Progress®

Securing Your Progress® Application Server (PAS) for OpenEdge® with Token-Based Authentication



Rob Marshall
Senior Principal Consultant, OpenEdge Professional Services

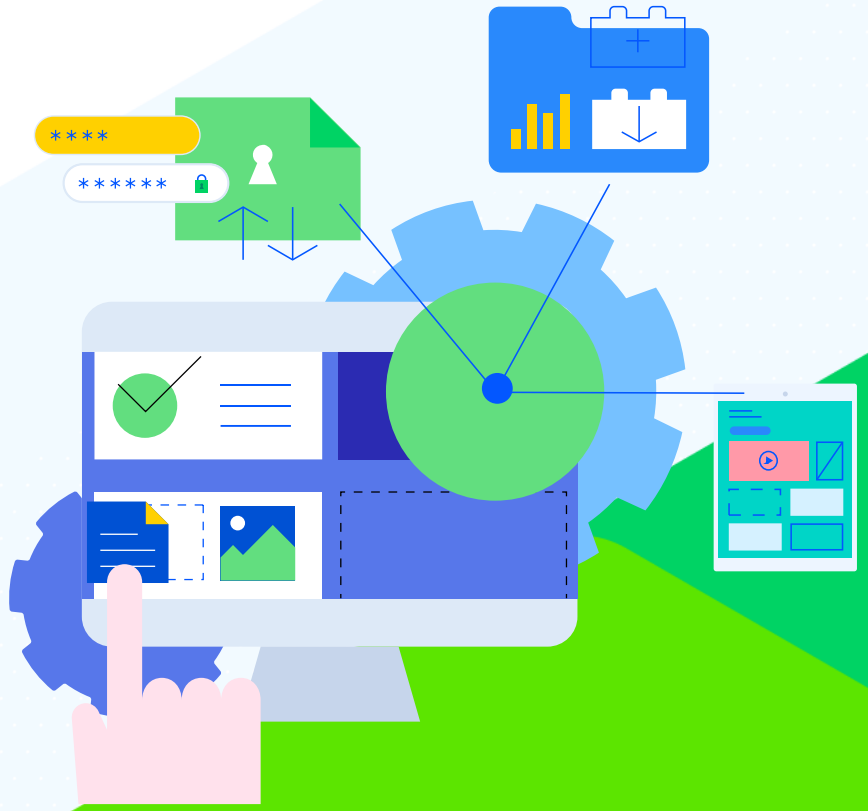
Agenda

- What is the problem?
- What are the options and how do they work?
- What does PAS for OpenEdge provide?

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

What to know about PAS for OpenEdge

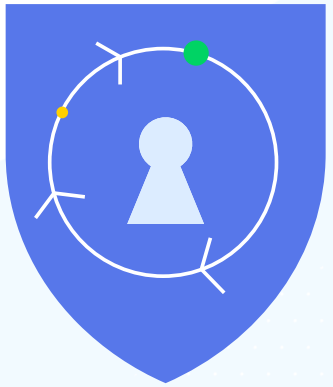
- PAS for OpenEdge is built on Tomcat and this makes it a webserver
- This means all communication is over HTTP/HTTPS



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

What is the problem?

We want to secure our application and data



Solution: Authenticate user

Problem: HTTP is stateless

This means that every request is a blank slate & the server has no memory of who you are

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

What is the problem?



Request 1:
Here are my credentials

✓ Verified

Request 2:
(half a second later)

✗ Who are you?

Solving this amnesia problem is
why authentication exists

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

What are the options?

- Basic Authentication
- Bearer Token
- JSON Web Token (JWT)
- Security Assertion Markup Language (SAML)
- Nonstandard Tokens



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Basic Authentication

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Basic authentication

- Oldest option
- Combines username and password with a colon
- Base64 Encode the resulting string
- Put it in the HTTP authorization header
- Send it with each request

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Basic authentication

admin

secretPassword12345!

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Basic authentication

admin

secretPassword12345!

admin:secretPassword12345!

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Basic authentication

admin

secretPassword12345!

admin:secretPassword12345!

YWRtaW46c2VjcmV0UGFzc3dvcmQxMjM0NSE=

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Basic authentication

admin

secretPassword12345!

admin:secretPassword12345!

YWRtaW46c2VjcmV0UGFzc3dvcmQxMjM0NSE=

Authorization: BASIC YWRtaW46c2VjcmV0UGFzc3dvcmQxMjM0NSE=

This is encoded, not encrypted!
It can be reversed by anyone intercepting the HTTP message.

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Basic authentication

When to use:

- Internal tools
- Server-to-server
- Quick prototypes
- CI/CD pipelines

When NOT to use:

- Public-facing APIs
- User login pages
- Mobile apps

Why?

Because simplicity is more important than sophistication

Why?

Because credentials are sent on each request, so each request is a chance for something to go wrong

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Bearer Token

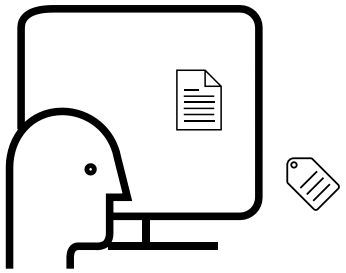
© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

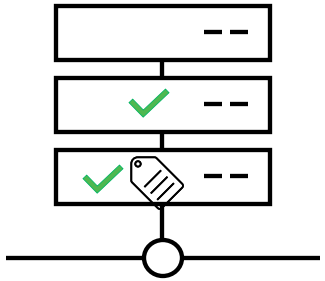
Bearer token

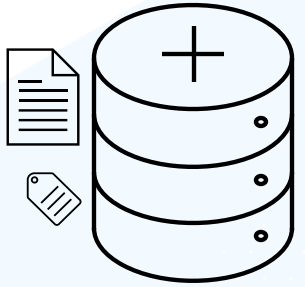
- Only login once with credentials to receive a token
- Token is a string with random characters
- On each subsequent request send the token in the authorization header, no more credentials required
- Token is opaque, so does not contain any information, so an intercepted HTTP message does not provide any information about credentials

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Bearer token







© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Bearer token

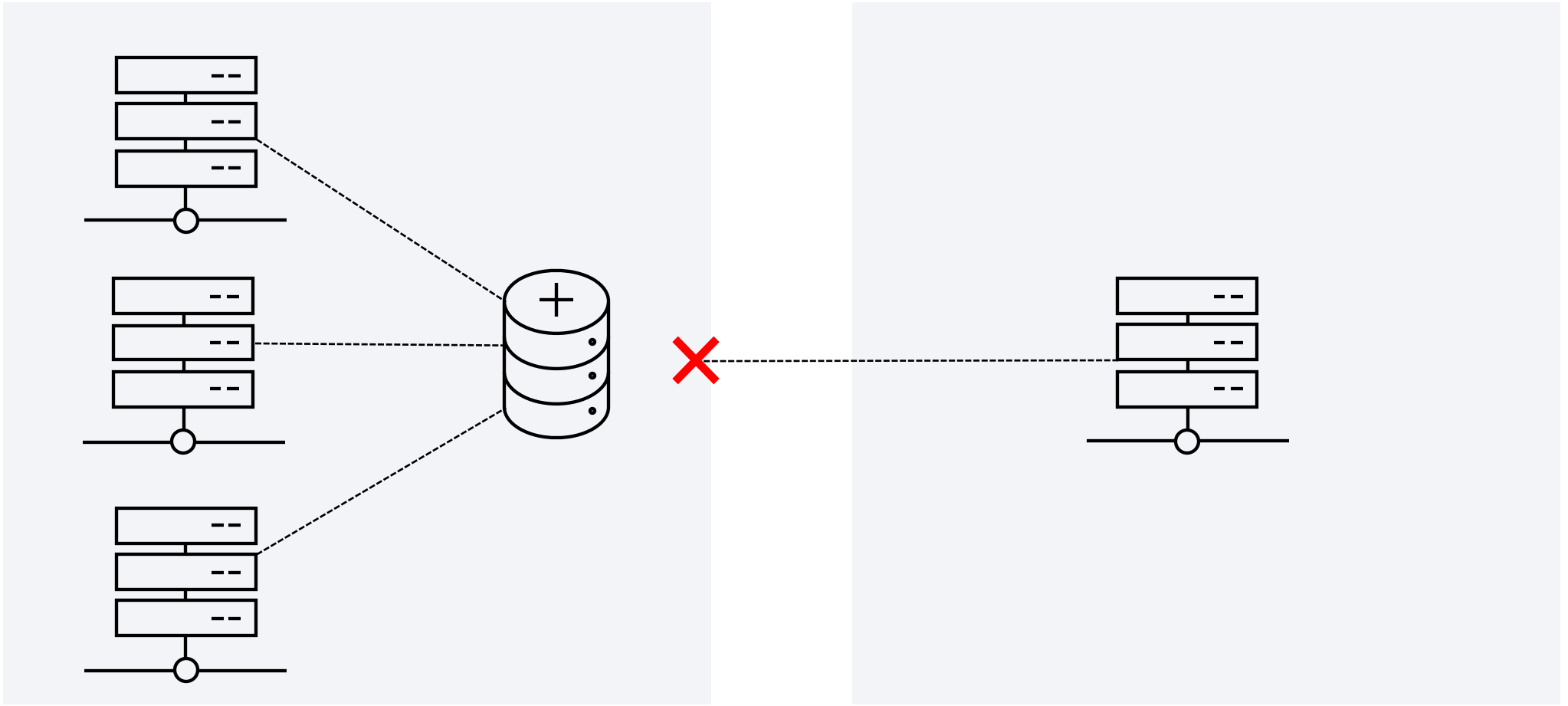
- More secure than basic authentication
- Opaque, so no information in token
- Requires database query on each request, which is a performance penalty

Problem: Scalability

- More servers means more connections to the token storage
- Authentication becomes a separate infrastructure
- Scaling wall

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Bearer token



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Bearer Token

When to use:

- Single app on a single server with sessions

When NOT to use:

- Public-facing APIs
- Multiple servers

Why?

- Easy revocation
- Simple logic
- Proven technology

Why?

- Scalability issues with token storage
- Performance issues

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

What If the Token Didn’t Need the Database?

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JSON Web Token (JWT)

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT

- Contains all needed information
- Consists of three parts, all Base64-encoded
  - **Header**
  - **Payload**
  - **Signature**
- This is actually a JSON Web Signature (JWS), but is generally referred to as JWT

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT

eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJxVGY5NVZtbmlDa2pMYXVvdE05aTNxMFc0c
0lpVDhTSkpDT2IxQ2hyTWVJln0.eyJleHAiOjE3ODI4ODkzMzEsImlhdCI6MTc4Mjg4OTAzMSwianRpIjoi
b25ydHJvOmNmODQ1OWM4LTNhNTktYzQyYy01ZTE2LWM2YjZiMTg3NDA3MyIsImlzcyI6Imh0dHA6
Ly9sb2NhbGhvc3Q6ODA4MC9yZWFsbXMvd29ya3Nob3AiLCJhdWQiOiJhY2NvdW50Iiwic3ViIjoiZDI2
MTYwYWUtYzY3Yi00ZTdiLTk5NDAtZjgxOTU3YjlkMjQ5IiwidHlwIjoiQmVhcmVyIiwiYXpwIjoib0F1dGhD
bGllbnQiLCJzaWQiOiJpQXRFd2hYY0IwNExEV0NvNjlqakVoNUQiLCJhY3IiOiIxIiwiYWxsb3dlZC1vcmln
aW5zIjpbIi8qIl0sInJlYWxtX2FjY2VzcyI6eyJyb2xlcyI6WyJkZWZhdWx0LXJvbGVzLXdvcmtzaG9wIiwib2
ZmbGluZV9hY2Nlc3MiLCJBcHBVc2VyIiwidW1hX2F1dGhvcml6YXRpb24iXX0sInJlc291cmNlX2FjY2Vz
cyI6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50IiwibWFuYWdlLWFjY291bnQtbGlua3
MiLCJ2aWV3LXByb2ZpbGUiXX19LCJzY29wZSI6IkFwcFVzZXIiLCJuYW1lIjoiRGFmZnkgRHVjayIsInB
yZWZlcnJlZF91c2VybmFtZSI6ImRhZmZ5IiwiZ2l2ZW5fbmFtZSI6IkRhZmZ5IiwiZmFtaWx5X25hbWUiOi
JEdWNrIn0.FUzpYV2fMfuMOTY-V0wW42Z_DJyIQZoOKOz4nlGdg0jvcHII6PsOwn589-
YSXuS3GUKvqPkdcqWqYuehayagqidWmVrnjlrRRHfJyI6jd9q1swAB5ztwCNu05_q0LBFtX4CBFn27Pz
AD14tBOYS9OhNeX9qv3KfAlJWQdQGgcaHLbzPwXt_srbWJNskXrQgHNUNXU9c9EU_mpKhd1NuiQ
QlBAlB2xzmaQptMA65sCQfVp9qAsrwKTOp8LPNv7zG2o9Xn_LBO74B2skbiEAQQsAo8toxyWCAyeE
giZ9tMMXKXtzPw7xvBfoyCpoODJQoZeg6NpVddIdOvo2YhE45w6w

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT - Header

eyJhbGciOiJSUzI1NiIsInR5cCI
gOiAiSldUIiwia2lkIiA6ICJxVGY
5NVZtbmlDa2pMYXVvdE05aT
NxMFc0c0lpVDhTSkpDT2IxQ2
hyTWVJIn0

```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid": "qTf95VmniCkjLauotM9i3q0W4sliT8SJJCOb1ChrMeI"
}
```

- Instructions on how to verify the token
- alg – Algorithm used to sign the token
- typ – Token type (usually JWT)
- kid – Key ID optional (depending on algorithm)

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT - Payload

eyJleHAiOjE3ODI4ODkzMzEsImlhdCI6MTc4Mjg4OTA
zMSwianRpIjoib25ydHJvOmNmODQ1OWM4LTNhNTk
tYzQyYy01ZTE2LWM2YjZiMTg3NDA3MyIsImlzcyI6Im
h0dHA6Ly9sb2NhbGhvc3Q6ODA4MC9yZWFsbXMvd
29ya3Nob3AiLCJhdWQiOiJhY2NvdW50Iiwic3ViIjoiZDI
2MTYwYWUtYzY3Yi00ZTdiLTk5NDAtZjgxOTU3YjlkMj
Q5IiwidHlwIjoiQmVhcmVyIiwiYXpwIjoib0F1dGhDbGllb
nQiLCJzaWQiOiJpQXRFd2hYY0IwNExEV0NvNjlqakV
oNUQiLCJhY3IiOiIxIiwiYWxsb3dlZC1vcmlnaW5zIjpbIi
8qIl0sInJlYWxtX2FjY2VzcyI6eyJyb2xlcyI6WyJkZWZhd
Wx0LXJvbGVzLXdvcmtzaG9wIiwib2ZmbGluZV9hY2Nl
c3MiLCJBcHBVc2VyIiwidW1hX2F1dGhvcml6YXRpb2
4iXX0sInJlc291cmNlX2FjY2VzcyI6eyJhY2NvdW50Ijp7
InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50IiwibWFuYW
dlLWFjY291bnQtbGlua3MiLCJ2aWV3LXByb2ZpbGUi
XX19LCJzY29wZSI6IkFwcFVzZXIiLCJuYW1lIjoiRGF
mZnkgRHVjayIsInByZWZlcnJlZF91c2VybmFtZSI6ImR
hZmZ5IiwiZ2l2ZW5fbmFtZSI6IkRhZmZ5IiwiZmFtaWx5
X25hbWUiOiJEdWNrIn0

{
  "exp": 1782889331,
  "iat": 1782889031,
  "jti": "onrtro:cf8459c8-3a59-c42c-5e16-c6b6b1874073",
  "iss": "http://localhost:8080/realms/workshop",
  "aud": "account",
  "sub": "d26160ae-c67b-4e7b-9940-f81957b9d249",
  "typ": "Bearer",
  "azp": "oAuthClient",
  "sid": "iAtEwhXcB04LDWCo69jjEh5D",
  "acr": "1",
  "scope": "AppUser",
  "name": "Daffy Duck",
  "preferred_username": "daffy",
  "given_name": "Daffy",
  "family_name": "Duck"
}

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT - Payload

- Actual data about the user
- Contains claims (name – value pairs)
  - Custom claims, f.i.
    - ✓ name
    - ✓ roles
    - ✓ OR whatever your application needs
  - Registered claims, f.i.
    - ✓ sub – subject
    - ✓ iat – issued at
    - ✓ exp – expiration
- Base64-encoded; NOT encrypted
- Never put in sensitive PII data, like passwords, credit card numbers, API keys or SSNs

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT - Signature

FUzpYV2fMfuMOTY-V0wW42Z_DJyIQZoOKOz4nlGdg0jvcHII6PsOwn589-YSXuS3GUKvqPkdcqWqYuehayagqidWmVrnjIrRRHfJyI6jd9q1swAB5ztwCNu05_q0LBFtX4CBFn27PzAD14tBOYS9OhNeX9qv3KfAlJWQdQGgcaHLbzPwXt_srbWJNskXrQgHNUNXU9c9EU_mpKhd1NuiQQlBAlB2xzmaQptMA65sCQfVp9qAsrwKTOp8LPNv7zG2o9Xn_LBO74B2skbiEAQQsAo8toxyWCAyeEgiZ9tMMXKXtzPw7xvBfoyCpoODJQoZeg6NpVddldOvo2YhE45w6w

- This is what makes the JWT trustworthy
- Is generated by:
  - Combining the encode header and encoded payload, separated by a “.”
  - Running this through the algorithm specified in the header, using a secret key
- If a single character is changed in the header or payload, the signature won’t match
- If the signature doesn’t match the server, it should reject the token

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT

- Does not need a database lookup to verify the token
- Token is verified mathematically using the signature
- Token contains all information
- Works over multiple servers without shared storage
- Tokens are verified independently on each server using the same secret key

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT - Revocation

- The problem is revocation
- With Bearer tokens, revocation is easy
  - Remove token from database -> token not accepted anymore
- JWT is not easy
  - No database to remove from
  - Valid until expiration time has passed

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT - Revocation

- Short expiration time - minutes, not hours
- The shorter the lifespan, the smaller the window of vulnerability
- Creates new problem:
  - Requires the user to log in again if the token has expired
- Solution: Refresh token

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT – Access & Refresh tokens

On login, two tokens are issued:

- Access token
  - ✓ JWT
  - ✓ Short-lived (f.i. 15 minutes)
  - ✓ Self-contained
  - ✓ Used to authenticate to server
- Refresh token
  - ✓ Opaque
  - ✓ Long-lived (f.i. 7 days)
  - ✓ Stored
  - ✓ Revocable
  - ✓ Used to get a new Access token

Best of both worlds. Speed when needed, control when it matters.

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWT

When to use:

- Public APIs
- Multiple servers/applications

Why?

- Stateless
- Fast
- Scalable

When NOT to use:

- Anywhere basic or bearer would suffice

Why?

- Complex

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

HTTPS

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

HTTPS

- Always use HTTPS when using Basic, Bearer of JWT authentication
- All three send basic text
- Base64 Encoding is not security—it is a coding trick to translate anything to a safe, standard set of 64 characters so it can be sent over HTTP
- HTTPS encrypts the complete HTTP message, making sure an intercepted message cannot be read (easily)

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JSON Web Encryption (JWE)

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

JWE

JWT with encrypted payload

Use Case	JWT	JWE
Data Visibility	Signed only, plain-text payload	Encrypted, payload is ciphertext
Primary Use	Authentication that transmits nonsensitive claims	Securely transmits highly sensitive claims (f.i. financial, medical, personal-type PII)
Security Focus	Integrity	Integrity and confidentiality

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

OAuth 2.0

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

OAuth 2.0

- Authorization framework
- Provides a protocol to manage and grant access to resources
- Produced token is often formatted and handled as a JWT
- Can be stateful and allows for revocation
- Application server can check with authorization server (identity provider) if token is still valid, even when not expired (introspection)
- Use when you need to implement third-party logins (i.e., Google, Github, Okta) or handles complex delegated permissions

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

OpenID Connect (OIDC)

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

OpenID Connect (OIDC)

- Authentication protocol
- Builds on top of OAuth 2.0
- Provides basic profile information of the user
- JWT becomes an Identity token, instead of an Access token, which only has enough information to log in
- Can also be used in two steps:
  - Login provides an Access token, which is passed to the application
  - Application calls authorization server with Access token and receives an Identity token
- This allows for cleaner access token and only permitted applications can get the user profile information

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Security Assertion Markup Language (SAML)

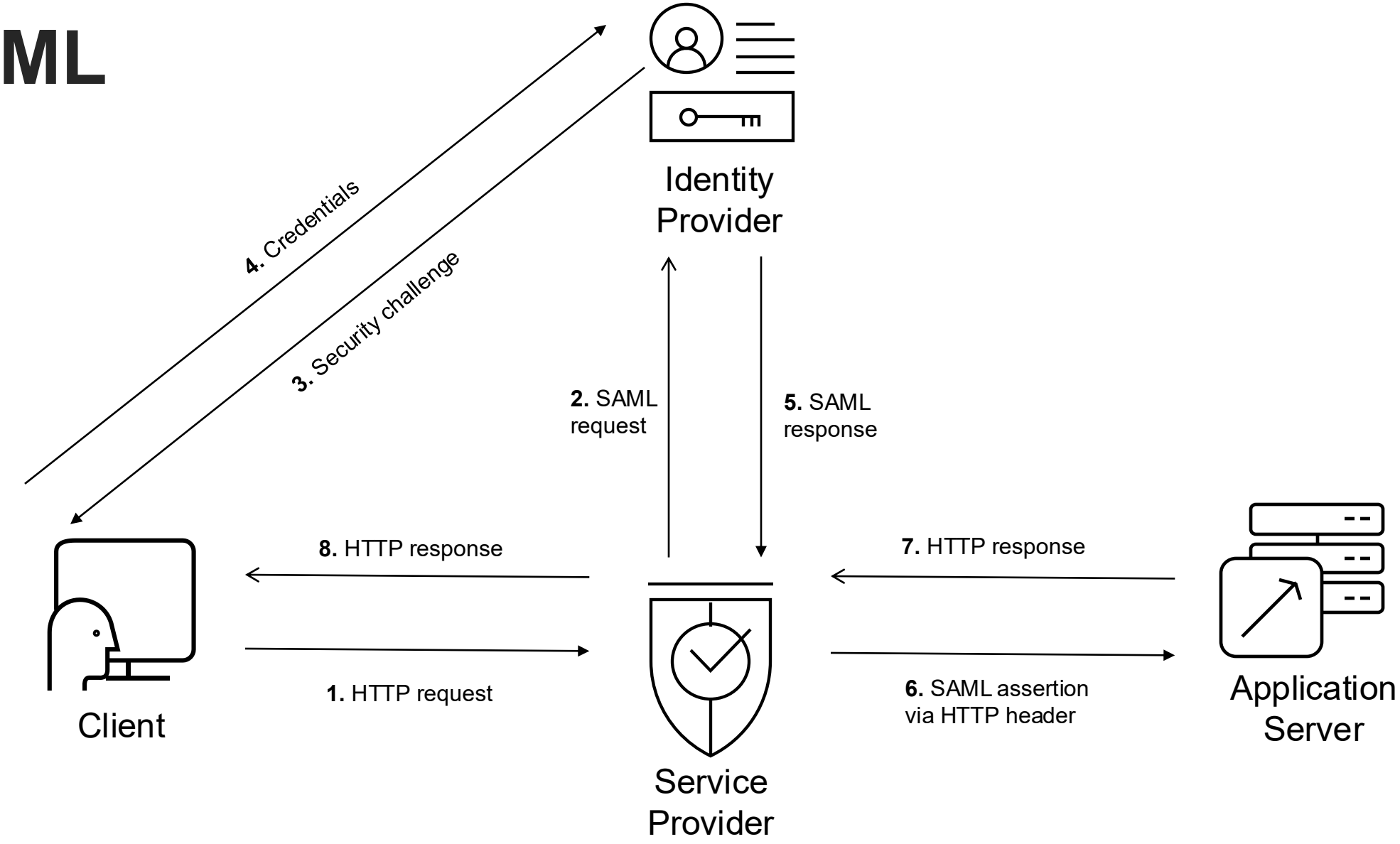
© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

SAML

- XML-based standard, includes complex payload, digital signatures and encryption options
- Enterprise Single Sign-On (SSO) and identity federation
- Handles identity propagation between different organizations
- Often links internal corporate directory (Active Directory) with third-party SaaS application (f.i. Salesforce)
- Highly robust and flexible, but makes it complex to implement and parse
- Needs a service provider, as the token is never passed directly to the client application

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

SAML



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Nonstandard Tokens

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Nonstandard Tokens

- Provider specific implementations
- Often like Bearer or JWT token implementations
- For the OpenEdge platform, this is called HTTP SSO

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

PAS for OpenEdge

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

PAS for OpenEdge

- Security is handled by Spring Security Framework
- Basic authentication
  - Available on all transports (APSV, SOAP, REST, WEB)
  - Validation via LDAP, AD, OERealm or local file
- Bearer token
  - Not directly supported
  - Similar option with form authentication using a JSESSIONID cookie (only on WEB and REST)
  - Multiserver option through clustering

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

PAS for OpenEdge

- JWT/OAuth2/OIDC
  - Only available for WEB and REST
  - JWT or stricter OAuth2 configuration available
  - Introspection
  - Two-step OIDC
- SAML
  - Only available for WEB and REST
  - Until OpenEdge 12.4, a separate service provider was mandatory
  - OpenEdge 12.5+ Spring Security can be configured to act as a service provider

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

PAS for OpenEdge

- HTTP SSO
  - OpenEdge nonstandard token format
  - One PAS for OpenEdge configured as token producer
    - ✓ Uses Basic authentication to receive credentials
    - ✓ Validates credentials using local options
    - ✓ Creates a serialized Client-Principal token for client upon validation, optional with refresh token
  - Other PAS for OpenEdge instances can be configured as token consumers
    - ✓ Validates Client-Principal token received in authorization header based on scope

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

PAS for OpenEdge

- Security configuration is done on the webapp
- Configuration files are in <instance folder>\webapps\<webapp folder>\WEB-INF
  - oeablSecurity.properties
  - oeablSecurity.csv
  - oeablSecurityJWT.csv

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

PAS for OpenEdge Configuration – oeablSecurity.properties

- Spring Security configuration for authentication
- Properties for the following:
  - How credentials are received
    - ✓ client.login.model
  - How credentials are validated
    - ✓ http.all.authmanager
  - How the generated Client-Principal object is filled

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

PAS for OpenEdge Configuration – oeablSecurity.csv

- Spring Security configuration for authorization
- Spring Security uses role-based authentication
- Roles are assigned during authentication
- List of rules parsed from top to bottom
- If a matching rule is found, no further lines are evaluated
- If a rule grants permission to a role, it implicitly denies all other roles

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

PAS for OpenEdge Configuration – oeablSecurity.csv

- Rule: "url-pattern","<method>","<spring-access-expression>“
- url-pattern:
  - Always starts with a “/”, as it is a relative path to the webapp
  - Can contain wildcards
    - ✓ * - all files/endpoints in the current folder or as a part of the name (f.i. *.html)
    - ✓ ** - all files and folders from here
- Single HTTP method (f.i. GET or POST) or a * for all HTTP methods
  - Not possible to specify multiple specific methods in one rule → create two equal rules with different methods

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

PAS for OpenEdge Configuration – oeablSecurity.csv

- spring-access-expression
  - This is a Spring Security Framework function to evaluate the roles of a user and determine if the user has access to combination the url pattern and the method of the request, including these functions:
    - ✓ PermitAll() – Anybody has access
    - ✓ HasAnyRole(“<role-name>”[,”<role-name>”]…) – a list of role names of which the user must have one
    - ✓ DenyAll() – Nobody has access; used at the end of the file to deny access to all nonspecified URL and method combinations

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

PAS for OpenEdge Configuration – oeablSecurityJWT.csv

- Used when client.login.model=oauth2
- JWT might not contain roles
- Authorization for JWT is based on scope of the token
- Same structure as oeablSecurity.csv
- Different functions
  - hasAnyAuthority() instead of hasAnyRole()

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Troubleshooting

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Troubleshooting

- Logging can be found in <instance folder>\logs
  - <ablapp>.<datestamp>.log
  - <ablapp>.authn.<datestamp>.log
  - <ablapp>.authz.<datestamp>.log
- Additional logging can be enabled in <instance>\ablapps\<ablapp>\conf\logging-ablapp.xml
  - This applies to all webapps connected to the ABLApp
  - Enable modules by uncommenting lines
  - Set logging level to DEBUG for more information
  - Logging will be put in the <ablapp>.<datestamp>.log file

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

```
<!--  Spring Security OAuth2 -->
<!--
        <logger name="org.springframework.security.oauth2" level="INFO" />
        <logger name="org.springframework.security.jwt" level="INFO" />
-->

<!--  Spring Security SAML -->
<!--  Comment or uncomment as a block of class loggers
          <logger name="org.springframework.security.saml2" level="INFO"/>
          <logger name="org.opensaml" level="INFO"/>
          <logger name=" PROTOCOL_MESSAGE " level="INFO"/>
-->
```

```
<!--  Spring Security OAuth2 -->
<!-- -->
        <logger name="org.springframework.security.oauth2" level="DEBUG" />
        <logger name="org.springframework.security.jwt" level="DEBUG" />

<!--  Spring Security SAML -->
<!--  Comment or uncomment as a block of class loggers -->
          <logger name="org.springframework.security.saml2" level="DEBUG"/>
          <logger name="org.opensaml" level="DEBUG"/>
          <logger name=" PROTOCOL_MESSAGE " level="DEBUG"/>
```

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Workshop



- Configure the PAS for OpenEdge to validate tokens
- Use Keycloak as an identity provider
- Use Bruno as API client to test configuration

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Keycloak

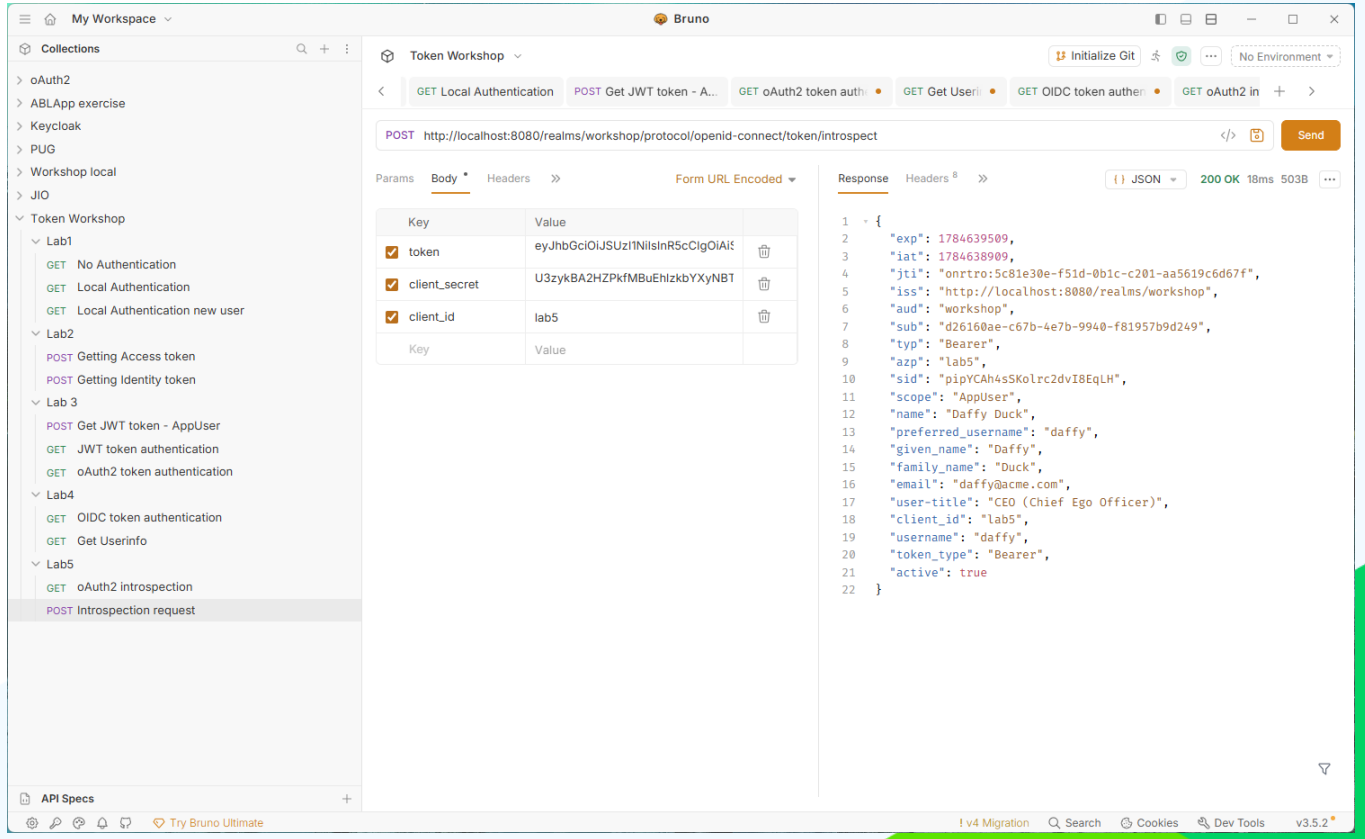
- Identity and Access Management (IAM) server
- Open source
- OpenID Connect & SAML
- Free download
- Web-based console
  - www.keycloak.org



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Bruno

- API client
- Open source
- Local storage of collections
- Git integration
- Free download
  - www.usebruno.com



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

News You Can Use

- Learn how to access the **OpenEdge AI Assistant**
- Get started with the **OpenEdge MCP Server**
- Book a demo of **Progress Agentic RAG**
- Modernize to **OpenEdge 12.8** today
- Explore what's new in **OpenEdge 13.0**
- Work with **Progress Professional Services** to accelerate adoption
- Learn about **security capabilities** across OpenEdge
- Explore the **developer tools** available for OpenEdge today
- Connect with us: **Discord, Community, CVP and more**



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.


Progress®