JSON Web Token

- JSON Web Tokens (JWT) are an open standard (RFC 7519) that defines a compact and self-contained format for securely transferring information between parties as a JSON object.
- They are primarily used for authentication and authorization in web applications and API services.
- They consist of three parts: header, payload, and signature.

Internet Engineering Task Force (IETF)
Request for Comments: 7519
Category: Standards Track
ISSN: 2070-1721

M. Jones
Microsoft
J. Bradley
Ping Identity
N. Sakimura
NRI
May 2015

JSON Web Token (JWT)

Abstract

JSON Web Token (JWT) is a compact, URL-safe means of representing claims to be transferred between two parties. The claims in a JWT are encoded as a JSON object that is used as the payload of a JSON Web Signature (JWS) structure or as the plaintext of a JSON Web Encryption (JWE) structure, enabling the claims to be digitally signed or integrity protected with a Message Authentication Code (MAC) and/or encrypted.

Status of This Memo

This is an Internet Standards Track document.

RFC 7519

JWT structure

- Header: Contains metadata about the token type and the encryption algorithm used.
- Payload: Contains the claims, which are statements about the user's identity and any other relevant data.
- Signature: It's a digital signature generated by combining the header, payload, and a secret key, which is used to verify the integrity of the token.

```
eyJhbGciOiJIUzI1NiI
sInR5cCI6IkpXVCJ9.e
yJzdWIiOiIxMjM0NTY3
ODkwIiwibmFtZSI6Ikp
vaG4gRG9lIiwiaWF0Ij
oxNTE2MjM5MDIyfQ.Sf
lKxwRJSMeKKF2QT4fwp
MeJf36POk6yJV_adQss
w5c
```

Singularity Hackers 0xWORD My Public Inbox

Creation and verification

- Generation of a JWT on the server:
  - The server generates a JWT by combining the header and the payload, and then signs the token using a secret key.
- Sending the JWT to the client:
  - Once generated, the server sends the JWT to the client, usually as part of an HTTP response.
- Verification and decoding of the JWT on the server:
  - When the client makes a subsequent request, it includes the JWT in the authorization header.
  - The server verifies the integrity of the token by verifying the signature using the same secret key.
  - If the signature is valid, the server decodes the payload and performs corresponding actions, such as authenticating the user or authorizing access to resources.

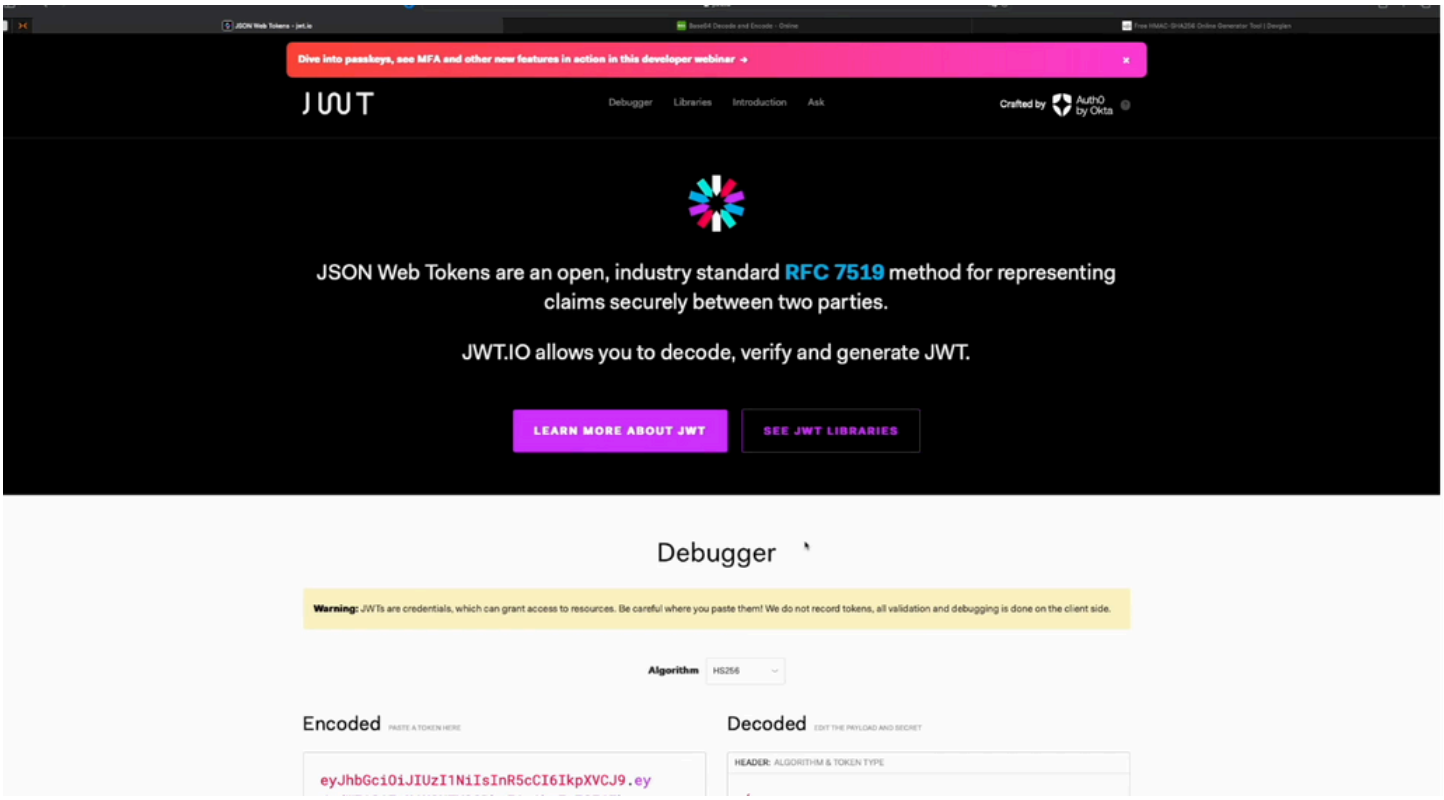
Use cases

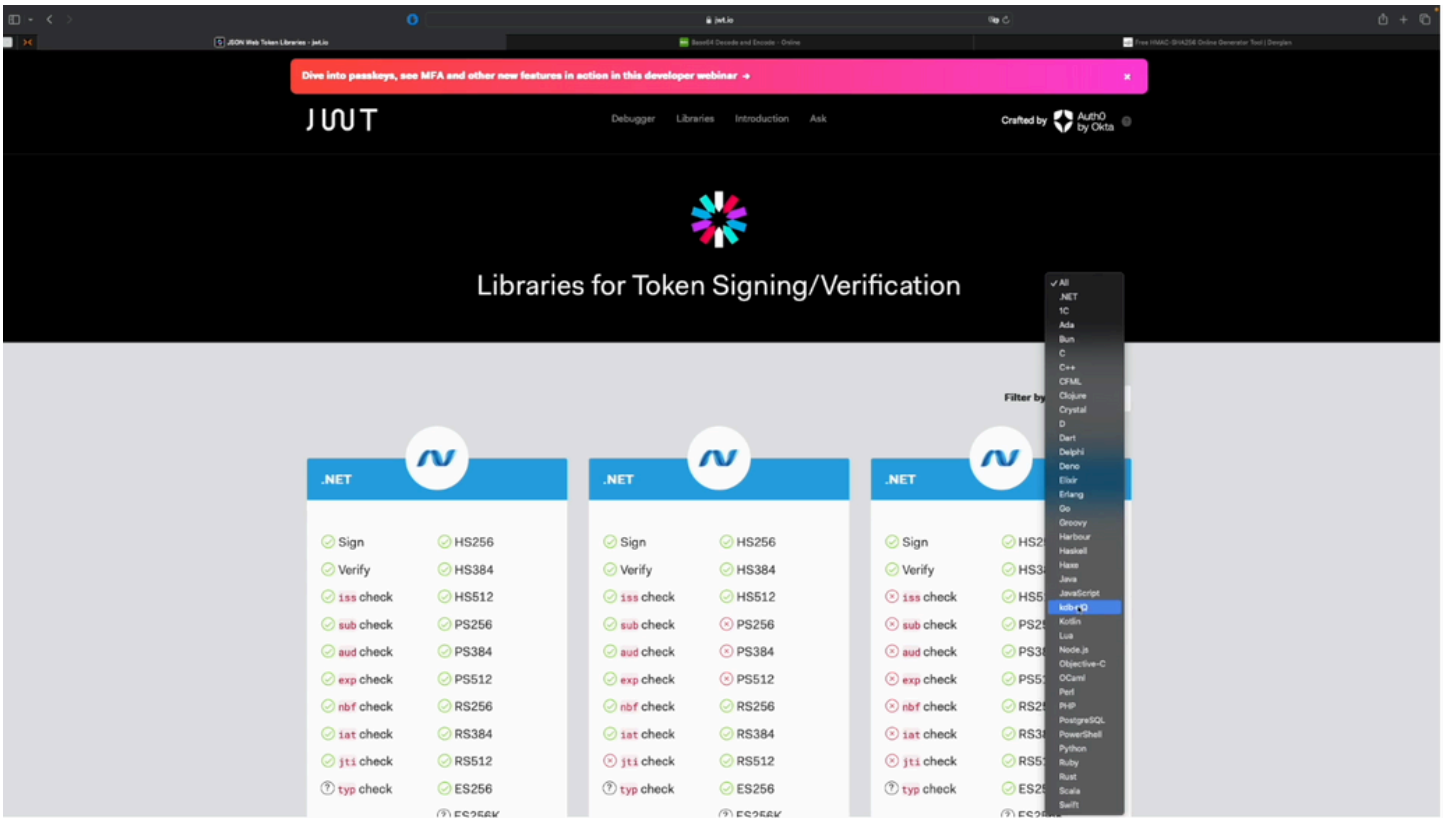
- User authentication in web applications and APIs:
  - JWTs are used to securely authenticate users, avoiding the need to store sessions on the server.
- Authorization and access control to resources:
  - JWTs contain authorization information that can be used to control access to protected resources within an application or API.
- Single Sign-On (SSO) and identity federation:
  - JWTs facilitate the implementation of SSO, allowing a user to log in once and access multiple services without having to re-authenticate.
- Transfer of information between services:
  - JWTs are useful for securely transferring information between different services, as the information is signed and encrypted within the token.

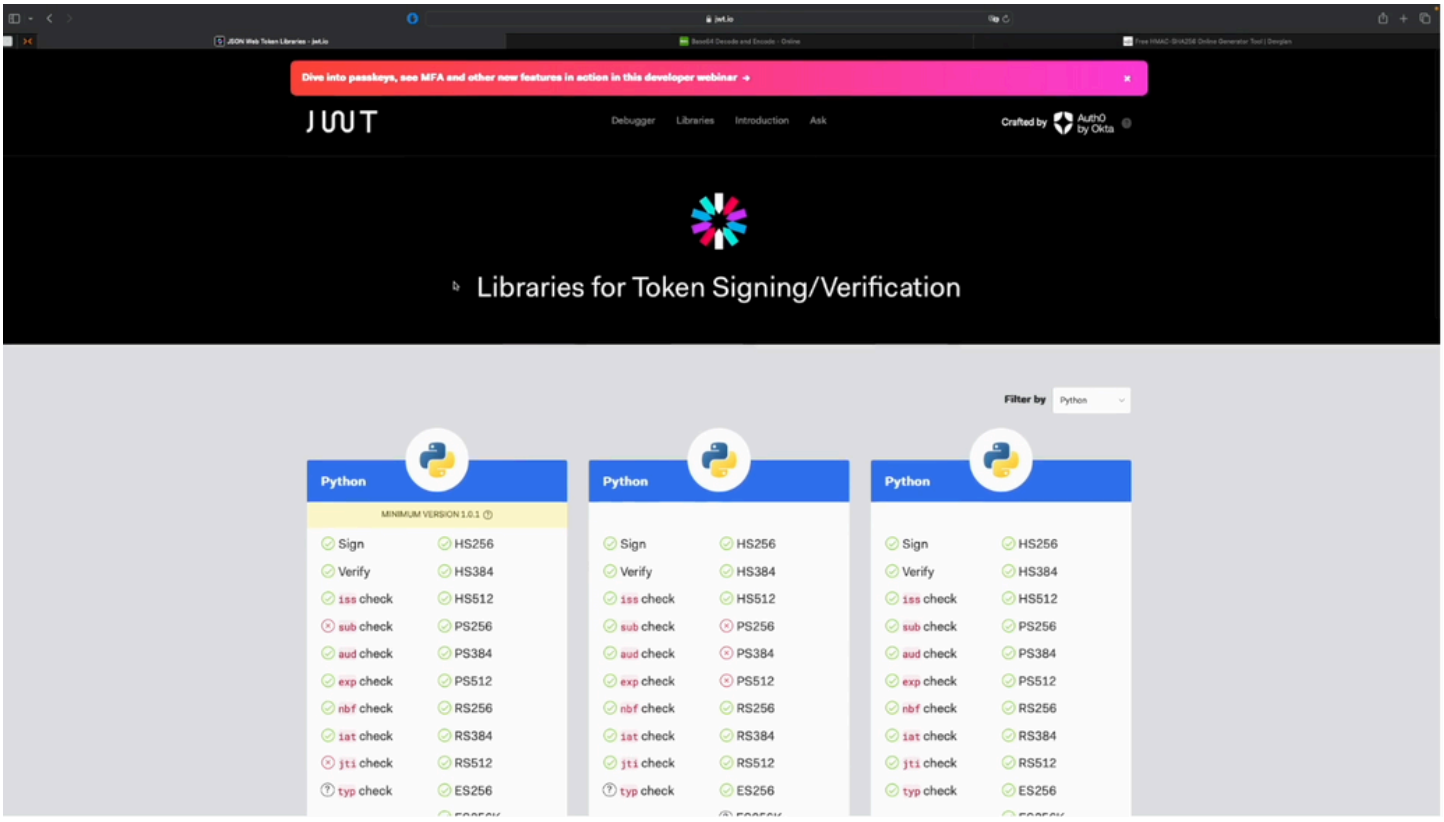
Singularity Hackers 0xWORD My Public Inbox

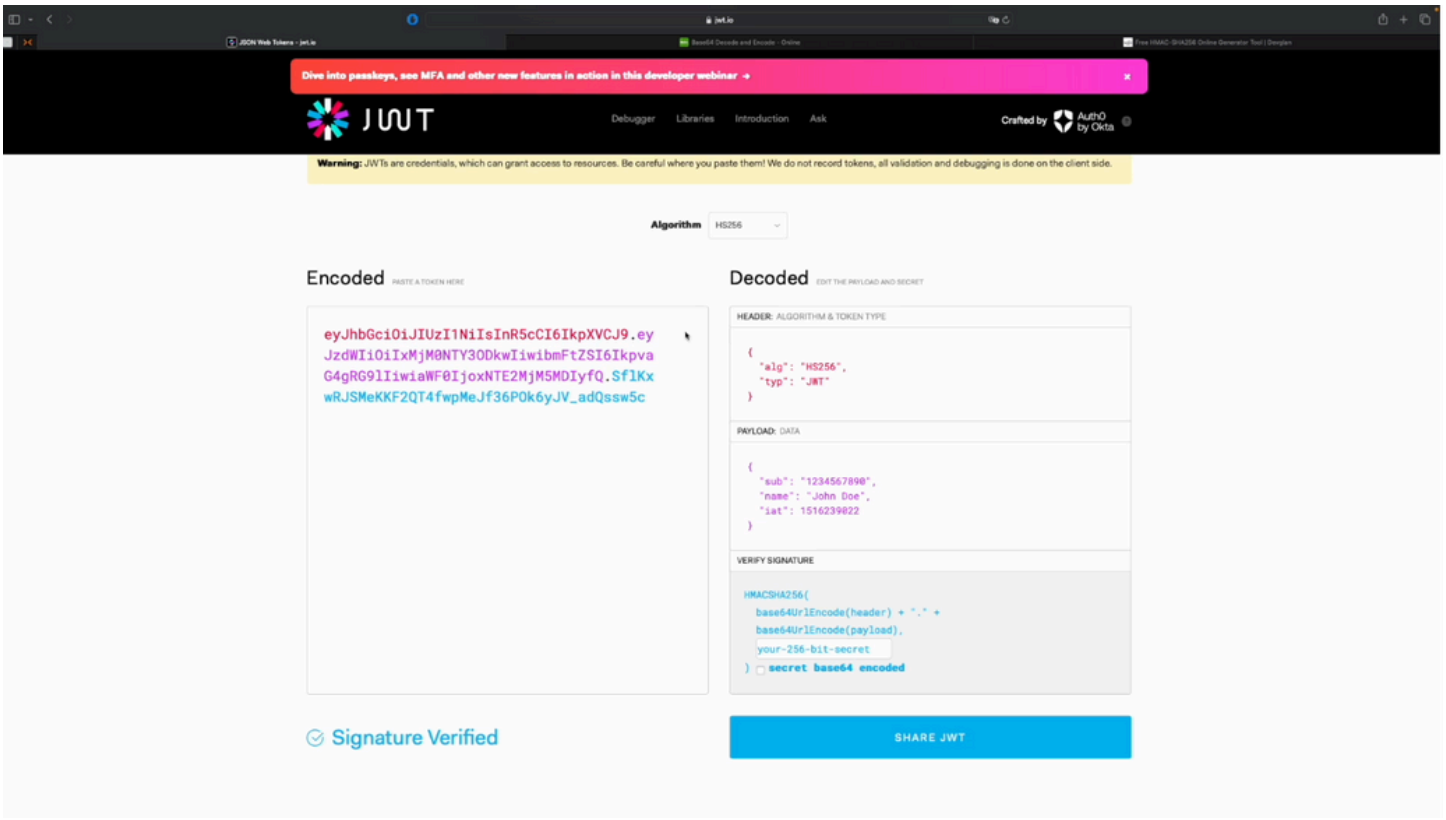
Best practices and security considerations

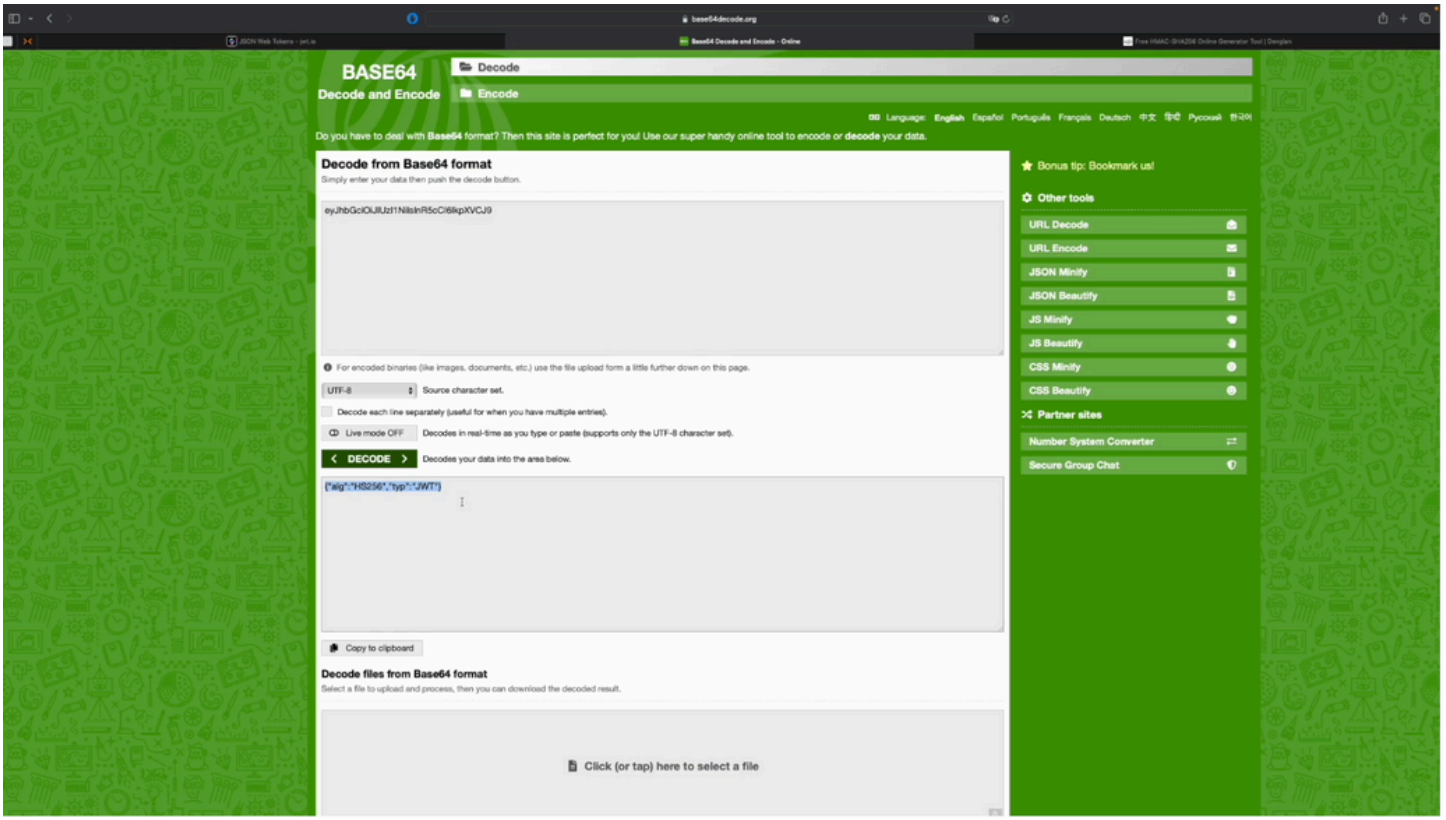
- Recommended encryption algorithms.
- Secure handling of secrets and signing keys.
- Expiration and renewal of JWTs.
- Mitigation of common attacks.

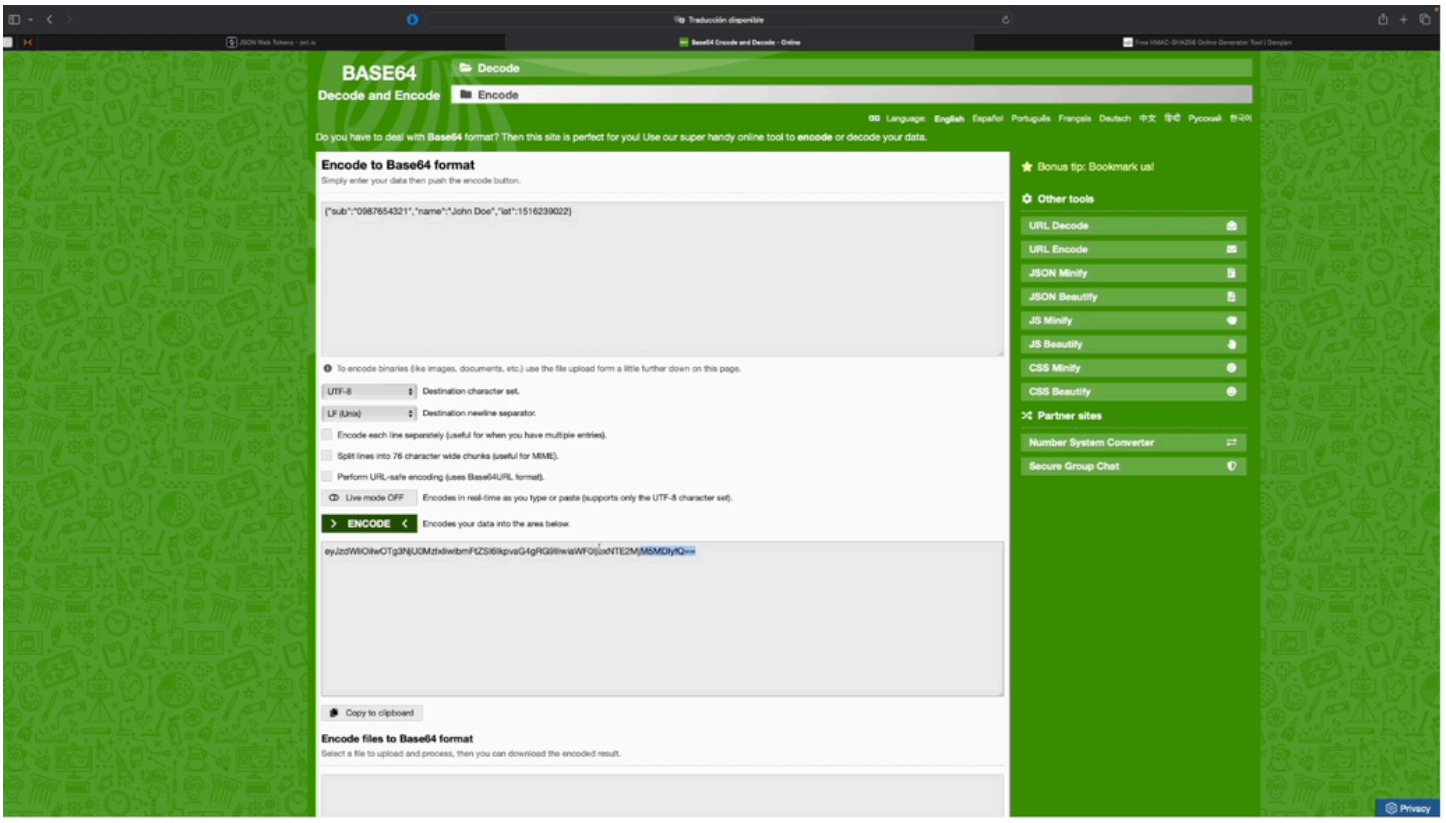


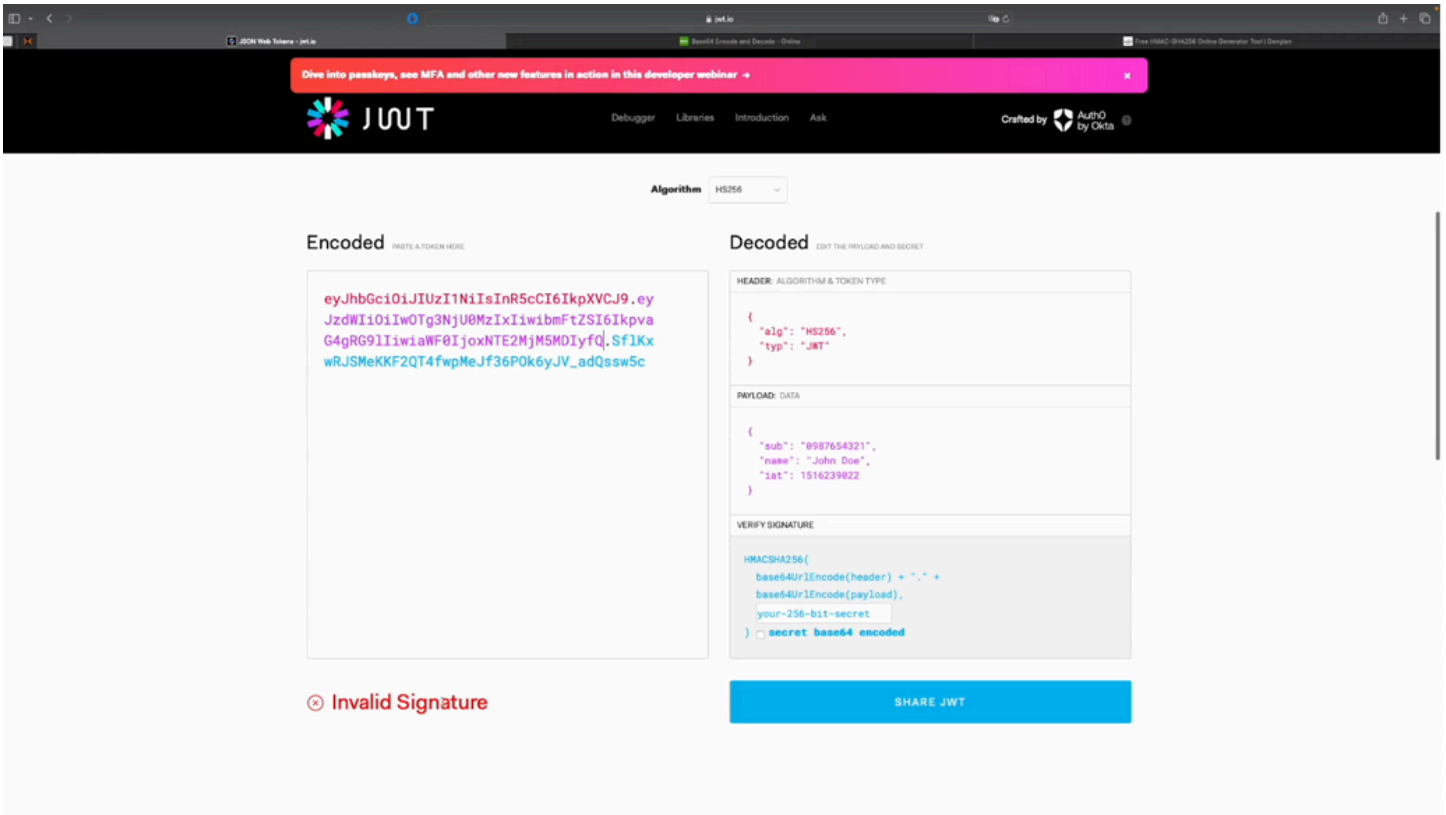

Dive into passkeys, see MFA and other new features in action in this developer webinar →
JWT
Debugger
Libraries
Introduction
Ask
Crafted by Auth0 by Okta
Libraries for Token Signing/Verification
Filter by
All
.NET
1C
Ada
Bun
C
C++
CFML
Clojure
Crystal
D
Dart
Delphi
Deno
Elixir
Erlang
Go
Groovy
Harbour
Haskell
Haxe
Java
JavaScript
kdb+
Kotlin
Lua
Node.js
Objective-C
OCaml
Perl
PHP
PostgreSQL
PowerShell
Python
Ruby
Rust
Scala
Swift
.NET
Sign
Verify
iss check
sub check
aud check
exp check
nbf check
iat check
jti check
typ check
HS256
HS384
HS512
PS256
PS384
PS512
RS256
RS384
RS512
ES256
.NET
Sign
Verify
iss check
sub check
aud check
exp check
nbf check
iat check
jti check
typ check
HS256
HS384
HS512
PS256
PS384
PS512
RS256
RS384
RS512
ES256
.NET
Sign
Verify
iss check
sub check
aud check
exp check
nbf check
iat check
jti check
typ check

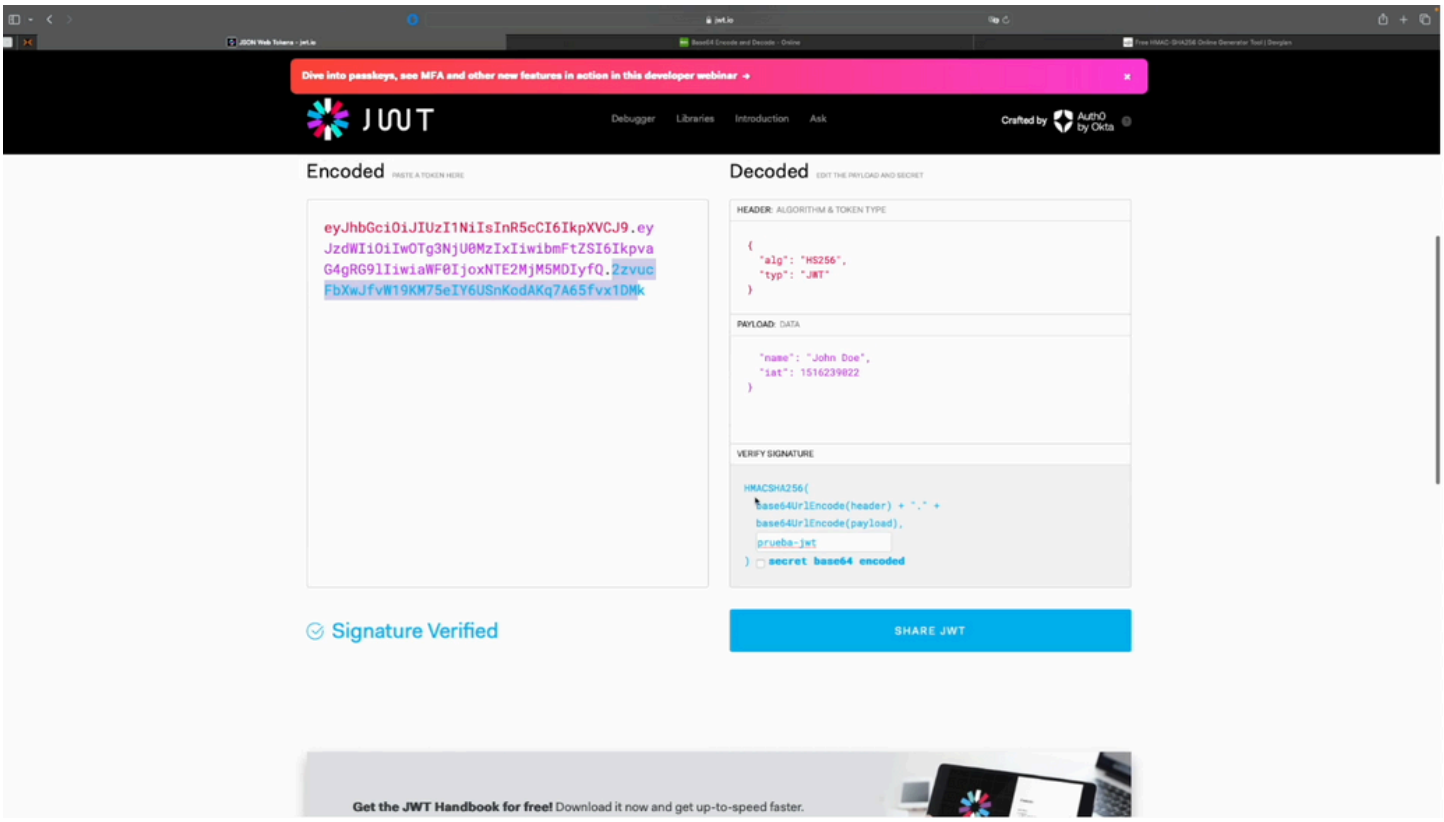

Dive into passkeys, see MFA and other new features in action in this developer webinar →
JWT
Debugger
Libraries
Introduction
Ask
Crafted by Auth0 by Okta
Libraries for Token Signing/Verification
Filter by
Python
Python
MINIMUM VERSION 1.0.1
Sign
Verify
iss check
sub check
aud check
exp check
nbf check
iat check
jti check
typ check
HS256
HS384
HS512
PS256
PS384
PS512
RS256
RS384
RS512
ES256
Python
Sign
Verify
iss check
sub check
aud check
exp check
nbf check
iat check
jti check
typ check
HS256
HS384
HS512
PS256
PS384
PS512
RS256
RS384
RS512
ES256
Python
Sign
Verify
iss check
sub check
aud check
exp check
nbf check
iat check
jti check
typ check
HS256
HS384
HS512
PS256
PS384
PS512
RS256
RS384
RS512
ES256

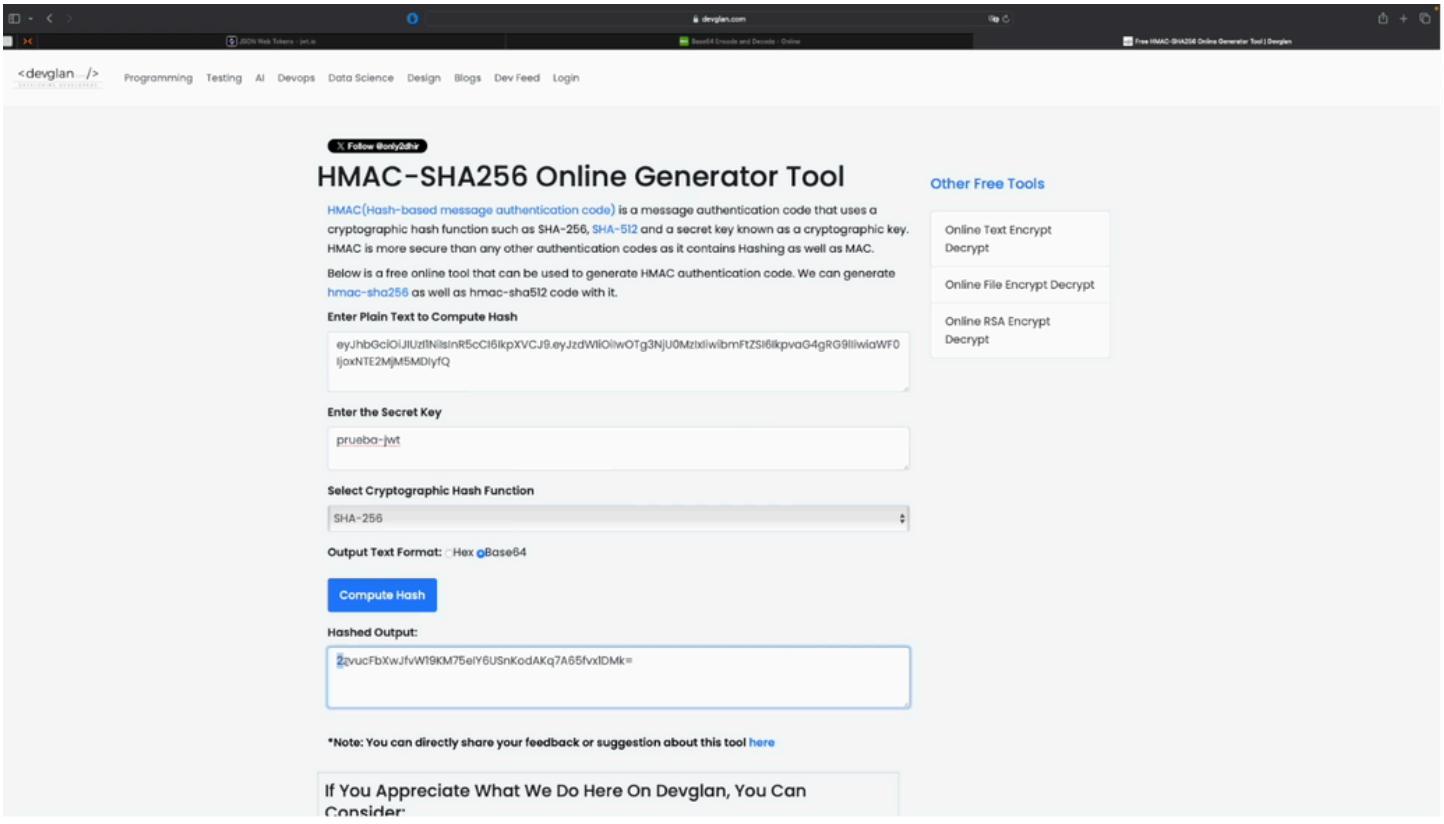

Dive into passkeys, see MFA and other new features in action in this developer webinar →
JWT
Debugger
Libraries
Introduction
Ask
Crafted by Auth0 by Okta
Warning: JWTs are credentials, which can grant access to resources. Be careful where you paste them! We do not record tokens, all validation and debugging is done on the client side.
Algorithm HS256
Encoded PASTE A TOKEN HERE
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
Decoded EDIT THE PAYLOAD AND SECRET
HEADER: ALGORITHM & TOKEN TYPE
{ "alg": "HS256", "typ": "JWT" }
PAYLOAD: DATA
{ "sub": "1234567890", "name": "John Doe", "iat": 1516239022 }
VERIFY SIGNATURE
HMACSHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload), your-256-bit-secret) secret base64 encoded
Signature Verified
SHARE JWT


BASE64
Decode and Encode
Decode
Encode
Language: English Español Português Français Deutsch 中文 हिंदी Русский
Do you have to deal with Base64 format? Then this site is perfect for you! Use our super handy online tool to encode or decode your data.
Decode from Base64 format
Simply enter your data then push the decode button.
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.
UTF-8 Source character set.
Decode each line separately (useful for when you have multiple entries).
Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 character set).
DECODE Decodes your data into the area below.
{"alg":"HS256","typ":"JWT"}
Copy to clipboard
Decode files from Base64 format
Select a file to upload and process, then you can download the decoded result.
Click (or tap) here to select a file
Bonus tip: Bookmark us!
Other tools
URL Decode
URL Encode
JSON Minify
JSON Beautify
JS Minify
JS Beautify
CSS Minify
CSS Beautify
Partner sites
Number System Converter
Secure Group Chat


BASE64
Decode and Encode
Decode
Encode
Do you have to deal with Base64 format? Then this site is perfect for you! Use our super handy online tool to encode or decode your data.
Encode to Base64 format
Simply enter your data then push the encode button.
{"sub":"0987654321","name":"John Doe","iat":1516239022}
To encode binaries (like images, documents, etc.) use the file upload form a little further down on this page.
UTF-8
Destination character set.
LF (Unix)
Destination newline separator.
Encode each line separately (useful for when you have multiple entries).
Split lines into 76 character wide chunks (useful for MIME).
Perform URL-safe encoding (uses Base64URL format).
Live mode OFF
Encodes in real-time as you type or paste (supports only the UTF-8 character set).
ENCODE
Encodes your data into the area below.
eyJzdWIiOiIwOTg3NjU0MzIxIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ==
Copy to clipboard
Encode files to Base64 format
Select a file to upload and process, then you can download the encoded result.
Bonus tip: Bookmark us!
Other tools
URL Decode
URL Encode
JSON Minify
JSON Beautify
JS Minify
JS Beautify
CSS Minify
CSS Beautify
Partner sites
Number System Converter
Secure Group Chat
Privacy


Dive into passkeys, see MFA and other new features in action in this developer webinar →
JWT
Debugger
Libraries
Introduction
Ask
Crafted by Auth0 by Okta
Algorithm
HS256
Encoded
PASTE A TOKEN HERE
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIwOTg3NjU0MzIxIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
Decoded
EDIT THE PAYLOAD AND SECRET
HEADER: ALGORITHM & TOKEN TYPE
{
"alg": "HS256",
"typ": "JWT"
}
PAYLOAD: DATA
{
"sub": "0987654321",
"name": "John Doe",
"iat": 1516239022
}
VERIFY SIGNATURE
HMACSHA256(
base64UrlEncode(header) + "." +
base64UrlEncode(payload),
your-256-bit-secret
) secret base64 encoded
Invalid Signature
SHARE JWT


Dive into passkeys, see MFA and other new features in action in this developer webinar →
JWT
Debugger
Libraries
Introduction
Ask
Crafted by Auth0 by Okta
Encoded
PASTE A TOKEN HERE
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIwOTg3NjU0MzIxIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.2zvucFbXwJfvW19KM75eIY6USnKodAKq7A65fvx1DMk
Decoded
EDIT THE PAYLOAD AND SECRET
HEADER: ALGORITHM & TOKEN TYPE
{
"alg": "HS256",
"typ": "JWT"
}
PAYLOAD: DATA
{
"name": "John Doe",
"iat": 1516239022
}
VERIFY SIGNATURE
HMACSHA256(
base64UrlEncode(header) + "." +
base64UrlEncode(payload),
prueba-jwt
) secret base64 encoded
Signature Verified
SHARE JWT
Get the JWT Handbook for free! Download it now and get up-to-speed faster.


<devglan /> Programming Testing AI Devops Data Science Design Blogs Dev Feed Login
Follow @ankitsharma_007
HMAC-SHA256 Online Generator Tool
HMAC(Hash-based message authentication code) is a message authentication code that uses a cryptographic hash function such as SHA-256, SHA-512 and a secret key known as a cryptographic key. HMAC is more secure than any other authentication codes as it contains Hashing as well as MAC.
Below is a free online tool that can be used to generate HMAC authentication code. We can generate hmac-sha256 as well as hmac-sha512 code with it.
Enter Plain Text to Compute Hash
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIwOTg3NjU0MzIxIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ
Enter the Secret Key
prueba-jwt
Select Cryptographic Hash Function
SHA-256
Output Text Format: Hex Base64
Compute Hash
Hashed Output:
2zvucFbXwJfvW19KM75eIY6USnKodAKq7A65fvx1DMk=
*Note: You can directly share your feedback or suggestion about this tool here
Other Free Tools
Online Text Encrypt Decrypt
Online File Encrypt Decrypt
Online RSA Encrypt Decrypt
If You Appreciate What We Do Here On Devglan, You Can Consider:

Conclusion

- JWTs are JSON objects used for securely transferring information between parties in a distributed system.
- They consist of three parts: header, payload, and signature, ensuring the integrity and authenticity of the token.
- Robust encryption algorithms should be used along with secure handling of secret keys.
- There are multiple use cases such as authentication, authorization, Single Sign-On (SSO), and secure information transfer between services.
- Best practices should be applied including setting expiration dates, mitigating attacks, and using secure encryption algorithms.