IdP
- Edit SPs to redirect traffic and steal data
- Perform many other similar malicious actions

Access Retention

Impact: This category includes malicious operations that would prolong and/or hide a compromise, for example:

- A stolen session expires, but the Teleport session obtained can be refreshed without it
- An IdP user is deprovisioned, but no action is taken on the Teleport cluster
- A Teleport user is detected as a malicious actor and needs to be blocked
- An IdP is compromised

Forced User Automatic Provisioning via Username Edit

Impact: In Teleport, a user identifier passed by the IdP is required to set the temporary user name in the cluster. Customers can configure it as they wish and use any attribute available in the IdP. In particular, if the attribute is user-controlled, compromised IdP users could always change their names and consequently get a new auto-provisioned user in Teleport.

In this case, attackers could:

- Bypass MFA-based protection mechanisms. The SSO authentication to a new temporary user allows them to enroll the first MFA device while holding the role obtained via attributes mapping

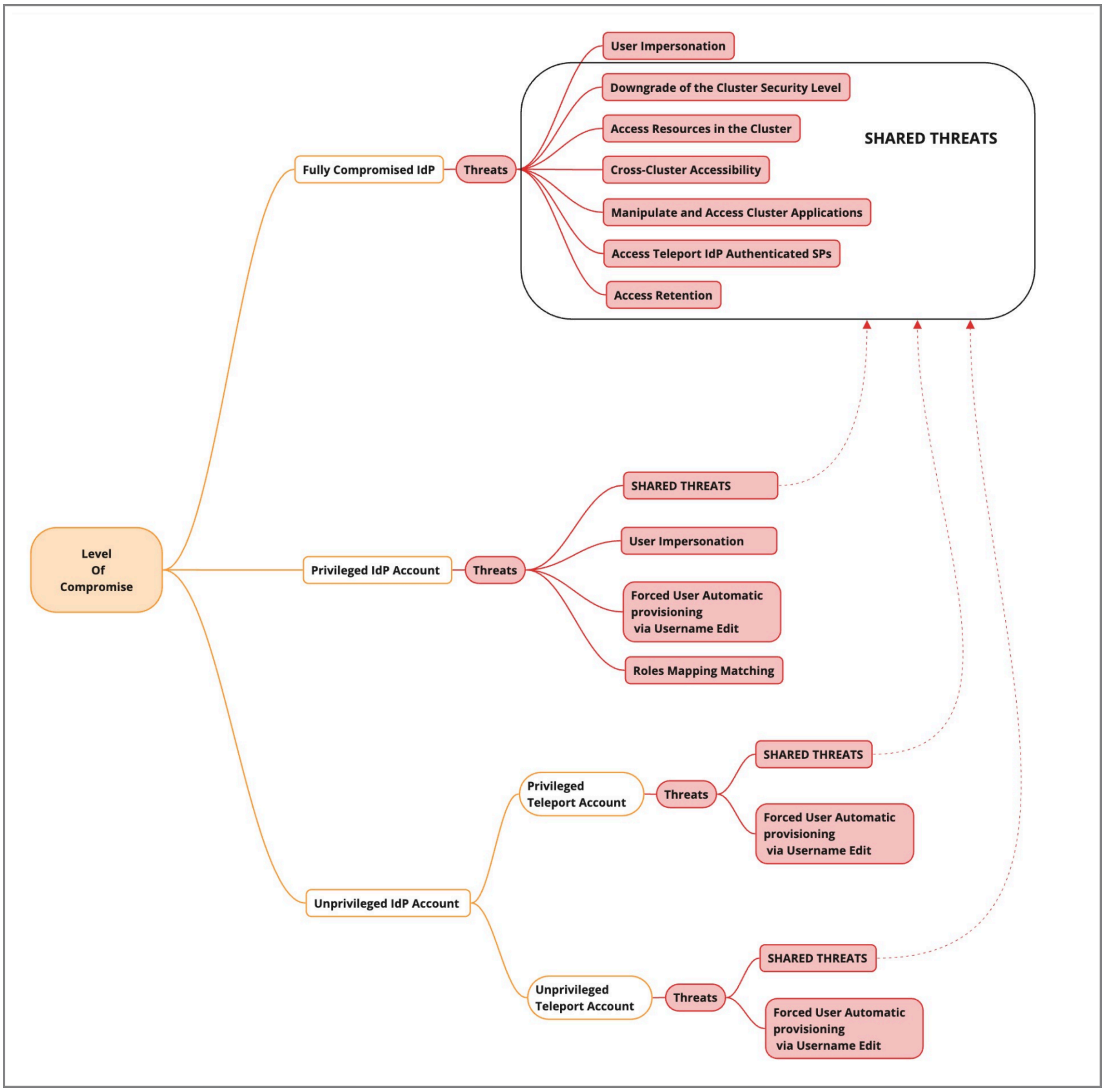
Roles Mapping Matching

Impact: Teleport requires a set of group to role mappings in order to dynamically assign privileges to a temporary SSO user. Teleport supports Regular Expressions to match attributes that should be mapped to roles. Attackers, with editing capabilities for the grouping attribute at the IdP, may rename their own group to match a regex assigned to a privileged Teleport role in the connector.

In this case, attackers could:

- Bypass the grouping attribute uniqueness constraint if a loose regex is configured in the Teleport SSO Connector

Threats Map


Level Of Compromise
Fully Compromised IdP
Threats
User Impersonation
Downgrade of the Cluster Security Level
Access Resources in the Cluster
Cross-Cluster Accessibility
Manipulate and Access Cluster Applications
Access Teleport IdP Authenticated SPs
Access Retention
SHARED THREATS
Privileged IdP Account
Threats
SHARED THREATS
User Impersonation
Forced User Automatic provisioning via Username Edit
Roles Mapping Matching
Unprivileged IdP Account
Privileged Teleport Account
Threats
SHARED THREATS
Forced User Automatic provisioning via Username Edit
Unprivileged Teleport Account
Threats
SHARED THREATS
Forced User Automatic provisioning via Username Edit

Teleport Protection Mechanisms

In this section, we introduce the protection mechanisms available in Teleport on-premise that are relevant for our analysis. This work was performed using release *v15.0.0-dev* of Teleport Enterprise.

Per-session MFA

In Teleport, per-session MFA is an advanced security feature that protects users against compromises of their on-disk Teleport certificates. Teleport requires additional multi-factor authentication checks when starting a new session for SSH, Kubernetes, databases or desktops [7].

This perfectly applies to the concept of a compromised SSO user. In this situation, the authentication via SSO is always accepted, hence the certificate is always issued to compromised SSO sources, if they bypass / control the MFA on the IdP.

Such a restriction could be enabled cluster-wide for a wider audience or per-role to target specific accesses.

An example of a Teleport YAML configuration to enable the feature cluster-wide:

```
auth_service:
    authentication:
    # require per-session MFA cluster-wide
      require_session_mfa: yes
```

An example role configuration to enable the feature on a target role:

```
kind: role
version: v7
metadata:
  name: example-role-with-mfa
spec:
  options:
    # require per-session MFA for this role
    require_session_mfa: true
  allow:
    ...
  deny: ...
```

Access Requests

Just-in-time Access Requests allow Teleport users to request access to a resource or role, depending on need. The request can then be approved or denied by a configurable number of approvers.

Access Requests can be used to implement the principle of least privilege in an organization, leaving an attacker with no permanent admins to target. Users can instead receive elevated privileges for a limited period of time. Request approvers can also be configured with limited cluster access, so they will not be high value targets [9].

Access Requests support two main use cases: Role Access Requests and Resource Access Requests. Additionally, the Access Request lifecycle in Teleport includes the following configurable areas:

- When a user must make a request
- What permissions a user can request
- How long elevated permissions can last
- How many users can approve or deny different kinds of requests

Below is an example of combining an approver role with a contractor role.

Approver YAML

```
kind: role
version: v5
metadata:
  name: approver
spec:
  allow:
    review_requests:
      roles:
      - 'app-manager-role'
```

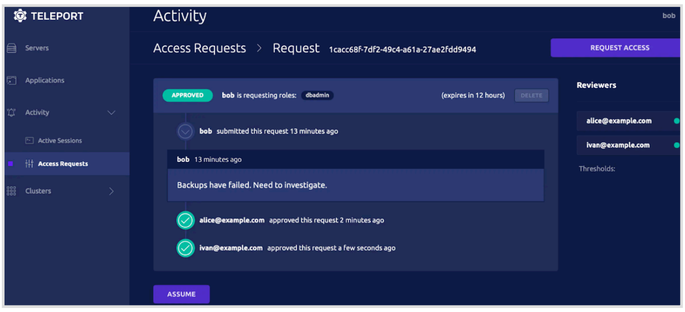
Contractor YAML

```
kind: role
version: v5
metadata:
  name: contractor
spec:
  allow:
    request:
      roles: ['app-manager-role']
```

In the presented case, the contractor is able to create an Access Request in order to assume the *app-manager-role* after the approval.

Dual Authorization

The Dual Authorization feature implements just-in-time Access Requests to require the approval of two team members for a privileged role. It improves the security of the infrastructure and prevents a single successful phishing attack from compromising it [8].



Activity

Access Requests > Request 1cacc68f-7df2-49c4-a61a-27ae2fdd9494

APPROVED bob is requesting roles: dbadmin (expires in 12 hours)

bob submitted this request 13 minutes ago

bob 13 minutes ago

Backups have failed. Need to investigate.

alice@example.com approved this request 2 minutes ago

ivan@example.com approved this request a few seconds ago

alice@example.com

ivan@example.com

By enabling this, Access Requests can be further restricted to respect certain criteria under the vigilance of multiple reviewers.

Mandatory MFA Enrollment

It is possible to set a mandatory requirement for a user to enroll an MFA device when they create an account, in order to authenticate using that device, when they begin a new Teleport session [10].

To make MFA mandatory for all users, second_factor must be set to one of the following values: *otp*, *webauthn*, or *on*.

An example YAML configuration:

```
auth_service:
  authentication:
    second_factor: webauthn
```

If MFA should be a requirement for all users while letting them choose an OTP or WebAuthn device, then the *on* option could be used, since the other options restrict users to a single type of MFA device.

It should be noted that MFA challenges, at login, only apply for local users, hence SSO users are not impacted by such enforcement.

WebAuthn

Web Authentication (*WebAuthn*) is a web standard published by the World Wide Web Consortium (W3C). It is a core component of the FIDO2 project, having the goal of standardizing an interface for authenticating users to web-based apps and services with public-key cryptography.

Teleport supports WebAuthn as a second authentication factor, usable for logging in to Teleport (`tsh login` or from the login page on the Web UI) and for logging in to individual SSH nodes or Kubernetes clusters (`tsh ssh` and `kubectl`). WebAuthn support includes hardware devices, such as YubiKeys or SoloKeys, as well as biometric authenticators like Touch ID and Windows Hello (`tsh` and Web UI).

WebAuthn has replaced U2F in Teleport from previous versions. If you haven't configured U2F before, no further action is necessary—any U2F devices are automatically supported.

MFA for Administrative Actions

Teleport features the ability to enforce an additional MFA verification for administrative actions. It is applied to administrative actions performed from any Teleport client, including `tctl`, `tsh`, the Teleport Web UI, and Teleport Connect. Adding an MFA restriction to administrative actions limits the capabilities of compromised admins by re-verifying the user's identity, promptly before performing any administrative action. The full list of protected actions is reported in the RFD 131 about administrative actions MFA [11].

The additional layer provided help in securing against the unwanted exploitation of compromised admins in Teleport. This type of approach was particularly effective in the studied IdP compromise cases. This is because existing users are no longer usable by attackers to perform administrative actions, without also compromising the second factor already enrolled in the impersonated accounts. Attacks like Org2Org mappings would leave the attacker with a valid session in Teleport, but the functionalities considered administrative would require extra MFA checks in Teleport to be approved - de facto blocking many lateral movements through Teleport.

Device Trust

The functionality allows enforcing the use of trusted devices within a Teleport cluster [17]. Resources protected by the device mode `required` will enforce the use of a trusted device, in addition to the established user's identity and enforced roles.

The enforcement mode can be configured as role-based (using RBAC) or as cluster-wide configuration. The device check supports the following resources:

- Apps (role-based enforcement only)
- SSH nodes
- Databases
- Kubernetes clusters
- First MFA Device Enrollment

In particular, the last entry is useful to protect against compromised IdP forcing the auto-provisioning of new Teleport users.

Hardening Guideline

Reducing the Impact of IdP Attacks

During the study, multiple configurations were tested on both Teleport and the linked Identity Provider solutions, in order to reproduce all discussed threat scenarios. As a result, a list of hardening steps and best practices to configure Teleport, with an external IdP, was produced to protect the cluster under all circumstances.

Just-in-Time Access Requests and Dual Authorization

As described in Teleport documentation [8], "`Just-in-time Access Requests allow Teleport users to request access to a resource or role depending on need. The request can then be approved or denied based on a configurable number of approvers`".

Such a feature can be used to implement the least-privilege principle inside the cluster and control how privileges are assumed and used. Additionally, with Dual Authorization, it is possible to enforce the approval of multiple team members to perform critical actions [8].

Configuration Suggestions

With regard to the IdP compromise scenarios studied in this research, using Access Requests and Dual Authorization, with reviewers being SSO users, is ineffective because we assume the attacker to have full or limited impersonation capabilities.

A secure configuration should have:

- Privileged roles assigned to local users only
- SSO roles mappings should follow the ephemeral admin strategy and always require them to request additional privileges
- Reviewers that are local users to prevent impersonation and self-acceptance from the IdP (In Teleport, it is not possible to authenticate as an SSO user, with a username which is part of the local users pool)

Authentication Connector and IdP Configuration Security

IdP Configuration Suggestions

As described in the Privileged IdP Account Compromise case, while Teleport should be SSO provider-independent, there are two main attributes that customers should be aware of in their IdP, when setting a connector:

- `username` field - Used for Teleport user provisioning
- `group` field - Used in the attributes-to-roles mapping in the connector configuration file

Before setting the authentication connector in Teleport, we suggest reviewing the IdP in use to find:

- A `username` field candidate, which is not editable by end-users and is unique in the IdP's users pool
- A `group` field candidate, which is editable only by a very restricted group of admin users and is unique in the organization

The listed constraints are necessary to prevent automatic provisioning of new Teleport users, forced by generic users or potential Roles Mapping Matching attacks.

Teleport Configuration Suggestions

In Teleport, a secure authentication connector should not be open to potential Roles Mapping Matching attacks (see *Privileged IdP Account Compromise*).

An example `connector.yaml`:

```
kind: saml
version: v2
metadata:
  name: corporate
spec:
[…Redacted…]
  attributes_to_roles:
    - {name: "groups", value: "okta-admin", roles: ["access"]}
    - {name: "groups", value: "okta-dev", roles: ["dev"]}
     # note that wildcards can also be used. the next line instructs Teleport
     # to assign "editor" role to any user who has the SAML attribute that begins with
"admin":
     - { name: "group", value: "admin*", roles: ["editor"] }
     # regular expressions with capture are also supported. the next line instructs Teleport
     # to assign users to roles `admin-1` if his SAML "group" attribute equals 'ssh_admin_1':
     - { name: "group", value: "^ssh_admin_(.*)$", roles: ["admin-$1"] }
[…Redacted…]
```

In the example above, any user being part of a group starting with the prefix `admin` will receive editor privileges. Congruently, any IdP user capable of creating a new group or renaming an existing one could exploit that action to get `editor` privileges in Teleport. As observed in this study, usually multiple types of admin users or team leaders have such capabilities in the connected IdP.

We strongly recommend avoiding complex string matching in role mapping definitions to prevent such attacks. Instead, fixed values should be mapped to protect against the exposed attack patterns. In fact, groups usually follow the uniqueness property at the IdP level, hence it will not be possible to have two matches by exploiting a rename operation.

Additional Identity Confirmation Layer via MFA Features

The approach, taken into consideration and explored within this technical review, consisted of implementing an additional layer of security over the identity provider in place. In order to have an additional confirmation of the incoming SSO user's identity, various MFA-based features are present in Teleport and applicable to mitigate the majority of the observed attack patterns.

The following lists the settings that are applicable for a comprehensive security posture:

- **Per-session MFA** - helps protecting session initiations with MFA requirements, but has some known Results And Limitations (see *Fully Compromised IdP - Access resources in the cluster*).

- **Administrative Actions MFA Requirement** - prevents most cases of privileges exploitation and escalations from a compromised IdP user; In Teleport v15, it is enabled by default with *WebAuthn* enabled as a second factor.

- **WebAuthn As Second Factor** - WebAuthn is a modern web authentication standard that uses physical devices (such as YubiKeys) for authentication. It offers a significant leap forward in securing user identities and preventing attacks like vishing, by adding another layer of physical authentication to verify user identities. This means that even if an attacker successfully tricks a user over the phone, they would still be unable to access accounts, without the necessary physical keys. By reducing the value and utility of stolen user information, WebAuthn can significantly mitigate the risks associated with breaches and the subsequent threats of phishing.

- **Device Trust** - Teleport decided to extend the Device Trust protection to the first MFA device enrollment in Teleport v16. By doing so, malicious auto-provisioned SSO users are no longer usable to bypass MFA-based protections in Teleport when the trusted device management and enforcement is configured.

Additional mechanisms and best practices can be found in the Teleport guide: `Reducing the Blast Radius of Attacks` [16].

SP-to-IdP Weak Spots

While studying the topic, we realized that certain mechanisms in place between the IdP-SP have a significant impact on the overall security. It should be noted that they are not strictly related to Teleport as the service provider and these aspects should be considered in any IdP-SP relationship, where the service provider wants to secure its services, in the event of an IdP compromise.

Auto-Provisioning

Auto-provisioning a user from a Single Sign-On provider (IdP) is the process typically triggered by the user's first login to a specific service provider (SP). This mechanism ensures that the user's account is created in the SP, granting instant access to IdP-sourced identities in the SP, without the need for manual intervention. While it is convenient for user access management at scale, removing the administrative burden on the various services, this research evidenced the limit it introduces, while attempting to secure the service provider. The concept of a new user is completely dependent on the attributes passed as the user ID by the IdP, making it in control of this critical action.

As a consequence, a malicious IdP trusted by the SP could be easily exploited to trigger the creation of a brand-new user at the SP - de facto bypassing MFA protection mechanisms based on pre-existing devices enrolled for a user. By design, every user must be able to set up new MFA devices on first login, which allows attackers to get their devices set on their new malicious user.

Despite being a hard-to-secure design issue, some design strategies can be applied to protect the SP without losing the advantages of SSO-provisioned users:

- **Enforce the usage of trusted devices to enroll the first MFA Device;** Device enrollment should be performed either by a device admin or by the end-user. At that step, a secure private key is created in the device and its public key counterpart is registered with the service. During the enrollment, a token created by a device admin is exchanged for the opportunity to enroll the corresponding device.

- **Implementing an ephemeral admin strategy**; Do not automatically map administrative privileges to incoming SSO users. Instead, require local admin users of the SP to approve or deny Access Requests for privileged actions. As highlighted, reviewers should not be able to authenticate as SSO users, otherwise a threat actor with full impersonation capabilities could self-approve requests.

- **Protecting administrative actions and access to critical resources** with a second layer of MFA validation. Even if bypassable with user automatic provisioning in the same role context, it is still important to protect the SP against abuse from existing users being impersonated or stolen, by any means.

Access Control Mappings Security

While a full compromise of an IdP is the worst case scenario, it is more likely for an attacker to obtain access as a privileged user at the IdP level. As explained in the Privileged IdP Account Compromise section, each of the analyzed IdPs has different roles and immutable fields usable to create access control mappings.

Consequently, it is important to configure both the IdP and the SP role mappings to minimize the possibility of role/user matching from middle management roles, like team leaders. Using immutable and unique attributes at the IdP to assign roles in the SP prevents any privileged user without special capabilities, like super-admins, to manipulate the SSO attributes and escalate privileges in the service provider.

Security Checklist

Doyensec has drafted a checklist to verify whether your Teleport cluster has all the available protections and best practices in place to enhance security against IdP compromise scenarios.

- ☐ **Just-in-time Access Requests** is configured according to the least-privilege principle; Request reviewers are only local users (i.e., No SSO users as reviewers);

- ☐ **Dual-Authorization** is set to further restrict access to administrative actions and implement the concept of ephemeral administrators. Requests reviewers are only local users (i.e., No SSO users as reviewers);

- ☐ **SSO Connectors (IdPs)** are configured to **restrict roles mappings and automatic provisioning** capabilities from non-admin IdP users
  - ☐ The `username` field (IdP-side) mapped as Teleport `username` is not editable by end-users and is unique in the IdP's users pool;
  - ☐ The `group` field (IdP-side), used to map roles in Teleport, is editable by a very restricted group of users in the IdP and is unique in the organization;
  - ☐ The Teleport SSO Connector does not apply lax string matching to map roles. Instead, fixed values from the IdP group are mapped to roles;

- ☐ **Device Trust** can be configured to **protect against new SSO users** being auto-provisioned from a compromised IdP. By enforcing it, new SSO users need to perform the **first MFA device enrollment** from a trusted device;

- ☐ **Access Lists** granting administrative permissions **(see RFD 131 [11]) do not have**:
  - ☐ **SSO identities.** Only local users should obtain high privileges via access list**;**
  - ☐ **Implicit rules referencing attributes obtained from the SSO source**;
  - ☐ **Dangling Identities** which are no longer part of the cluster;

- ☐ An **additional Identity Confirmation Layer** is applied
  - ☐ **Per-session MFA** is applied cluster-wide to restrict access to various resources with MFA devices;
  - ☐ **WebAuthn** is forced as second factor to avoid OTP-related attacks;
  - ☐ **Administrative Actions MFA Requirement** is active for admin actions, with MFA challenges;

- ☐ **Detection & Incident Response Strategies** are in place
  - ☐ There are watchdogs listening on the valuable events emitted by Teleport (Please refer to the detection section of each threat analyzed in this paper to build custom rules);
  - ☐ **Moderated Sessions** admins are configured as local users, ready to join or assess suspicious sessions;
  - ☐ Admins with SSO Connectors management and locking capabilities are ready to be used to block new malicious sessions or invalidate existing ones;

- ☐ **Teleport roles do not reference external values** taken from the IdP mappings;

Threats Analysis Results

This chapter presents the outcomes of our analysis that led to the creation of the *Hardening Guideline* and *Security Checklist* presented before.

Fully Compromised IdP

1. User Impersonation

Teleport Detection Capability

The following events can be used to identify a user impersonation attack.

Detect user creation:

- **T1002I**. UserCreateCode is the user create event code (The *connector* value should be different from *local* to indicate SSO users)

Detect SSO Connector CRUD operations:

- **T8200I**. *SAMLConnectorCreatedCode* is the SAML connector created event code
- **T8201I**. *SAMLConnectorDeletedCode* is the SAML connector deleted event code
- **T8202I**. *SAMLConnectorUpdatedCode* is the SAML connector updated event code
- **T8100I**. *OIDCConnectorCreatedCode* is the OIDC connector created event code
- **T8101I**. *OIDCConnectorDeletedCode* is the OIDC connector deleted event code
- **T8102I**. *OIDCConnectorUpdatedCode* is the OIDC connector updated event code
- **T8000I**. *GithubConnectorCreatedCode* is the Github connector created event code
- **T8001I**. *GithubConnectorDeletedCode* is the Github connector deleted event code
- **T80002I**. *GithubConnectorUpdatedCode* is the Github connector updated event code

Alternatively, events such as `oidc.*`, `saml.*` and `github.*` can be used for monitoring.

Detect SSO User Login Attempts:

- **T1001I.** *UserSSOLoginCode* is the successful SSO user login event code
- **T1001W.** *UserSSOLoginFailureCode* is the unsuccessful SSO user login event code

Detect Sessions:

- **T2000I.** *SessionStartCode* is the session start event code
- **T2001I.** *SessionJoinCode* is the session join event code

Teleport Protection Features

The following features are relevant to mitigate this threat:

- MFA for administrative actions
- WebAuthn
- Per-session MFA
- Dual Authorization
- Access Requests
- Device Trust

Results And Limitations

MFA-based protections fail to protect the cluster from a fully compromised IdP performing user impersonation.
In fact, existing SSO users without MFA devices and newly auto-provisioned SSO users can be used to bypass such protections, since they can have their first MFA device added by the attacker impersonating them.

Given the role-mapping based on incoming IdP-sourced attributes, attackers in the specified case can just create new users in the compromised IdP, assign them to the group with the highest Teleport role and add the required MFA device, after their first authentications. Then, they can exploit the given roles.

Nevertheless, the bypass of the MFA protections is not possible if the cluster has *Device Trust* configured. The functionality enforces the usage of a trusted device when the first MFA device is being added for a user. Consequently, attackers using auto-provisioned SSO users will need a trusted device to bypass the MFA protections in place.

2. Downgrade the Cluster Security Level

Teleport Detection Capability

The following events can be used to identify a security level downgrade attack.

Detect MFA Modifications:

- **T1006I.** *MFADeviceAddEventCode* is an event code for users adding MFA devices
- **T1007I.** *MFADeviceDeleteEventCode* is an event code for users deleting MFA devices

Alternatively, the events `mfa.add` and `mfa.delete` can be used for additional scrutiny.

Detect Roles Modifications:

- **T9000I.** *RoleCreatedCode* is the role created event code
- **T9001I.** *RoleDeletedCode* is the role deleted event code
- **T9002I.** *RoleUpdatedCode* is the role created event code

Detect Access Requests Operations:

- **T5000I**. *AccessRequestCreateCode* is the access request creation code
- **T5001I**. *AccessRequestUpdateCode* is the access request state update code
- **T5002I**. *AccessRequestReviewCode* is the access review application code
- **T5003I**. *AccessRequestDeleteCode* is the access request deleted code
- **T5004I**. *AccessRequestResourceSearchCode* is the access request resource search code

Detect Bot Operations:

- **TB001I**. *BotCreateCode* is the `bot.create` event code
- **TB002I**. *BotUpdateCode* is the `bot.update` event code
- **TB003I**. *BotDeleteCode* is the `bot.delete event` code
- **J001I**. Bot join operation

Detect instance join and join token creation:

- **TJ002I**. Instance join
- **TJT00I**. *ProvisionTokenCreateCode* is the event code for creating a provisioning token, also known as Join Token

Teleport Protection Features

The following features are relevant to mitigate this threat:

- MFA for administrative actions
- Dual Authorization
- Access Requests

Results And Limitations

Most of the infrastructure-critical operations available to edit the cluster configuration were found to be protected by the MFA for administrative actions feature.

The operations below were found to lack of protection when MFA for administrative actions is active. While they do not permit a full compromise, they are still valuable for an attacker in the scoped context:

- Lock management operations. Possibility to lock out the admins that should lock out the compromised accounts / remove the SSO connector etc.
- Cluster alerts. Possibility to perform internal phishing
- Applications operations. Potentially sensitive like in Cloud Authenticator applications
- Integrations operations
- Session recordings operations. Possibility to play recordings and get confidential information form them
- DB actions
- K8s editings
- Windows desktop operations
- Assuming approved access request privileges

Teleport is planning to support listed operations as opt-in MFA requirement in the future.

3. Access Resources in the Cluster

Teleport Detection Capability

Given the extended number of resource types, the listing was excluded in this case. When implementing detection, we would suggest tuning the detection capability with the respective codes found at `teleport/ib/events/codes.go`.

Teleport Protection Features

The following features are relevant to mitigate this threat:

- Per-session MFA
- Dual Authorization
- Access Requests

Results And Limitations

From the per-session MFA documentation [7], currently known Results And Limitations for the feature are:

- For SSH connections besides the Web UI, the `tsh` or Teleport Connect client must be used for per-session MFA (The OpenSSH ssh client does not work with per-session MFA)
- Only `kubectl` supports per-session WebAuthn authentication for Kubernetes
- Database access with per-session MFA only works with `tsh db connect` (Per-session MFA for databases is not supported in Teleport Connect)
- For desktop access, only WebAuthn devices are supported

4. Cross-Cluster Accessibility

Teleport Detection Capability

The following events can be used to identify unauthorized access to another cluster.

Detect Trusted Clusters CRUD operations:

- **T7000I**. *TrustedClusterCreateCode* is the event code for creating a trusted cluster
- **T7001I**. *TrustedClusterDeleteCode* is the event code for removing a trusted cluster
- **T7002I**. *TrustedClusterTokenCreateCode* is the event code for creating a new provisioning token for a trusted cluster (Deprecated in favor of [*ProvisionTokenCreateEvent*])

Teleport Protection Features

The following features are relevant to mitigate this threat:

- Per-session MFA
- MFA for administrative actions

Results And Limitations

Per-session MFA protects against unwanted resources access in leaf clusters. All the Results And Limitations of the per-session MFA feature described in the previous threat apply.

When Teleport is configured to require MFA for administrative actions, MFA is required to create, update, or delete trusted clusters. Consequently, attackers could not leverage the creation of a malicious root cluster to impose full control over the compromised one.

5. Manipulate Cluster Applications

Teleport Detection Capability

The following events can be used to identify unauthorized access to applications in the cluster.

Detect App CRUD operations:

- **TAP03I.** *AppCreateCode* is the `app.create` event code
- **TAP04I.** *AppUpdateCode* is the `app.update` event code
- **TAP05I.** *AppDeleteCode* is the `app.delete` event code

Detect App Sessions:

- **T2007I.** *AppSessionStartCode* is the application session start code
- **T2008I.** *AppSessionChunkCode* is the application session chunk create code
- **T2009I.** *AppSessionRequestCode* is the application request/response code
- **T2010I.** *SessionConnectCode* is the session connect event code
- **T2011I.** *AppSessionEndCode* is the application session end event code

Teleport Protection Features

The following features are relevant to mitigate this threat:

- MFA for administrative actions

Results And Limitations

At the time of testing, Teleport did not protect:

- CRUD operations. Such actions are not included in RFD 131 [11]. Consequently, attackers compromising a Teleport user, with the capability to manipulate applications, could exploit them and manipulate applications

Nevertheless, MFA for administrative actions includes the *join token* creation. Since it is required to create applications, attackers are not able to easily create new applications, but it is still possible to edit existing ones.

6. Access Teleport IdP Authenticated Service Providers

Teleport Detection Capability

The following events can be used to identify unauthorized access to applications in the cluster.

Detect Teleport IdP Authentication:

- **TSI000I.** *SAMLIdPAuthAttemptCode* is the SAML IdP auth attempt code

Detect SAML IdP SPs CRUD operations:

- **TSI001W.** *SAMLIdPServiceProviderCreateFailureCode* is the SAML IdP service provider create failure code
- **TSI002I**. *SAMLIdPServiceProviderUpdateCode* is the SAML IdP service provider update code
- **TSI002W**. *SAMLIdPServiceProviderUpdateFailureCode* is the SAML IdP service provider update failure code
- **TSI003I**. *SAMLIdPServiceProviderDeleteCode* is the SAML IdP service provider delete code
- **TSI003W**. *SAMLIdPServiceProviderDeleteFailureCode* is the SAML IdP service provider delete failure code
- **TSI004I**. *SAMLIdPServiceProviderDeleteAllCode* is the SAML IdP service provider delete all code
- **TSI004W**. *SAMLIdPServiceProviderDeleteAllFailureCode* is the SAML IdP service provider delete all failure code

Teleport Protection Features

The following features are relevant to mitigate this threat:

- MFA for administrative actions

Results And Limitations

Management operations (CRUD) and session initiation are protected by the extra MFA-layer.

7. Access Retention

Teleport Detection Capability

The same events listed for the *Fully Compromised IdP - User Impersonation* threat should be watched.

Teleport Protection Features

The following features are relevant to mitigate this threat:

- Identities Lock
- Authentication Connectors Delete
- Device Trust

Results And Limitations

Automatic provisioning of SSO users may allow access retention, under specific conditions, if Device Trust is not in place and the SSO connector is not removed.

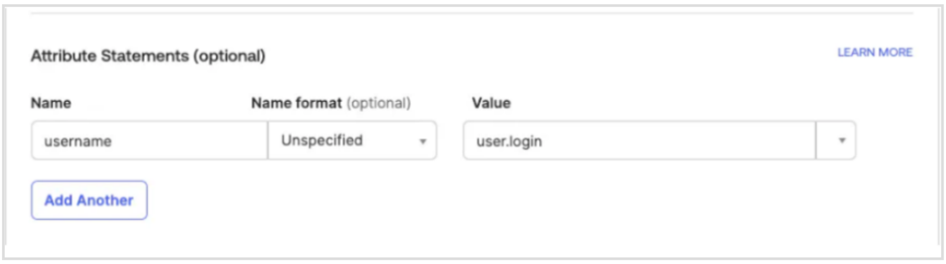
When there is a single SSO user compromise, if it is able to edit the IdP information used as *username* in Teleport, then a user lock approach will not be effective. In such cases, only the deletion of the authentication connector, linked to the compromised IdP, would prevent new authentications.

Privileged IdP Account Compromise

While Teleport is generally a SSO provider-independent solution, there are a few specific items that have relevance during a Privileged IdP Account Compromise scenario. In particular, there are two main settings that customers should be aware of in their IdP while setting up a connector.

- **Forced User Automatic provisioning via Username Edit** - Teleport requires an IdP-provided user ID to set the temporary user in the cluster. Customers can configure it as they wish and use any attribute available in the IdP.

  An example of an attribute statement setting in Okta to pass `user.login` as Teleport `username`:

  Attribute Statements (optional)

Name	Name format (optional)	Value
username	Unspecified	user.login

  Different update policies and administrative restrictions were observed on the analyzed IdPs. In particular, if the attribute is user-controlled, compromised IdP users could always change their names and consequently get a new auto-provisioned user in Teleport. As a result, the MFA devices linked to the user will not be present and attacker-controlled devices could be set to bypass the MFA protection layer in Teleport.

  The analysis was based on the username attribute, suggested by Teleport, in the documentation for each provider.

- **Roles Mapping Matching** - Teleport requires a set of group-to-role mappings in order to dynamically assign privileges to a temporary SSO user. The IdP must provide a valid group attribute, converted to a set of Teleport roles, according to the user-provided authentication connector.

  An example `connector.yaml`:

```
kind: saml
version: v2
metadata:
  name: corporate
spec:
  # display allows to set the caption of the "login" button
  # in the Web interface
  display: "Okta"
  # enables/disables idp-initiated saml login
  allow_idp_initiated: false
  # The last segment of the URL must be identical to the connector metadata name
  # when IdP-initiated login is enabled.
  acs: https://teleport-proxy.example.com:3080/v1/webapi/saml/acs/corporate
  attributes_to_roles:
    - {name: "groups", value: "okta-admin", roles: ["access"]}
    - {name: "groups", value: "okta-dev", roles: ["dev"]}
     # note that wildcards can also be used. the next line instructs Teleport
     # to assign "editor" role to any user who has the SAML attribute that begins with
"admin":
     - { name: "group", value: "admin*", roles: ["editor"] }
     # regular expressions with capture are also supported. the next line instructs
Teleport
     # to assign users to roles `admin-1` if his SAML "group" attribute equals
'ssh_admin_1':
     - { name: "group", value: "^ssh_admin_(.*)$", roles: ["admin-$1"] }
  entity_descriptor: |
    <paste SAML XML contents here>
```

In the example above, any user being part of an Okta group, starting with the prefix `admin`, will receive editor privileges. Different update policies and administrative restrictions were observed on the analyzed IdPs.

In particular, the grouping attribute is usually respecting a uniqueness constraint. Moreover, the presence of two different administrative levels capable of renaming groups was observed: `Tenant Admin` and `Team Leader`. The analysis was based on the grouping attribute, suggested by Teleport, in the documentation for each provider.

It should be noted that at this stage of a compromise, we assume that the attacker will be able to freely impersonate or create new users in any group. Because of that, Teleport protection and detection indicators should be considered as described in the *User Impersonation* threat, under the *Fully Compromised IdP* case.

In this case, for each IdP, the two sub-cases of impersonation related to the ability to spoof attributes will be analyzed.

1. Okta

Roles Mapping Matching

If the suggested Okta `group` attribute is used, attackers will have to compromise at least an Okta group administrator to perform role matching. Full impersonation capabilities are reachable only if an organization admin or super admin are compromised [13].

Since group names are unique in Okta, only Teleport authentication connectors using regular expressions to include multiple group names, in the same mapping, could be exploited by renaming a group.

Forced User Auto Provisioning via Username Edit

User updates, capable of automatically provisioning a new user in Teleport, with the same `group` attribute, are not possible if the Okta `username` was configured to be passed as the Teleport username. The Okta `username` is unique and does not change. In such a case, only the compromise of a full admin in Okta permits the impersonation as described in the previous level of compromise. Non-admin users in Okta will not be capable of automatically provisioning a new user and resetting MFA devices.

2. Google Workspace

Roles Mapping Matching

In Google Workspace, the ability to create and modify groups is typically restricted to users with administrative privileges. Additionally, users designed as owners of a group have the greatest control over the group, including the ability to rename it. According to Google's support page, `"Assigning someone the owner role gives them the greatest control over the group, so we recommend keeping the number of owners low"`[14].

Admins always get management permissions on every group.

Forced User Auto Provisioning via Username Edit

User updates, capable of automatically provisioning a new user in Teleport with the same group, are not possible if the `user.email` was configured to be passed as the Teleport `username`. The `userinfo.email` in Google Workspace refers to the primary email address of the user. This primary email address is typically set by the administrator and is the main email associated with the user's account. Users can change their secondary email address, but the primary email address is usually managed by the organization's administrator.

3. Entra ID (Azure)

Roles Mapping Matching

In Azure Active Directory (Azure AD), the management of security groups can be performed by users with the appropriate roles and permissions. According to Microsoft, `"Security Groups in Azure AD are used for managing objects in Azure AD and […] can be managed by users in the tenant that have Global Administrator, Directory Writers, Groups Administrator, Privileged Role Administrator, SharePoint Administrator, and User Administrator roles"`[15].

Forced User Automatic Provisioning via Username Edit

User updates, capable of automatically provisioning a new user in Teleport with the same group, are not possible if the Azure `user.userprincipalname` was configured to be passed as teleport username (`NameID`). Note that the domain should deny UPN updates in the first place, to have this level of protection. In such a case, only the compromise of an admin in Azure permits the impersonation as described in the previous threat scenario.

Non-admin users in Entra ID will not be capable of automatically provisioning a new user and resetting MFA.

4. GitHub

Roles Mapping Matching

Organization owners and team maintainers can access team settings and update the team's description and profile picture from the team's page. A secure configuration should not include regexes or wildcards in roles mappings.

Forced User Automatic provisioning via Username Edit

User updates, capable of automatically provisioning a new user in Teleport with the same group, are possible since the GitHub `username` can be changed. In such a case, a compromised user could have its MFA protections disabled in Teleport, by creating a new one via a username change. Without MFA protections, we know it is possible to perform any action with brand-new devices added by attackers.

5. Custom OIDC and SAML

Not applicable. Custom solutions leave the onus on the customer to verify:

- The ability to edit the IdP user attribute used to map the `role` in Teleport
- The ability to edit the IdP user attribute used to create the `username` in Teleport

Unprivileged IdP Account Compromise

1. Privileged Teleport Account Compromise

In the scenarios already explored, we assumed an attacker with access to only one account, with high privileges in a Teleport cluster. Given the extended analysis on single user capabilities performed in the *Fully Compromised IdP* level, no major differences were evidenced.

In this specific case, with respect to Access Requests and Dual-Authorization, even if the reviewers are part of the SSO users' pool, the infrastructure is protected, as long as the compromised user is not part of the reviewers group.

Additionally, locking out the compromised user is sufficient to prevent access retention, if a non-editable `username` attribute was set by the IdP admin, in the provider.

2. Unprivileged Teleport Account Compromise

In the scenarios already explored, we assumed an attacker with access to only one account without administrative privileges in a Teleport cluster. Given the extended analysis on single user capabilities performed in the *Fully Compromised IdP* level, no major differences were evidenced.

In this specific case, with respect to Identities Lock, locking out a compromised user is sufficient to prevent access retention, if a non-editable username attribute was set by the IdP admin, in the provider.

Conclusions

The technical analysis presented in this paper led to the identification of multiple attack patterns targeting an on-premise Teleport cluster from a compromised Identity Provider. Throughout this work, it was possible to demonstrate how applying a security-minded design and configuration of the Teleport cluster could significantly limit the outcomes of a successful SSO provider compromise and reduce the impact against the protected infrastructure.

Some of the security features in Teleport do provide the necessary infrastructure to protect systems and applications against the recent IdP compromises, however the user configuration plays an important role in the overall security of the cluster. In particular, it was observed that the extra MFA-layer to verify the SSO user's identity at the SP-side (Teleport in our study) was effective against malicious SSO impersonations.

A detailed hardening guide and a self-assessment checklist were written in the sections *Hardening Guideline* and *Security CheckList* to help Teleport customers defend against the vast majority of the identified attack strategies.

References

- **[1]** Doyensec, Security Advisory for CVE-2023-45683
  https://doyensec.com/resources/Doyensec_SecurityAdvisory_crewjam_saml_Q32023.pdf
- **[2]** Okta article, Tracking Unauthorized Access to Okta's Support System
  https://sec.okta.com/articles/2023/10/tracking-unauthorized-access-oktas-support-system
- **[3]** ZScaler article, Responding and Defending Against IdP Vendor Compromise
  https://www.zscaler.com/blogs/security-research/responding-and-defending-against-idp-vendor-compromise
- **[4]** Okta article, 5 Identity Attacks that Exploit Your Broken Authentication
  https://www.okta.com/resources/whitepaper/5-identity-attacks-that-exploit-your-broken-authentication/
- **[5]** Teleport documentation. Configure Single Sign-On
  https://goteleport.com/docs/access-controls/sso/
- **[6]** Teleport documentation, supported connectors
  https://goteleport.com/docs/access-controls/sso/#supported-connectors
- **[7]** Teleport documentation, Per-session MFA
  https://goteleport.com/docs/access-controls/guides/per-session-mfa/
- **[8]** Teleport documentation, Dual Authorization
  https://goteleport.com/docs/access-controls/guides/dual-authz/
- **[9]** Teleport documentation, JIT Access Requests
  https://goteleport.com/docs/access-controls/access-requests/
- **[10]** Teleport documentation, Mandatory MFA Enrollment
  https://goteleport.com/docs/management/security/reduce-blast-radius/#make-mfa-mandatory-for-tsh-login
- **[11]** Teleport RFD 131, Administrative Actions MFA
  https://github.com/gravitational/teleport/blob/master/rfd/0131-adminitrative-actions-mfa.md#administrative-actions
- **[12]** Teleport documentation, Multiple SSO Providers
  https://goteleport.com/docs/access-controls/sso/#multiple-sso-providers
- **[13]** Okta documentation, Administrators AC comparison
  https://help.okta.com/en-us/content/topics/security/administrators-admin-comparison.htm#Group
- **[14]** Google Workspace Support, Assign roles to a group's members
  https://support.google.com/a/answer/167094?hl=en#zippy=
- **[15]** Entra ID, Least privileged roles by task, Groups
  https://learn.microsoft.com/en-us/entra/identity/role-based-access-control/delegate-by-task#groups
- **[16]** Teleport Guide, Reduce The Blast Radius Of Attacks
  https://goteleport.com/docs/management/security/reduce-blast-radius/
- **[17]** Teleport documentation, Device Trust
  https://goteleport.com/docs/access-controls/device-trust/

About Doyensec

Doyensec was founded in 2017 by John and Luca who are its only stakeholders. The company exists to further the passion and focus of its creators. We aim to provide research-driven application security, enabling trust in our client's products and evolving the resilience of the digital ecosystem.

With offices in the US and Europe, Doyensec has access to a unique talent pool of security experts capable of providing worldwide consulting services.

We keep a small dedicated client base and expect to develop long term working relationships with the projects and people involved. We will find bugs, but we know that is just the first step in the process. At any stage of your security maturity, you can rely on Doyensec to solve your unique application security needs.

We value and rely on the following principles:

- **Passion**. We believe quality comes from passion and care. We love what we do, and continuously work on mastering our craft. Every engagement is finely executed with dedication and attention to details.

- **Expertise**. Our team has decades of experience in application security. We are industry leaders in penetration testing, reverse engineering, and source code review. Doyensec researchers have discovered numerous vulnerabilities in widely-deployed products, secured Fortune 500 enterprises, advised startups and worked with tech companies to eradicate security flaws.

- **Focus**. Security craftsmanship is all about individual attention and delivering tailored security services and products. We concentrate on application security and do fewer things, better.

- **Research**. The fast changing landscape of technologies and security threats requires constant innovation. We are dedicated to providing research-driven application security and therefore invest 25% of our time in building security testing tools, discovering new attack techniques and developing countermeasures.

DOYENSEC