International JavaScript Conference

by devmio

The Magic of JWTs

Elevating Security Beyond Lock and Key to Fort Knox Levels

Sakshi Nasha

Sakshi Nasha

- **Software Engineer** by day, code wizard by night—coding one line at a time.
- **Learning addict**—because tech never sleeps, and neither do I.
- **Public speaker & community advocate**—spreading tech love with FOSS United, Hackathons, Google Developer Groups, and more.
- **Athlete at heart**—whether it's Marathons, Basketball, Badminton, or Football, I'm always up for a challenge!
- Off the grid? Catch me trekking, cycling, or just soaking up nature to **recharge for the next big idea**.


1/3
GDG Pune


Welcome


Persistent
#SIH Software Edition
Winner
Out Of Box Thinkers
One Lakh Only
₹ 1,00,000/-

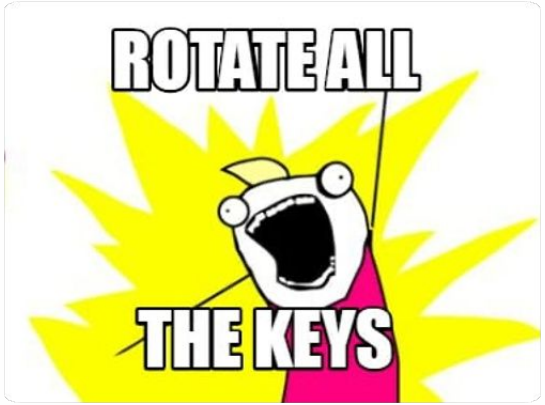
And many more to milestones
to achieve before I sleep !!!


praise me

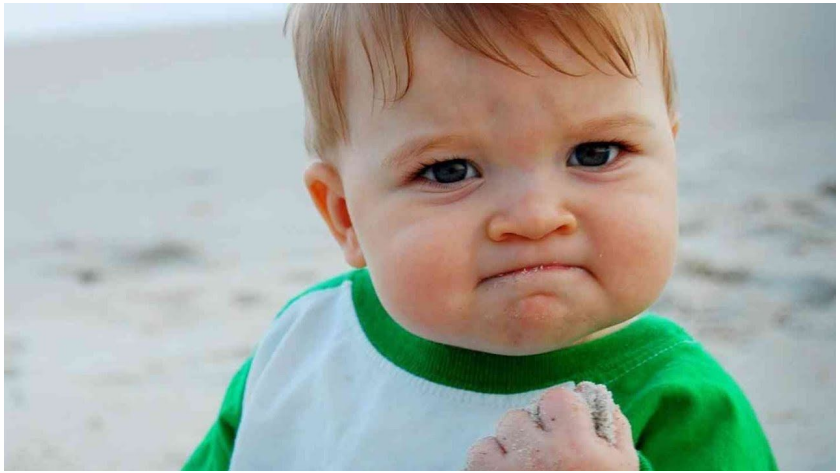
Agenda

- Overview of security challenges in cloud-native environments
- Understanding JWTs
- Deep dive into JWT usage in real-world applications
- Why JWTs are becoming a game-changer for JavaScript developers
- Demo
- Q&A - Bring it on!

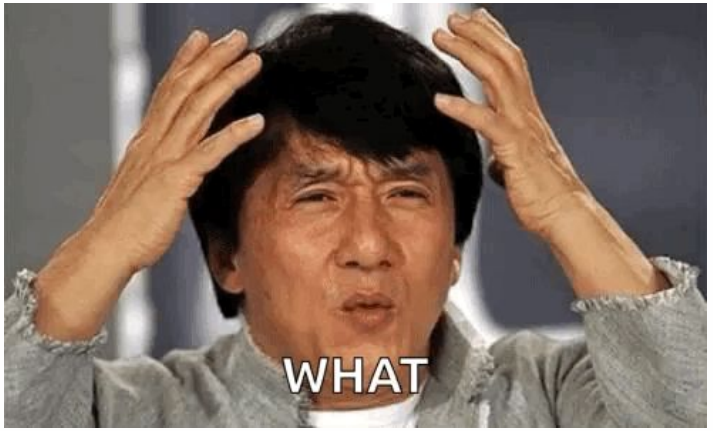
Security Challenges



Versatile Solution - Power of JWT




HOW?


WHAT

Decoding JWT

[header].[payload].[signature]

Algorithm HS256

Encoded PASTE A TOKEN HERE


```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJkZXZtaW8tMTk4IiwibmFtZSI6IlNha3NoaSBOYXNoYSIsImVtYWlsIjoic2Frc2hpQGdtYWlsLmNvbSIsImV4cCI6MTczMDYxODQ2NSwiaWF0IjoxNzI4MDI2NDY1LCJpc3MiOiJodHRwczovL2phdmFzY3JpcHQtY29uZmVyZW5jZS5jb20iLCJhdWQiOiJodHRwczovL2phdmFzY3JpcHQtY29uZmVyZW5jZS5jb20ifQ.EjJ5TUaddcyoYohJI-boSR7aA5REBhn6Ab2n-c2IP5I
```


Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE


```
{
  "alg": "HS256",
  "typ": "JWT"
}
```


PAYLOAD: DATA

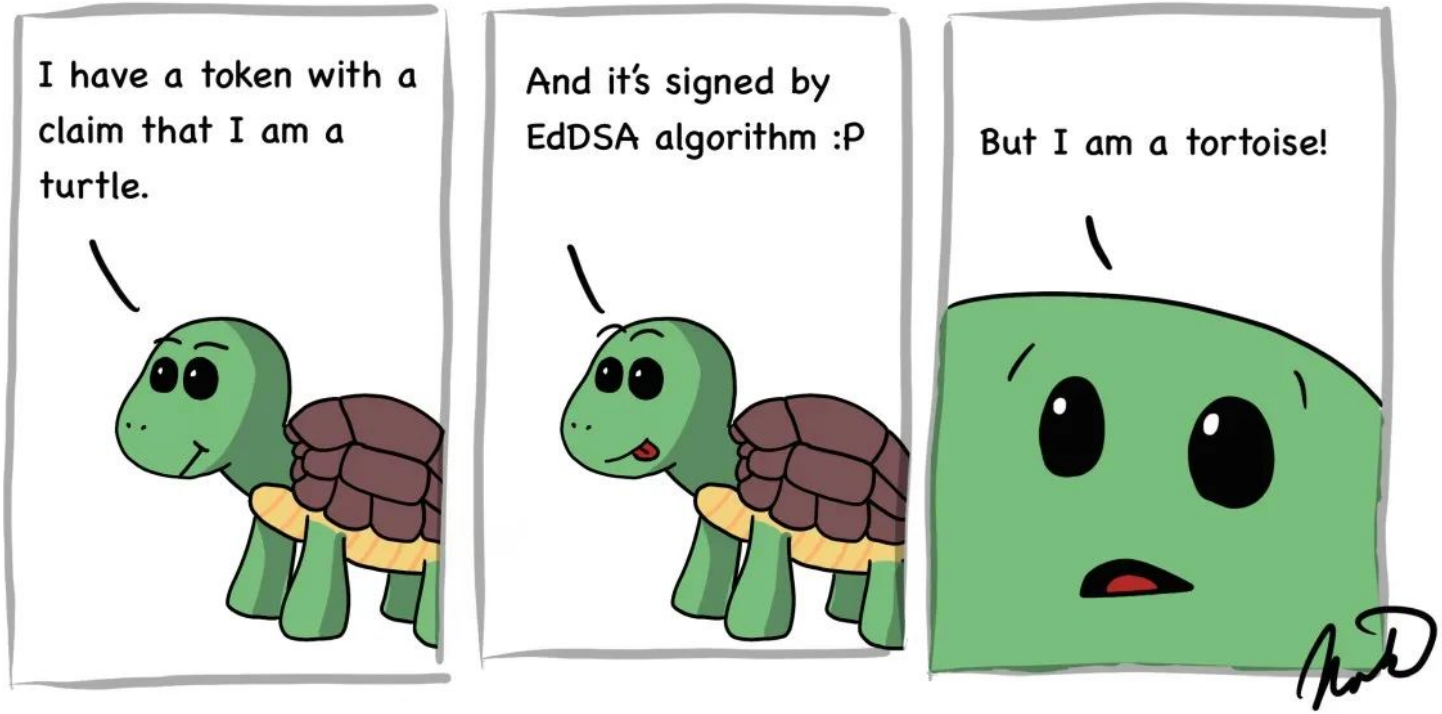

```
{
  "sub": "devmio-198",
  "name": "Sakshi Nasha",
  "email": "sakshi@gmail.com",
  "exp": 1730618465,
  "iat": 1728026465,
  "iss": "https://javascript-conference.com",
  "aud": "https://javascript-conference.com"
}
```


VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  your-256-bit-secret
) ☑ secret base64 encoded
```

Signature Verified

SHARE JWT

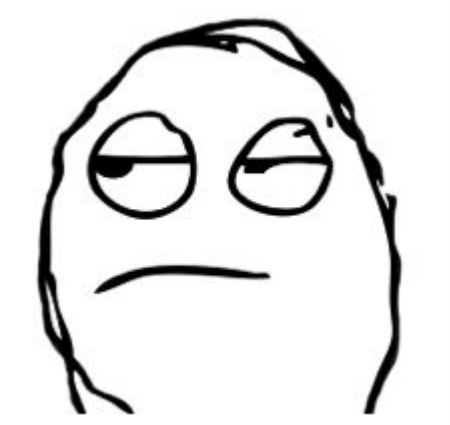

I have a token with a claim that I am a turtle.
And it's signed by EdDSA algorithm :P
But I am a tortoise!

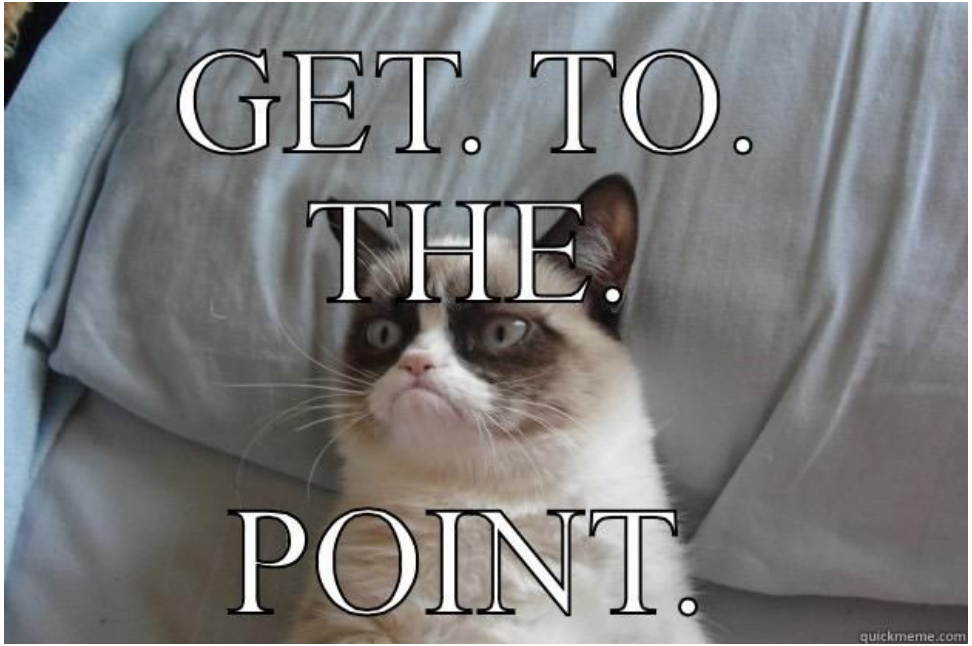
Example JWT Structure:

Here's a simplified example of how your JWT might look:

json Copy code

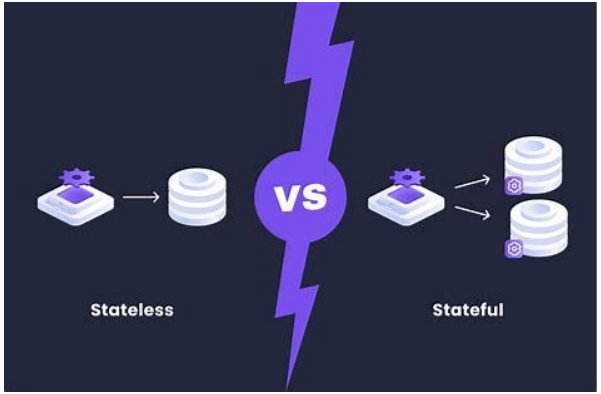
```json
{
  "sub": "1234567890", // Subject (user ID)
  "name": "Sakshi",
  "nickname": "Guddu",
  "iat": 1516239022 // Issued at timestamp
}
```



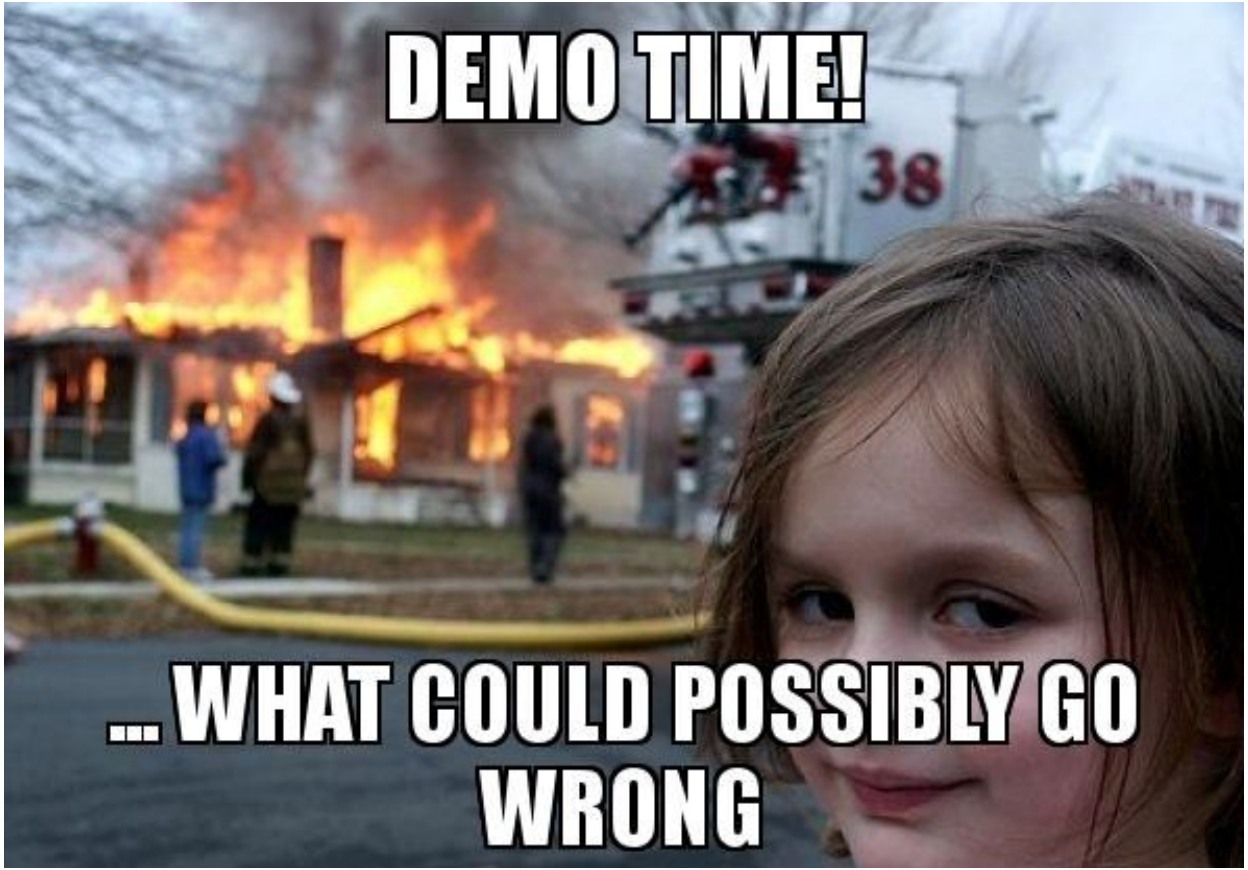

GET. TO.
THE.
POINT.
quickmeme.com

JWT USAGE in real world

https://example.com/reset-password?token=<JWT>



Demo ?


DEMO TIME!
38
...WHAT COULD POSSIBLY GO WRONG

Implement JWT in Node.js Authentication Workflow

1

Import the required libraries

`const jwt = require('jsonwebtoken')`

jsonwebtoken library is essential for handling **JSON Web Tokens (JWTs)** in your application.

2

Creating JWTs (Signing)

`jwt.sign({payload info: },secret key,{expiresIn:});`

Signing a JWT involves using a secret key or a public/private key pair to create a cryptographic signature that ensures the integrity and authenticity of the token.

Implement JWT in Node.js Authentication Workflow

3 Verify the JWTs

jwt.verify(token, process.env.SECRET_KEY)

- To ensure that JWT is legitimate & is not tampered, you need to **verify** the token by checking signature against secret/public key
- The jsonwebtoken library **verifies** the authenticity of the JWT and extract its payload.

4 Decoding the JWTs

const decodedToken = jwt.decode(token);

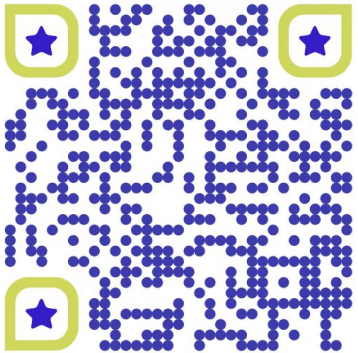
The jsonwebtoken library allows you to **decode** the JWT and inspect the payload and without verifying its signature.

Using Postman

iJS collection

The **iat (issued at) and exp (expiration)** claims are **standard claims** in a JWT and provide metadata about the token:

- **iat**: issued at time. This is the time when the JWT was created.
  - iat: 1730792189 is the timestamp (in seconds since the Unix Epoch) when the JWT was issued. In human-readable terms, this corresponds to **Sun, 06 Dec 2024 08:43:09 GMT**.
- **exp**: expiration time. This is the time when the JWT will no longer be valid.
  - exp: 1730795789 corresponds to a specific future time (in seconds). In this case, it is **Sun, 06 Dec 2024 09:43:09 GMT**, which is 1 hour after the iat (since we set the expiresIn option to '1h' when signing the JWT).



JWT’s - Game Changer

1. Statelessness
2. Scalability
3. Cross Domain Authentication
4. Expiration Control and enhance Security
5. Decentralized Authentication

USE WISELY

Q&A, Bring it on !

"Let’s connect on LinkedIn! Whether it’s a hackathon or a trek, I’m always up for an adventure!"


I'LL JUST
LEAVE NOW