




Insecure Zendesk SSO implementation by generating JWT client-side

Share:     


State  Resolved (Closed)


Disclosed **September 8, 2019 3:25pm +0530**

Reported To **Trint Ltd**

Asset **app.trint.com**
(Domain)

Weakness Password in Configuration File

Severity  High (8.1)

Participants 

Visibility Disclosed (Full)


[Collapse](#)

TIMELINE

**xh3n1** submitted a report to **Trint Ltd**.

Jul 10th (2 months ago)

Summary:

app.trint.com implements SSO to Zendesk, it does this by using JWT as described at <https://support.zendesk.com/hc/en-us/articles/203663816-Enabling-JWT-JSON-Web-Token-single-sign-on> 

This functionality has not been implemented securely because the JWT generation happens in the client-side. This is done by the Zendesk secret being hardcoded in the JavaScript code.

The secret is used to create JSON Web Tokens and then you can use the generated token to impersonate any customer in Zendesk. (therefore potentially getting access to their support tickets)

Whilst support.trint.com is marked as out of scope for the program, the described vulnerability isn't caused by Zendesk. The vulnerable component is in app.trint.com.

Assessment

The JavaScript source map files are available next to the minified production files. This significantly makes analyzing this issue easier.

- JavaScript file: <https://app.trint.com/static/js/app.e984c9df.js> 
- Sourcemap file: <https://app.trint.com/static/js/app.e984c9df.js.map> 

Looking at some of the UI views, I stumbled upon `static/js/modules/auth/pages/ZendeskLoadingPage.js`. I've attached a stripped version which shows the JWT generation:

```
[snip]
import { ZENDESK_DOMAIN } from 'modules/core/constants/index';

const { REACT_APP_ZENDESK_SECRET } = process.env;

[snip]

function RedirectToZendesk(props) {
  const { user, history } = props;

  function generateZendeskTokenAndRedirect() {
    const TIME_NOW_OBJECT = moment(Date.now());
    try {
      const payload = {
        iat: TIME_NOW_OBJECT.unix(),
        jti: uuid.v4(),
```

```

    name: `${user.profile.firstName} ${user.profile.lastName}`,
    email: user.username,
  });

  // encode zendesk token
  const zendeskToken = jwt.sign(payload, REACT_APP_ZENDESK_SECRET);
  window.location = `${ZENDESK_DOMAIN}/access/jwt?jwt=${zendeskToken}`;
} catch (err) {
  history.push('/error');
}
}

useEffect(
  () => {
    generateZendeskTokenAndRedirect(user);
  },
  [user],
);

return <Loader />;
}

[snip]

export default ZendeskLoadingPage;
```

Searching for `REACT_APP_ZENDESK_SECRET` in the sourcemap will show the JWT secret:

```
var REACT_APP_ZENDESK_SECRET = "oq1HJ4jXo99Wt41bwvLh9BxBVdgpI52CjkXbThow7UhwQGtJ";
```

Generating the JWT on the client-side like this allows anyone to mint an arbitrary JWT. It would probably be better to generate this on the server-side.

Reproduction steps

- As logged-in user press "Support" on <https://app.trint.com>
- Intercept the traffic and see the call to `https://trintsupport.zendesk.com/access/jwt?jwt=[JWT_TOKEN]`
- Logout of Zendesk
- Put the JWT token from above URI into <https://jwt.io> and decode it.

Example:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpYXQiOiE1NjI3MDk2NTksImp0aSI6IjIxZDAyOTg3LWU3YWItNDQ5MC05N2Q3LTc2YTBMZjJhOTV

```
{
  "iat": 1562709659,
  "jti": "21d02987-e7ab-4490-97d7-76a0f32a95c8",
  "name": "Test Test",
  "email": "b3871694@urhen.com"
}
```

- Now we can continue with tampering the JWT
 - Change IAT to the current Unix timestamp
 - Change JTI to a random UUID v4
 - Change email to the victim email address
 - Insert `oq1HJ4jXo99Wt41bwvLh9BxBVdgp152CjkXbThow7UhwQGtJ` as HMAC secret.
- Use the resulting JWT in a call to `https://trintsupport.zendesk.com/access/jwt?jwt=[JWT_TOKEN]`. You will be logged in as the victim.

Impact

Access to the Zendesk account of Trint customers. This includes potentially the support history of said user.

I haven't verified whether the same SSO flow can also be used against Zendesk administrators. If so, the risk would be higher.



tabascojellybeans HackerOne staff posted a comment.

Jul 11th (2 months ago)

Hi @xh3n1,

Thank you for your submission. Your report is currently being reviewed and the HackerOne triage team will get back to you once there is additional information to share.

Kind regards,

@tabascojellybeans



magicmouse HackerOne staff changed the status to Needs more info.

Jul 11th (2 months ago)

Hello @xh3n1,

Thanks for the report! After copying the resulting JWT, I am not logged in as the victim. Is there a step missing in your proof-of-concept?

Regards,

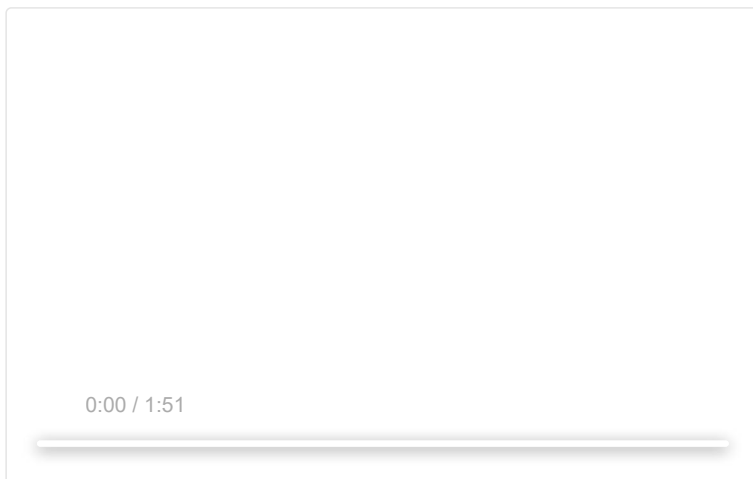
@magicmouse



xh3n1 changed the status to New.

Jul 13th (2 months ago)

The issue consistently reproduces for me. Kindly see below video for the reproduction steps:



Please do make sure that:

- `iat` is set to the current timestamp
- `jti` is set to a random UUID v4
- `email` is the victim's email address
- The JWT is quickly submitted after generation. Zendesk seems to not accept JWT with older timestamps.

Please do let me know if you require any further information to reproduce this issue.

1 attachment:

F527941: [Screen_Recording-trint.mov](#)



lollipop HackerOne staff changed the status to Needs more info.

Jul 18th (2 months ago)

Hi @xh3n1,

Thank you for your submission. This issue seems legit, but it seems that the access you have is limited since the account doesn't seem to have any activity or data apart from the ones you used in your JWT. Can you provide an email of an account which has some data so that we can verify the unauthorized access and properly evaluate the impact?

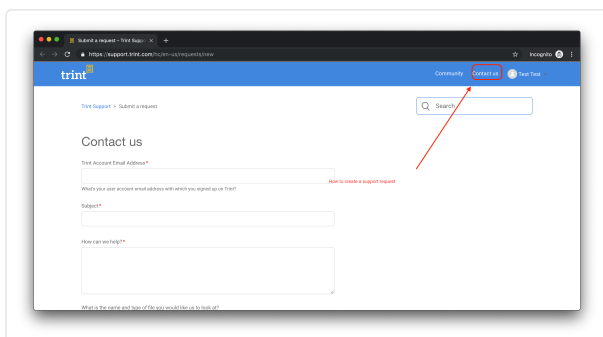
Kind regards,
@lollipop



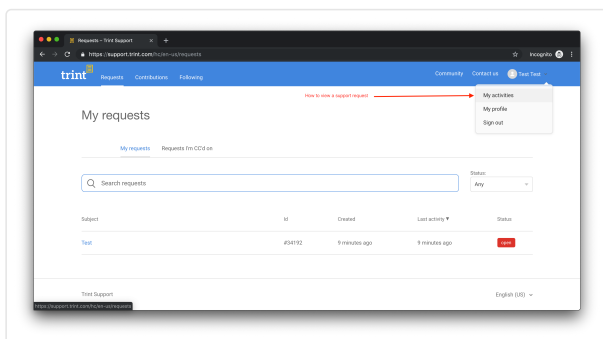
xh3n1 changed the status to ○ New.
Hi @lollipop,

Jul 19th (2 months ago)

Of course. After pressing on "Contact Us", you will end up on a support request form where you can file a request:



This request will then be associated with your account and accessible at <https://support.trint.com/hc/en-us/requests>



You could try the account "b3871694@urhen.com" and then go to <https://support.trint.com/hc/en-us/requests>. Please note that this account may already have been gotten suspended by the Trint team, so using your own one may be more reliable. (or they may have deleted this request itself :-))

Kind regards,
@xh3n1

2 attachments:

F532573: [my-activities.png](#)

F532572: [contact-us-button.png](#)

lollipop HackerOne staff updated the severity from Medium (4.4) to High (8.1).

Jul 19th (2 months ago)



lollipop HackerOne staff changed the status to ○ Triaged.
Hello @xh3n1,

Jul 19th (2 months ago)

Thank you for your submission! We were able to validate your report, and have submitted it to the appropriate remediation team for review. They will let us know the final ruling on this report, and when/if a fix will be implemented. Please note that the status and severity are subject to change.

Regards,
@lollipop



fabioantunes closed the report and changed the status to ○ Resolved.
Hello @xh3n1,

Aug 9th (about 1 month ago)

Thank you for bringing this to our attention.

We have now fixed it, and you are welcome to confirm this.

Thank you again for helping us keep our product secure.

[@fabioantunes](#)

○ [xh3n1](#) requested to disclose this report.

Aug 9th (about 1 month ago)

○ This report has been disclosed.

Sep 8th (18 hrs ago)