JSON Web Token (JWT) Validation

https://campus.barracuda.com/doc/95264619/

JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way to securely transmit the information between parties as a JSON object. JSON objects are digitally signed, so the information transmitted can be trusted.

The Barracuda WAF provides the mechanism to validate the JWT token received along with HTTP or HTTPS requests. The WAF reads the JSON Web Token and verifies the signature part of the token. Additionally, it also verifies JWT claims in the token, like "iat" (issued at), "exp" (expiry time) and "nbf" (not before time), etc.

The client sends the JWT token as a bearer token with the authorization header.

The JWT token consists of 3 parts of the data separated with a dot (".")

JSON web token = base64urlEncoding(header) + '.' + base64urlEncoding(payload) + '.' + base64urlEncoding(signature)

1. base64 encoded header data in JSON format.
2. base64 encoded payload data in JSON format.
3. base64 encoded data of the signature of payload created using the algorithm mentioned in the header part.

Example:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9. ey JzdWIiOiIxMjM0NTY3O

DkwIiwibm FtZSI6Ikpva G4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ . SflK

xwRJSMeKKF2QT4f wpMeJf36POk6yJV_adQssw5c

Here is the data after decoding the JWT token.

Header

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

Payload

```
{
        "sub": "1234567890",
        "name": "John Doe",
        "iat": 1516239022
}
```

Signature

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  <secret>
) secret base64 encoded
```

When an HTTP or HTTPS request has an "Authorization" header with a Bearer token, the WAF will be able to validate this token before the request is forwarded to the Resource server (for example, API server).

"Authorization: Bearer <token>"

The WAF provides two mechanisms to validate the JWT token:

- External Introspection
- Internal Introspection

External Introspection: The Barracuda WAF forwards the JWT token from a client request to the configured external introspection endpoint for token validation. Both the response received from the endpoint and the token are saved in the system's memory cache. A JWT token is considered valid if the response from the external introspection endpoint has the key "active" and its value is set to the Boolean value of true. If the value is set to the Boolean value of false, the token is considered invalid. Subsequent requests with the same token are validated against the cached information.

If the token is valid, the WAF forwards the request to the configured server, i.e., the API server (acting as a Resource server).

Internal Introspection: The Barracuda WAF locally validates the JWT token using the following 3 methods.

1. Public Key Certificates.
2. Shared Keys
3. JWKS URI.

The Barracuda WAF requires public keys to generate the JWT signature part from the header and payload part of the JWT token. Once it generates the signature using the header and payload data, it compares the generated signature with the signature of the JWT token received.

Users can configure all the public keys and associated key ID on the Barracuda WAF, or they can configure the JWKS URI endpoint to download the keys and key ids.

If the JWT token is generated by symmetric-key encryption, the admin can configure the shared key on the Barracuda WAF to validate the JWT signature using the header and payload data.

In both methods, the Barracuda WAF checks for valid claims in the payload before checking the signature validation.

Add a Validation Endpoint

1. Go to the **ACCESS CONTROL > Web Token Validation** page.
2. In the **Validation Endpoint** section, click **Add Validation Endpoint**.
3. On the **Add Validation Endpoint** page, specify values for the following:
    1. **Name**: The name of the validator. The allowed characters are alphanumerical and "_" (underscore) and "-" hyphen. A maximum of 64 characters.
    2. **Type**: Select the type.
        1. **External** - By default, External means that the WAF forwards the JWT to the external Authorization server.
        2. **Internal** - WAF verifies the JWT locally. Here, you can configure all these 3 methods in the order defined for performing internal validation:
            1. **Public Key Configuration** - The WAF requires these public keys to create the JWT signature from the header and payload of the JWT token. Once it generates the signature using the header and payload data, it compares the generated signature with the signature of the JWT token received.
                1. **Key ID** - Specify the key ID.
                2. **Public Keys** - Add the associated public key(s) that can be used for local JWT access token verification.
            3. **Shared Keys Configuration** - If the JWT token is generated by symmetric-key encryption, you can configure the symmetric key and associated key ID on the Barracuda Web Application Firewall.
                1. **Key ID** - Specify the key ID.
                2. **Shared Keys** - Add the associated shared key(s) that can be used for local JWT access token validation.
            4. **JWKS URI Configuration** - Users can configure the JWKs URI endpoint to download the keys and key IDs.
            5. **JWKS URI** - Specify the JWKs URI on which the Authorization server publishes the keys used to sign its JWT access tokens.
    3. **Introspection Endpoint**: This allows the user to configure the actual introspection

endpoint URL to validate the token. This is the URL of the Authorization server that provided these JWTs and is to be validated by the WAF. The WAF forwards the token to this URL endpoint. It is recommended to configure HTTPS URL for the security purpose.

4. **Client ID**: The client ID of the application registered with the Authorization server. Every application registered with the Authorization server has a unique client ID and client secret. This information is used to authenticate the token with the Authorization server.
5. **Client Secret**: The client secret of the application registered with the Authorization server.
6. **Auth Mechanism**: The type of the authentication method used to authenticate the JWT token with the Authorization server. The following Auth Mechanisms are supported:
    1. **Client Secret Basic** - The Barracuda Web Application Firewall sends the client ID and client secret to the Authorization server with a basic authorization header.
    2. **Client Secret Post** - The Barracuda Web Application Firewall sends the client ID and client secret to the Authorization server in the POST body.
    3. **Client Secret JWT** - The Barracuda Web Application Firewall sends the client information to the Authorization server in the JWT format signed using client secret.
    4. **Bearer Access Token** – The user configures the bearer access token that contains the user info of this application.
    5. **None** - Used when the client type is public and client authentication is not required.
7. **Validate Server**: Set to **Yes** to validate the authorization server certificate.

4. Click **Add Validation Endpoint**.

Add a JWT Profile

1. Go to the **ACCESS CONTROL > Web Token Validation** page.
2. In the **Validate Web Tokens** section, click **Add JWT Profile** next to the service.
3. On the **Add JWT Profile** page, specify values for the following:
    1. **JWT Profile Name**: Enter a name for the JWT profile.
    2. **Status**: Set to **On** to enforce checks on requests using this JWT profile.
    3. **Host Match**: Enter a host name to be matched against the host in the request. This can be either a specific host match or a wildcard host match with a single * anywhere in the host name. For example, *.example.com.
    4. **URL Match**: This is used to specify the matching criterion for the URL field in the request header. The URL should start with a "/" and can have only a maximum of one " * " that is treated as a wildcard. Example: /* /index.html /public/index.html
    5. **Validator**: Select a validator for which you want to bind this JWT profile.
    6. **Extended Match Sequence**: Specifies an ascending order sequence to prioritize the extended-match rules for conflicting URL and extended-match keys. Lower sequence number implies higher priority.
    7. **Extended Match**: An expression to match various parts of the request. This specifies matching criteria in addition to the URL match. Refer to help for how to write extended match expressions.
4. Click **Add JWT Profile**.