Making ClickHouse® Safe for AI: Using OAuth in Altinity Builds

Andrey Zvonov & Robert Hodges
Altinity Inc
28 April 2026

Run Open Source ClickHouse® Better

Altinity.Cloud Enterprise Support

Introducing OAuth and Token-Based Login

What is “OAuth”? It’s a scheme for access using tokens

OAuth2 aka Open Authorization 2.0

A standard for **authorizing** access to online services

OIDC aka Open ID Connect

A standard for **authenticating** access to online services

Benefits of token-based authentication and authorization

Enables single sign-on (Google, Azure AD, Keycloak...)

Widely supported

Eliminates complex password management for users

Of course, there is no free lunch

OAuth is complicated

Open source ClickHouse does not support it fully

Of course, there is no free lunch

OAuth is complicated

~~Open source ClickHouse does not support it fully~~

Altinity Antalya builds in OIDC + Auth2 (100% open source)

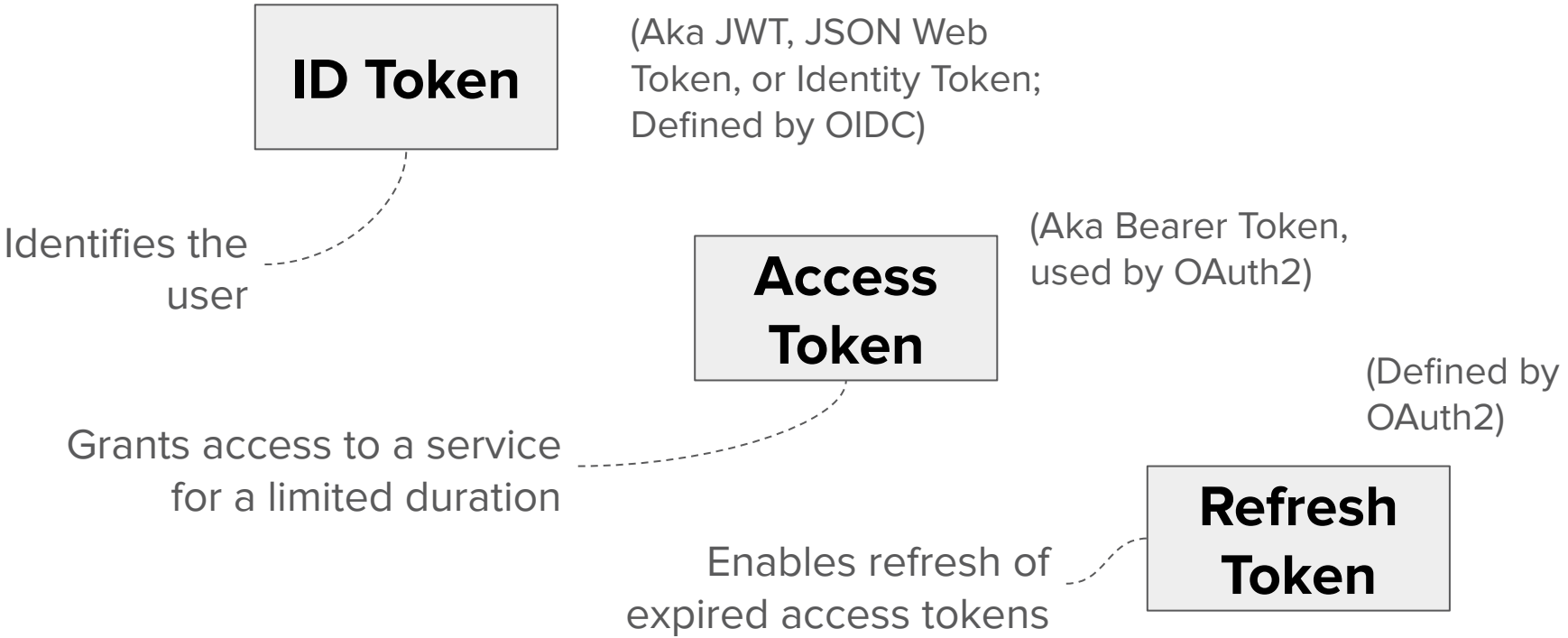
OAuth / OIDC is now available in Altinity Antalya Builds

- Not supported in in Altinity Stable (for now)
- Use Altinity Antalya version 25.8.16.20001 or higher
- Location of builds
  - Packages: https://builds.altinity.cloud/#altinityantalya
  - Containers: https://hub.docker.com/r/altinity/clickhouse-server/tags?name=antalya

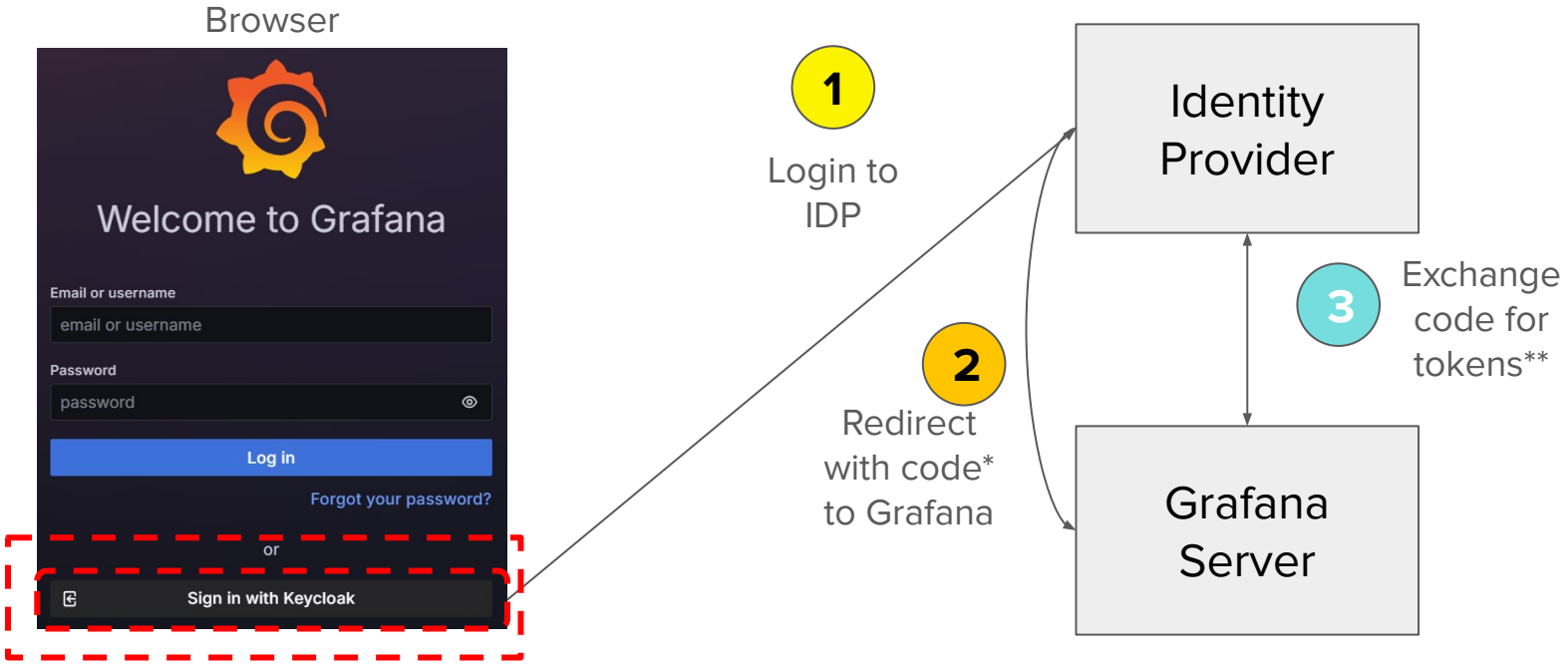
OAuth in action

Demo time!

Token-based auth builds on 3 different kinds of tokens



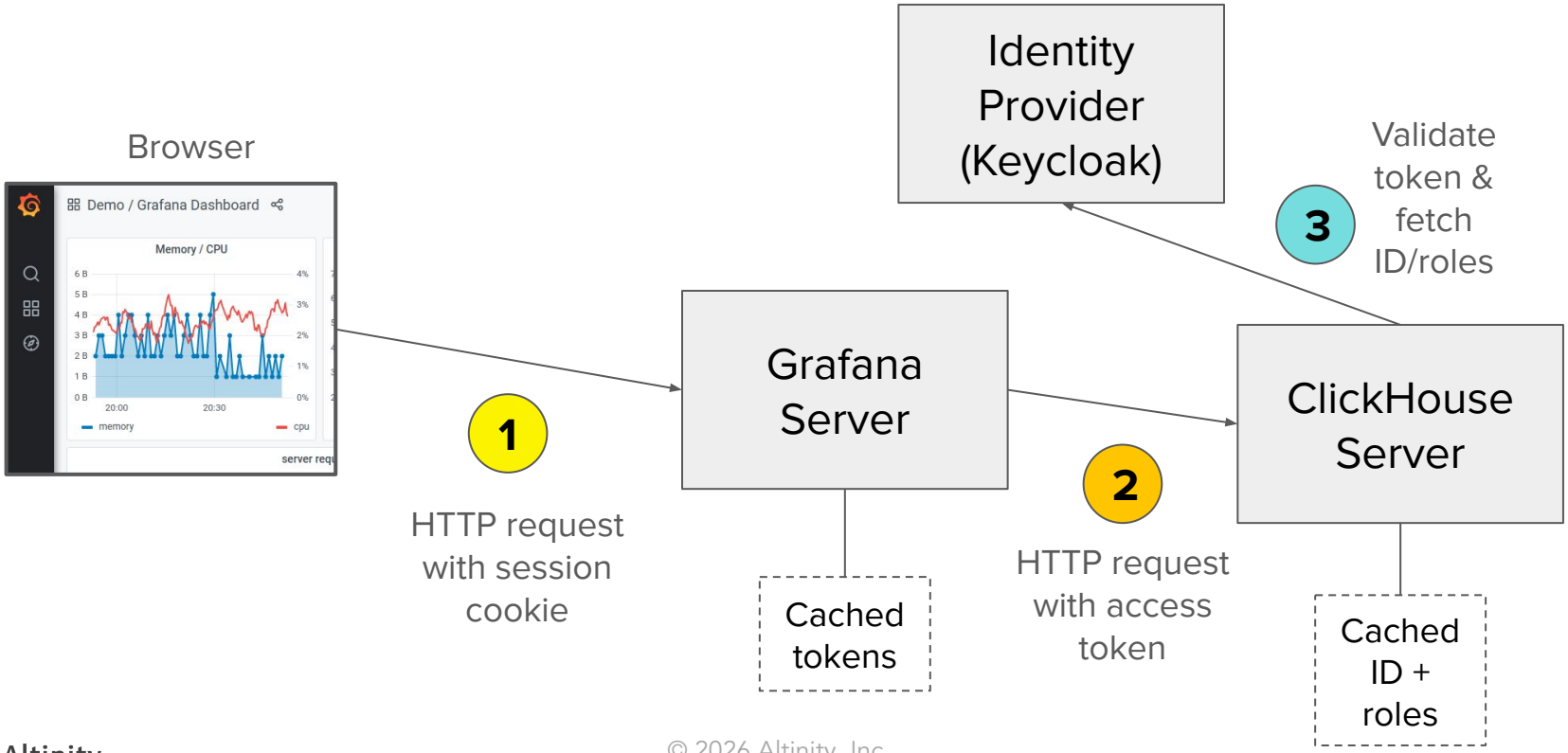
How authentication works in OAuth



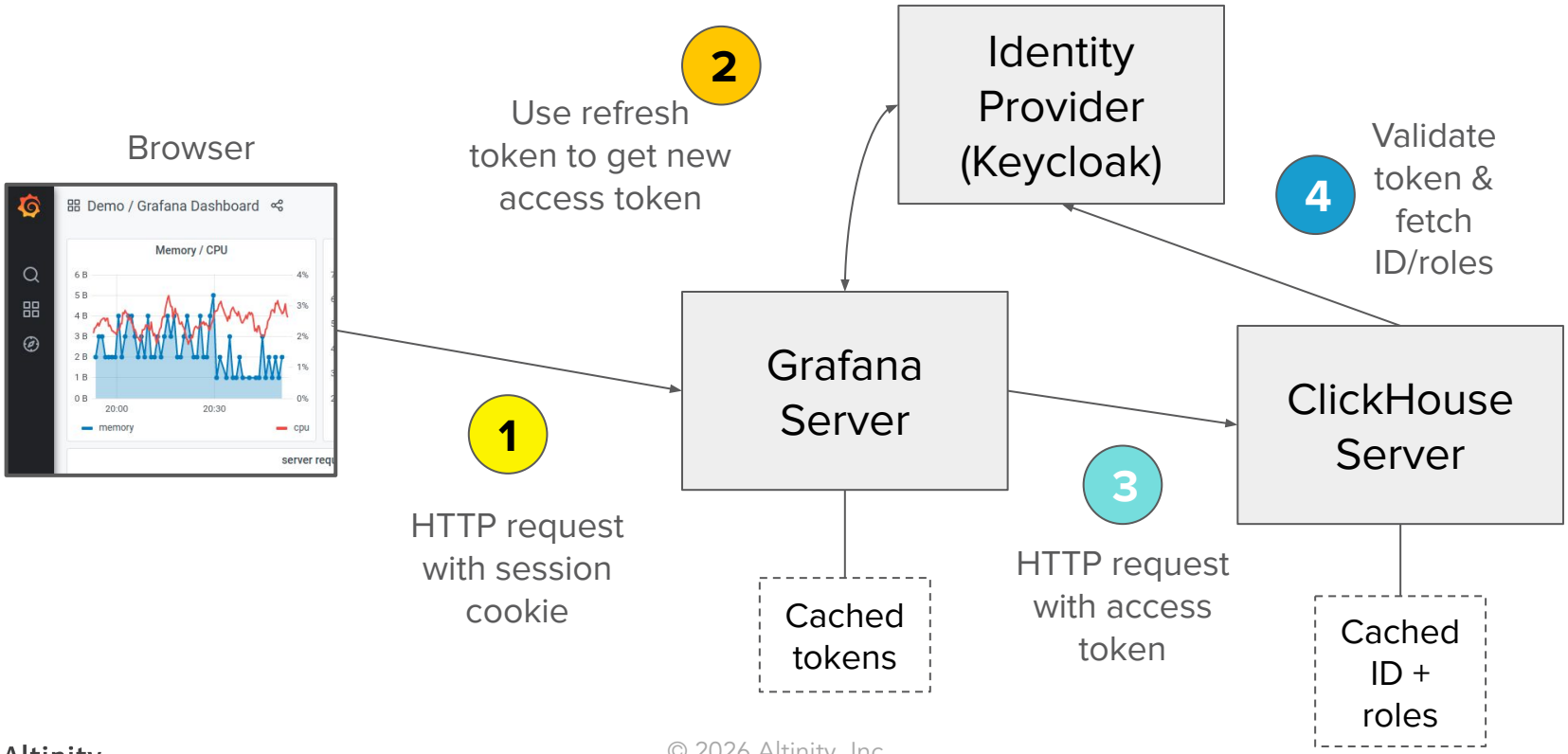
* Plus a nonce and other info for security

** Identity Token, Access Token, Refresh Token

What happens when Grafana accesses ClickHouse?

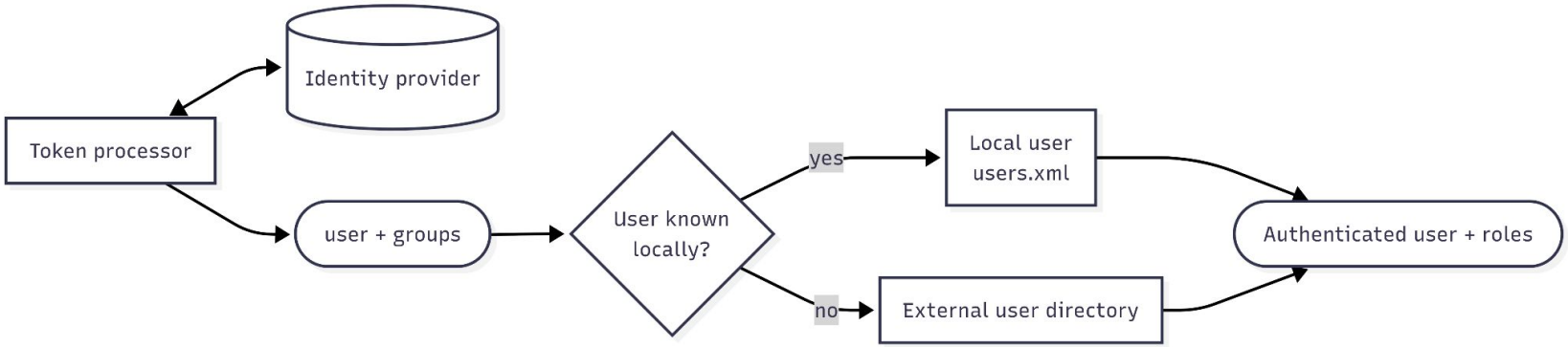


What happens when the access token expires?

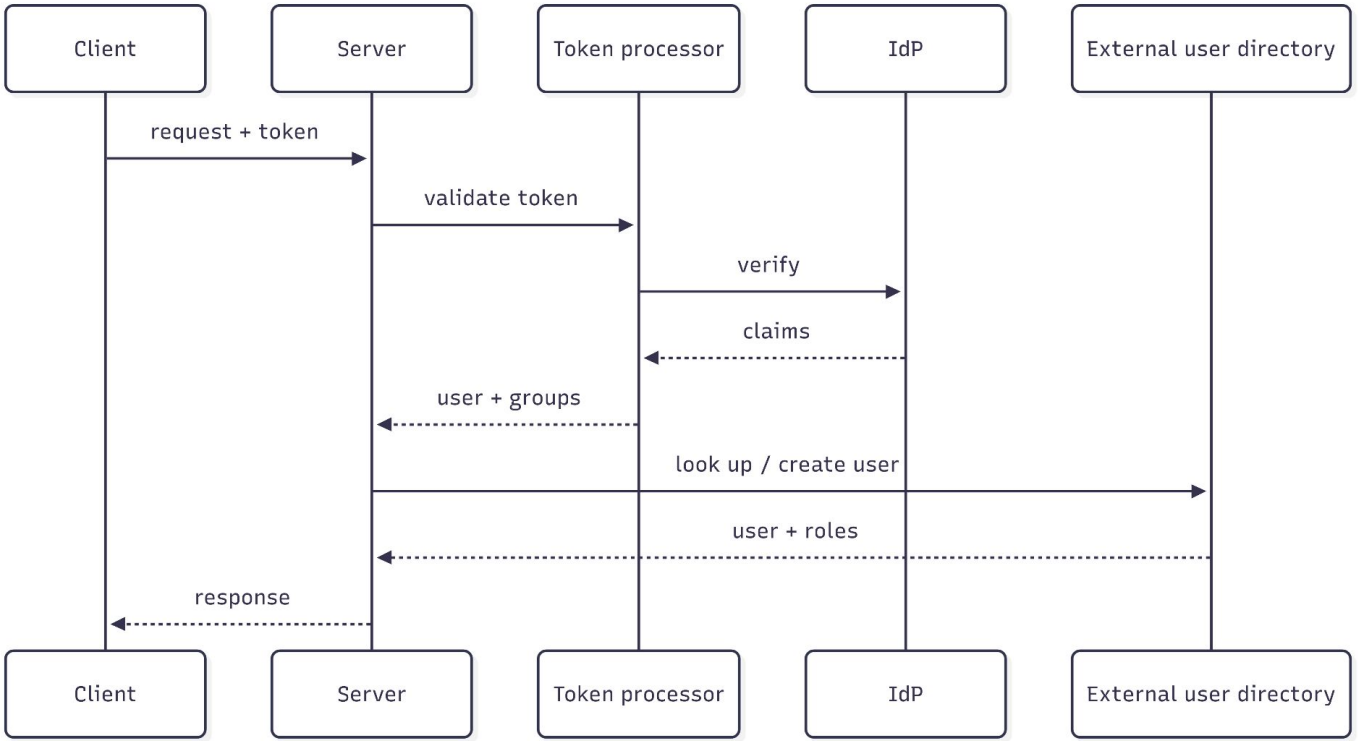


Configuring OAuth in Antalya Builds

ClickHouse Token Auth Flow



ClickHouse Token Auth Flow (2)



Configuration Example

```
<clickhouse>
    <token_processors>
        <keycloak>
            <type>OpenID</type>
            <userinfo_endpoint>uri1</userinfo_endpoint>
            <token_introspection_endpoint>uri2</token_introspection_endpoint>
            <jwks_uri>uri3</jwks_uri>
            <token_cache_lifetime>60</token_cache_lifetime>
            <username_claim>email</username_claim>
        </keycloak>
    </token_processors>
    <user_directories>
        <token>
            <processor>keycloak</processor>
            <common_roles>
                <general_role />
            </common_roles>
            <roles_transform>s/-/_/g</roles_transform>
        </token>
    </user_directories>
</clickhouse>
```

system.session_log is a handy tool to track users

```
$ cat /etc/clickhouse-system/config.d/session-log.xml
<!-- Enable session log to show logins. -->
<clickhouse>
    <session_log>
        <database>system</database>
        <table>session_log</table>
        <partition_by>toYYYYMM(event_date)</partition_by>
        <flush_interval_milliseconds>7500</flush_interval_milliseconds>
        <max_size_rows>1048576</max_size_rows>
        <reserved_size_rows>8192</reserved_size_rows>
        <buffer_size_rows_flush_threshold>524288</buffer_size_rows_flush_threshold>
        <flush_on_crash>false</flush_on_crash>
    </session_log>
</clickhouse>
```

Some best practices for OAuth

- Keep access token and token cache expirations short
  - The shortest of these two is the length it takes to cancel a user
  - If it's too short it will beat up your IDP
- Keep refresh token expirations relatively long
- Apply user profiles to set processing limits on accounts
- Be wary of granting write access, especially to AI agents
- Use LLMs to help pick off configuration problems and bugs

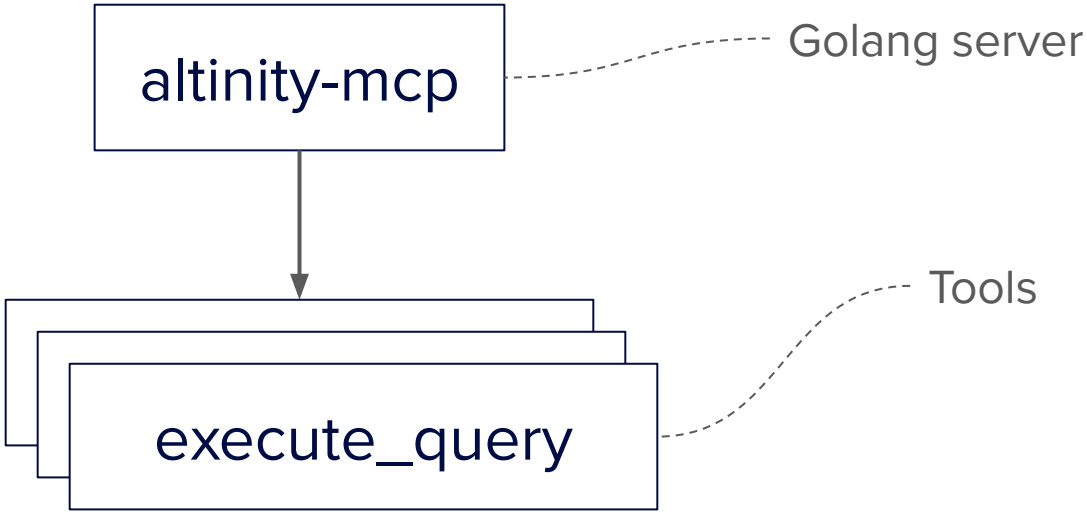
How we plan to use OAuth in the Altinity MCP server

How can AI applications connect to ClickHouse?

Model Context Protocol

A protocol to connect AI models to databases, development tools, and other useful “stuff”

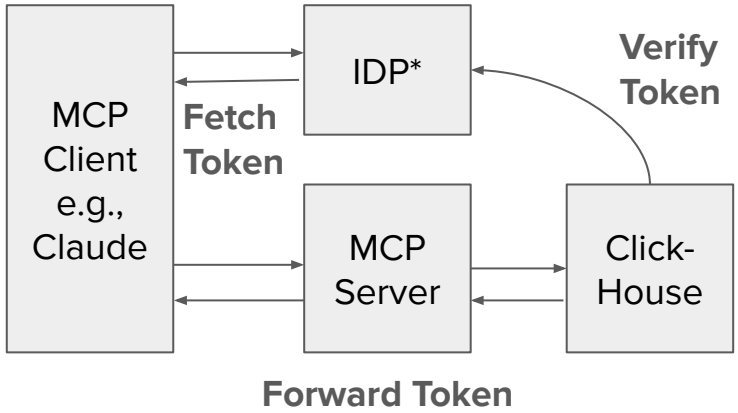
Altinity has an MCP server in development



Demo Time!!

altinity-mcp offers two different OAuth schemes

"Forward Mode"

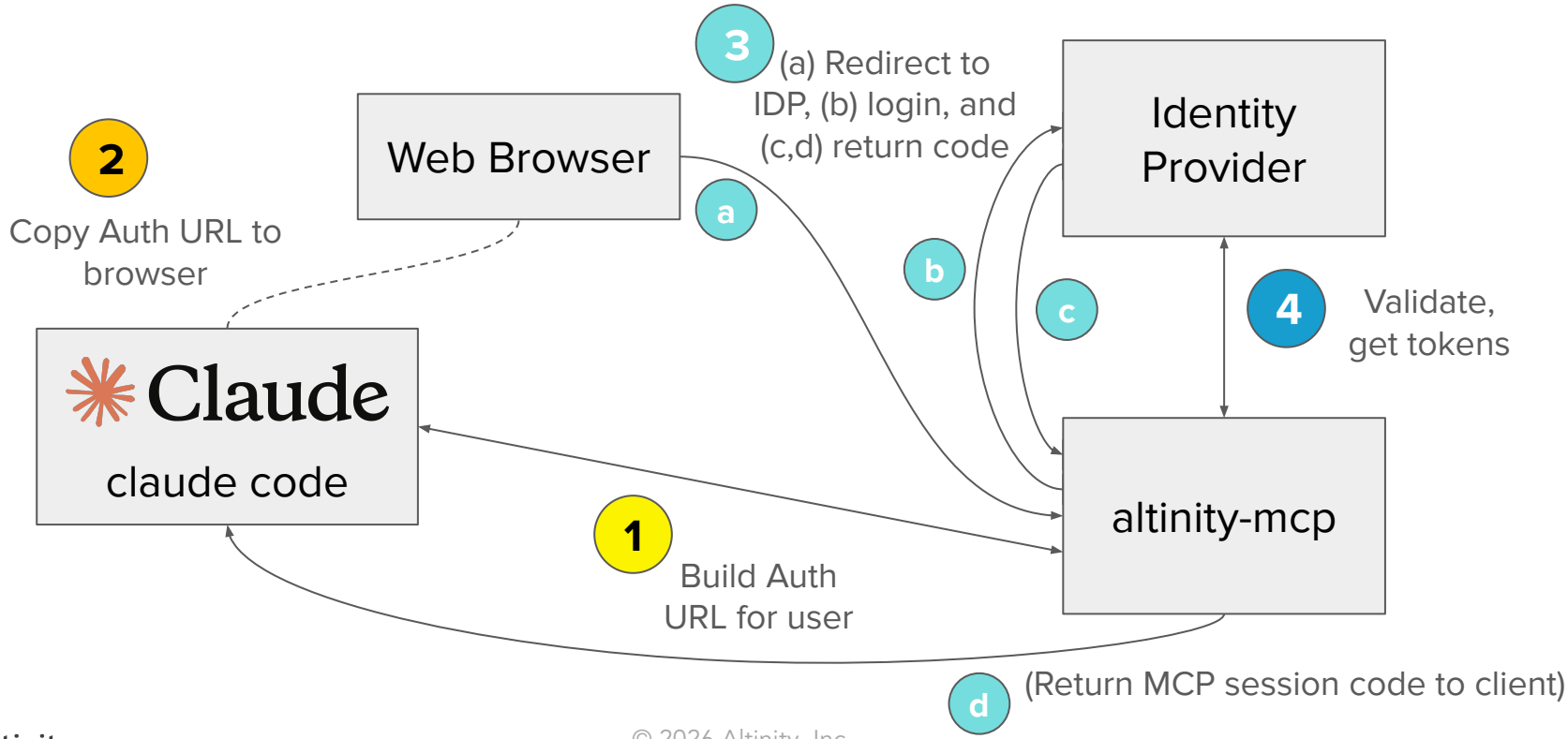


"Gating Mode"

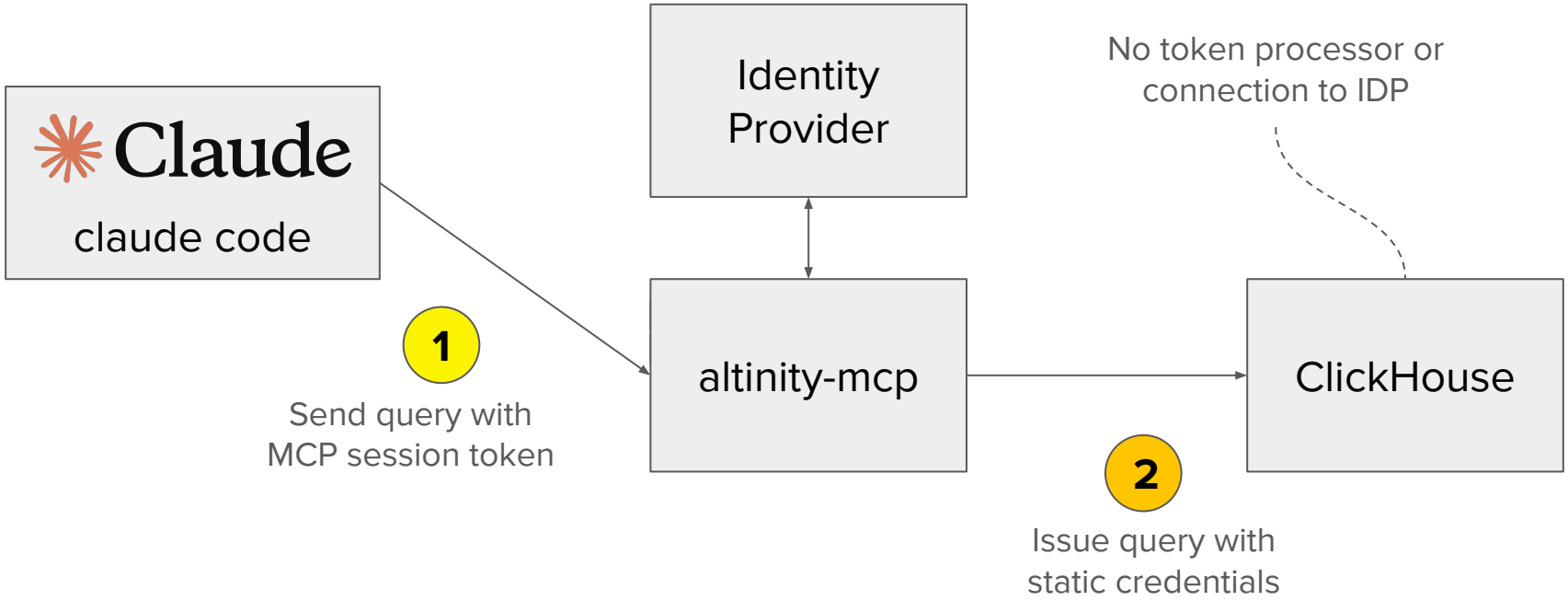
(Way too complicated to show here)

* Identity Provider

How browser-based login enables gating mode access



MCP operations on ClickHouse are simple in gated mode



Roadmap and wrap-up

Roadmap for OAuth Features

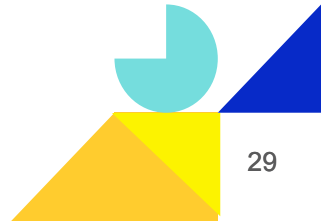
Release date	Feature
(Done-Q4 2025)	Keycloak Identity Provider Integration
(Done-Q4 2025)	Azure Active Directory (aka MS Entra ID) Integration
(Done-Q4 2025)	Grafana community plugin token-based auth certification
Q2 2026	clickhouse-client token-based auth (includes non-interactive scripts)
Q2 2026	Tableau JDBC connector token-based auth
Q2 2026	Python certification (clickhouse-connect)
Q2 (late)	UI support in Altinity.Cloud

Work together with us on OAuth support

All work on OIDC / OAuth2 is licensed under Apache 2.0

Many features are joint investments with customers

Contact us to find out more



Where to learn more?

Ask your favorite AI coding agent

https://github.com/Altinity/oauth-examples

Full documentation is on the way!

May 20 Webinar!

“AI Toys in the Attic”

Thank you!
Questions?

Contact us:

- Slack - https://altinity.com/slack
- Find out more - http://altinity.com

We’re hiring!

ALTINITY

Ensure your compute and storage match database needs

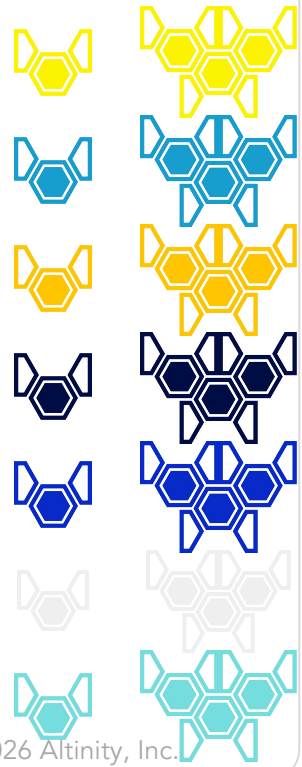
1. AWS Graviton instances are fast and cheap
2. Block storage enables vertical scaling and can failover when host dies
3. NVMe SSD gives the best raw performance but is tied to a single host and hard to configure in Kubernetes
4. Object storage is permission to play for large analytic databases

Icons- Transparent

Clickhouse (Native) Cluster

Director Cluster

Swarm Cluster



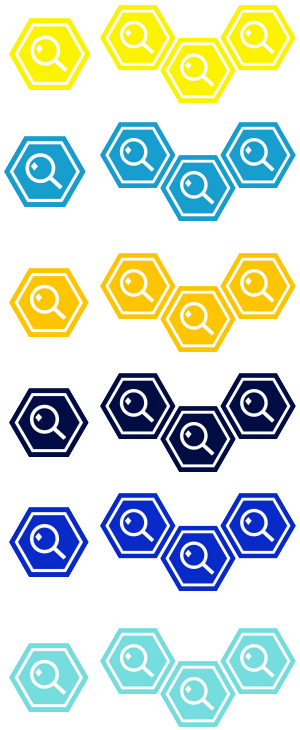
Keeper

Other

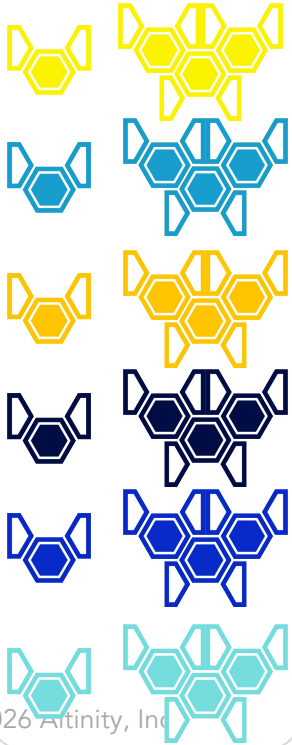
Icons-Stackable on White Backgrounds

Clickhouse (Native) Cluster

Director Cluster



Swarm Cluster



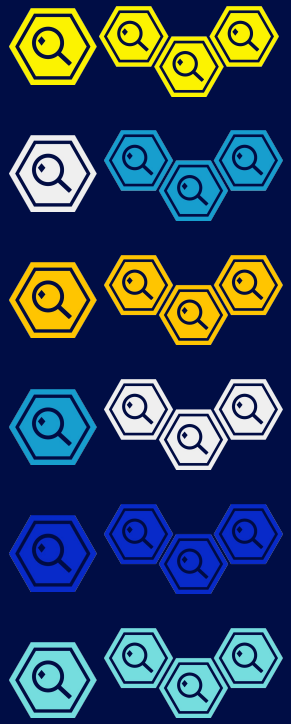
Keeper

Other

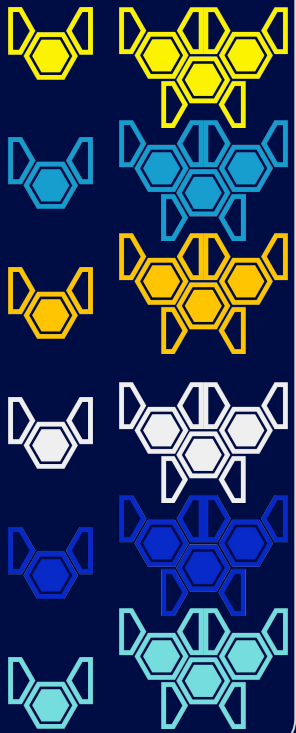
Icons-Stackable on Dark Backgrounds

Clickhouse (Native) Cluster

Director Cluster

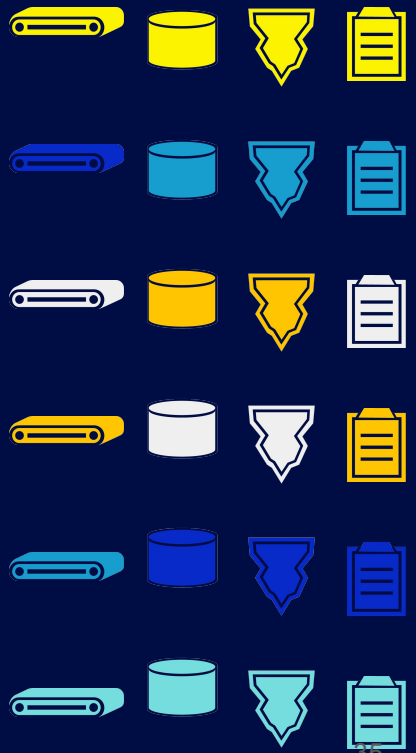


Swarm Cluster



Keeper

Other



Rescaling – adding more disks

```
templates:
  volumeClaimTemplates:
    - name: disk1
      spec:
        accessModes:
          - ReadWriteOnce
        resources:
          requests:
            storage: 100Gi
```

```
  settings:
    storage_configuration/disks/disk2/path: /var/lib/clickhouse2/
    storage_configuration/policies/default/volumes/default/disk: default
    storage_configuration/policies/default/volumes/disk2/disk: disk2
templates:
  volumeClaimTemplates:
    - name: disk1
      spec:
        accessModes:
          - ReadWriteOnce
        resources:
          requests:
            storage: 100Gi
    - name: disk2
      spec:
        accessModes:
          - ReadWriteOnce
        resources:
          requests:
            storage: 100Gi
  podTemplates:
    - name: default
      spec:
        containers:
          - name: clickhouse-pod
            volumeMounts:
              - name: disk1
                mountPath: /var/lib/clickhouse
              - name: disk2
                mountPath: /var/lib/clickhouse2
```

Restart rules

```
configurationRestartPolicy:
  rules:
    - version: "*"
      rules:
        - settings/*: "yes"
        # single values
        - settings/access_control_path: "no"
        - settings/dictionaries_config: "no"
        - settings/max_server_memory_*: "no"
        - settings/max_*_to_drop: "no"
        - settings/max_concurrent_queries: "no"
        - settings/models_config: "no"
        - settings/user_defined_executable_functions_config: "no"
        # structured XML
        - settings/logger/*: "no"
        - settings/macros/*: "no"
        - settings/remote_servers/*: "no"
        - settings/user_directories/*: "no"
        # these settings should not lead to pod restarts
        - settings/display_secrets_in_show_and_select: "no"
        - zookeeper/*: "no"
        - files/*.xml: "yes"
        - files/config.d/*.xml: "yes"
        - files/config.d/*dict*.xml: "no"
        - files/config.d/*no_restart*: "no"
        # exceptions in default profile
        - profiles/default/background_*_pool_size: "yes"
        - profiles/default/max_*_for_server: "yes"
```

Defined in operator configuration

Can be altered using ClickHouseOperatorConfiguration resource

Plan to utilize `system.server_settings.changeable_without_restart` eventually

Trick to avoid restarts for certain configuration files