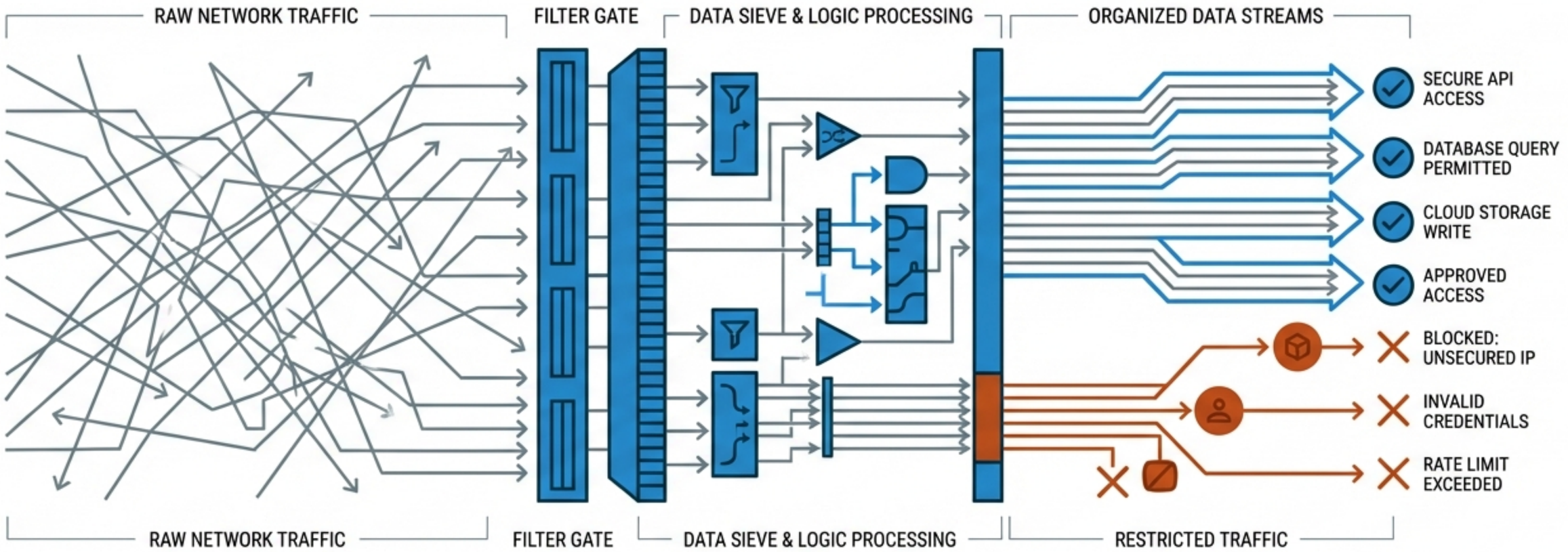
Logical Firewalls & Granular Control

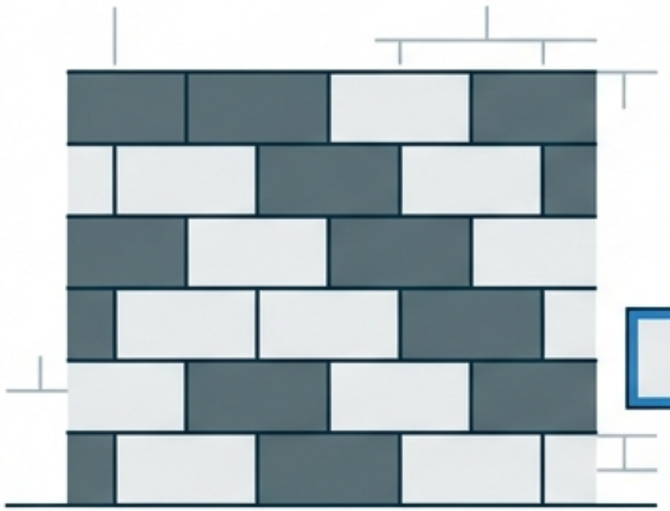
Securing Multi-Tier Data Access in the Hybrid Cloud



From coarse IP blocking to Attribute-Based Access Control (ABAC) with OpenLink Session Rules

The Security Gap in PaaS and Hybrid Clouds

Perimeter



Cloud IP Firewall. Scopes protection only to Client IP Addresses. Blind to application context.

The Vulnerability Gap

The Blind Spot: No visibility into **Client OS**, **Application Name**, or **Session Intent**.

No Context

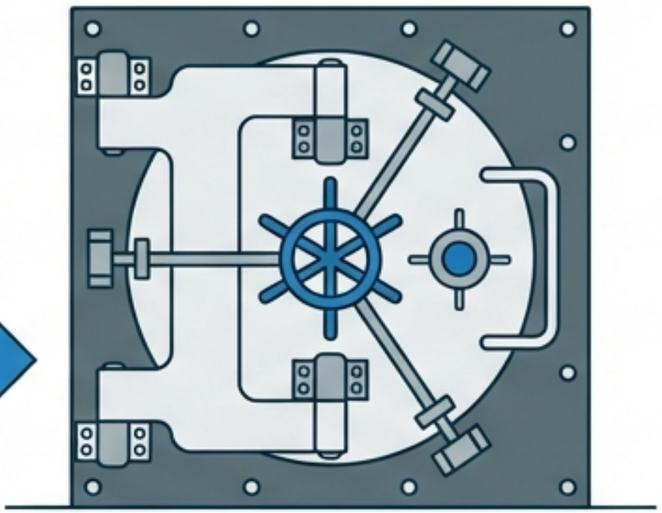
Client OS (Missing)

App Name (Unknown)

Session Intent (Unseen)

Unverified Connection

Database Core



DBMS Internal Roles. Handles user passwords and permissions *after* connection is established.

Consequence: Database administrators are forced to rely solely on IP scoping and internal DB roles, missing crucial context. Hyper-connectivity exposes data flow and privacy risks that simple IP blocking cannot address.

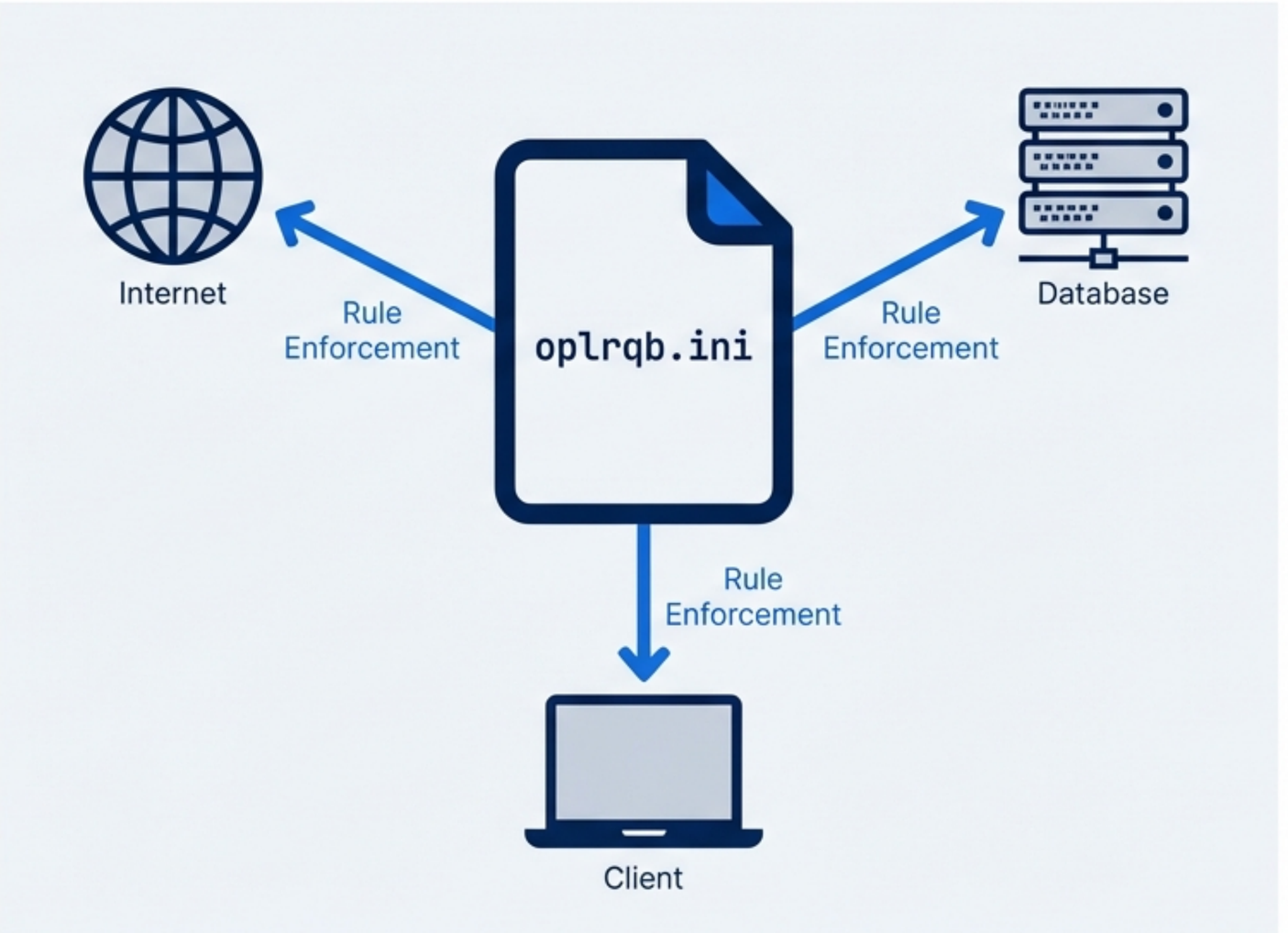
The Solution: The OpenLink Session Rules Book

Core Concept: The Logical Firewall

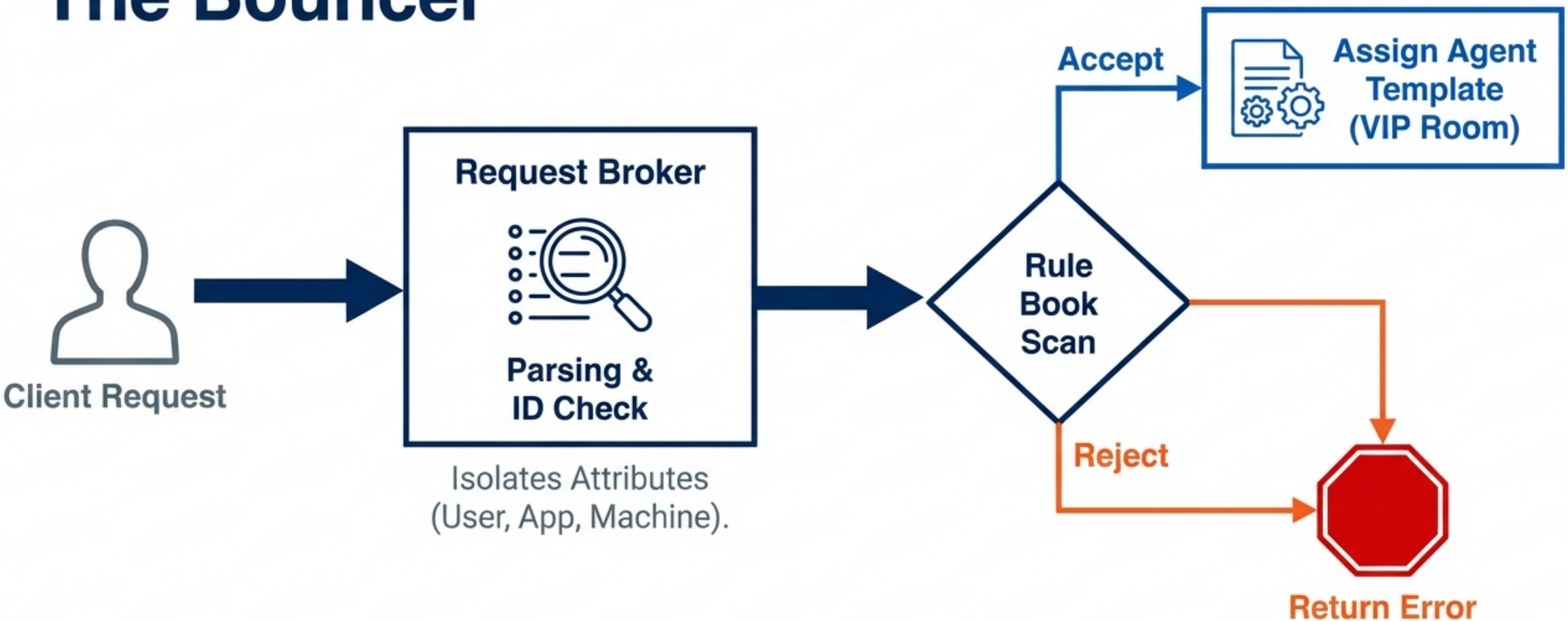
Instead of just blocking ports, we process Session Rules stored in a central text-based repository.

Key Capabilities:

1. **Centralized Control**: Configure infrastructure from a single point.
2. **Declarative Logic**: Use templates to determine client-server interaction.
3. **Context-Aware**: Decisions based on 'Connection Attributes'—Who, How, Where, and What.



Architecture: The Request Broker as “The Bouncer”



The Request Broker enforces the rules before a database session is even instantiated.

The Six Connection Attributes

Every connection is defined by these six data points used for evaluation.

User Configurable - Input by Human

Domain

The logical identifier for the OpenLink Agent (e.g., '`Oracle 8`').

User

The user identity claiming the connection.

Database

Specific target database name (e.g., 'pubs').

Non-User Configurable - Derived by System

OpSys

Client OS (e.g., 'win32', 'unix').

Machine

IP address or Network Alias.

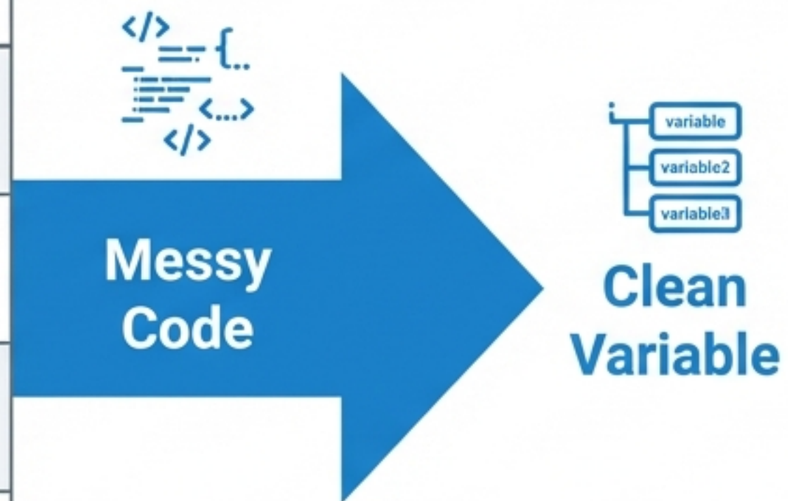
Application

Client software executable (e.g., 'MSACCESS').

Simplifying Logic with Aliases

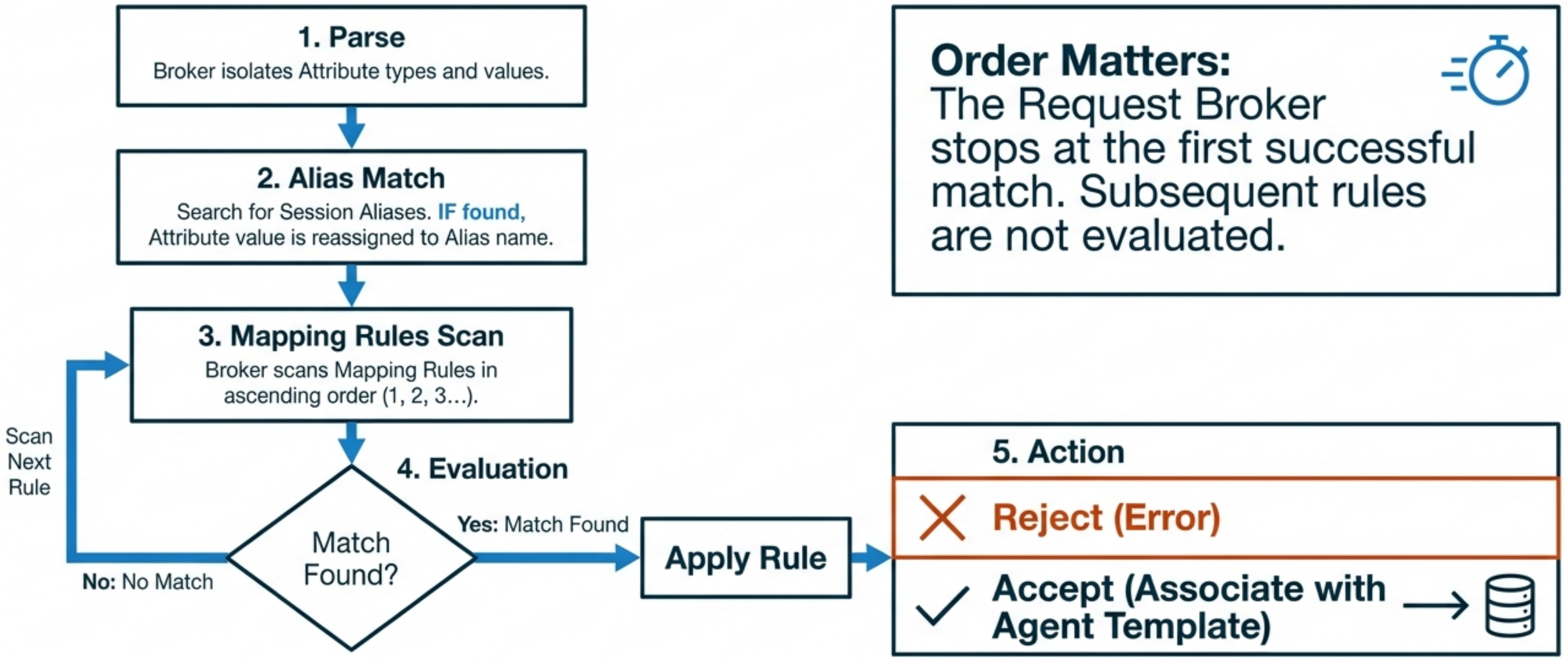
Aliases allow you to group complex Regular Expressions (Regex) into readable variable names. This separates the definition of “Who” from the rules of “What they can do.”

Attribute Type	Complex Regex (The Input)	Simple Alias (The Output)
Domain	`Oracle 8`	ora8
Machine	`123.123.*`	MyNetwork (LAN)
User	`^Test$\|^Mary$`	SalesTeam
Application	`MSACCESS|EXCEL`	OfficeApps



The Logic Flow: From Request to Session

A step-by-step process showing how an incoming request is parsed, matched, and assigned.



Recipe 1: The Global Lockdown

Goal: Enforce Read-Only access unconditionally across all ODBC/JDBC clients.

Configuration

The Rule

When:

*

Matches: Any Domain/Attribute.

↓

Then:

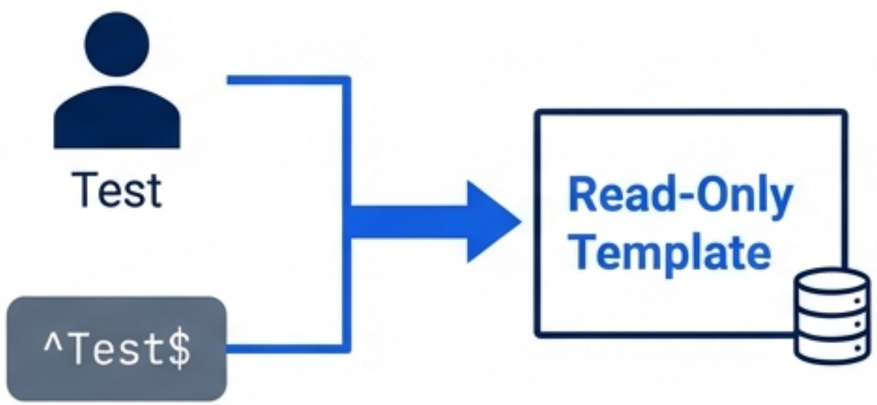
ACCEPT and map to '**ReadOnly_Agent**' template.

Outcome: Any update attempt (**INSERT**/**UPDATE**/**DELETE**) returns a database **error**, regardless of database user privileges.

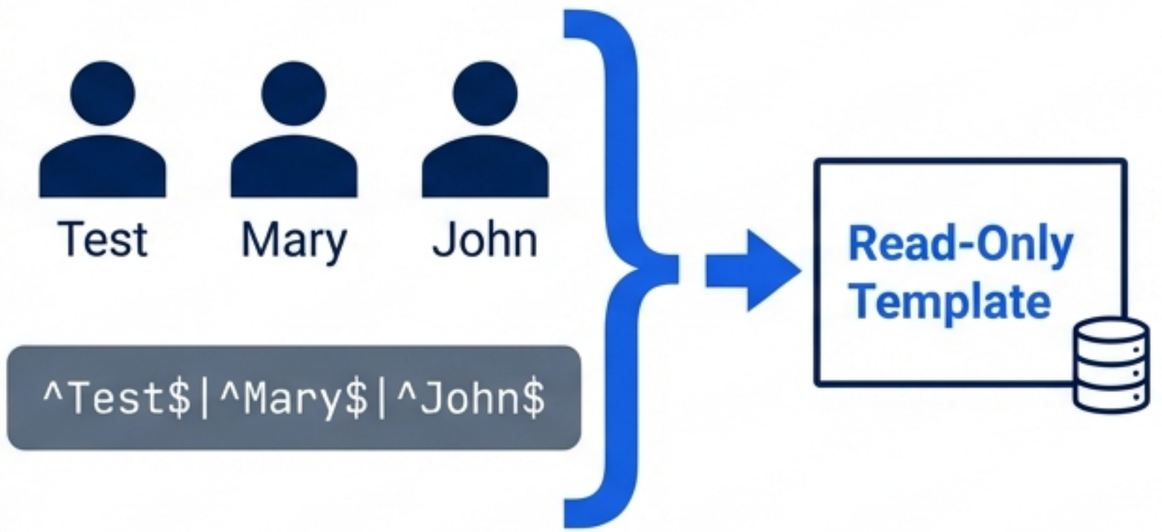
Recipe 2: Precision User & Group Control

Regex Toolkit
- `^` : Start of string
- `$` : End of string
- `|` : OR condition

Scenario: Single User Lockdown



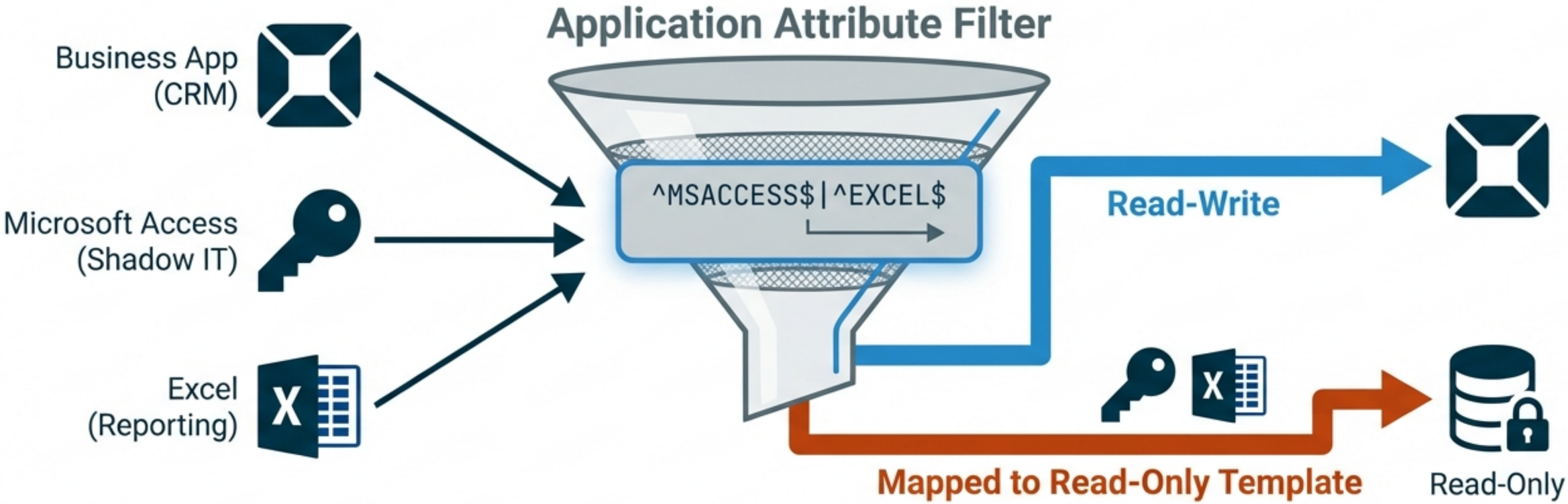
Scenario: Group Definition



"Granular control applied at the connectivity layer, before the database is even touched."

Recipe 3: Application Whitelisting & Shadow IT

Challenge: Database engines cannot distinguish between a legitimate CRM app and a user utilizing Microsoft Access to download sensitive tables.



Result: Prevent data leakage by forcing risky reporting tools into Read-Only mode.

Recipe 4: The ‘Geo-Fence’ (LAN vs. WAN)

Using Rule Ordering to create conditional access.

1 Rule #1: The Exception (Office LAN)

```
Machine Attribute starts with 123.123.*
```

→ **Action: Map to ReadWrite_Agent**

2 Rule #2: The Catch-All (Remote/Internet)

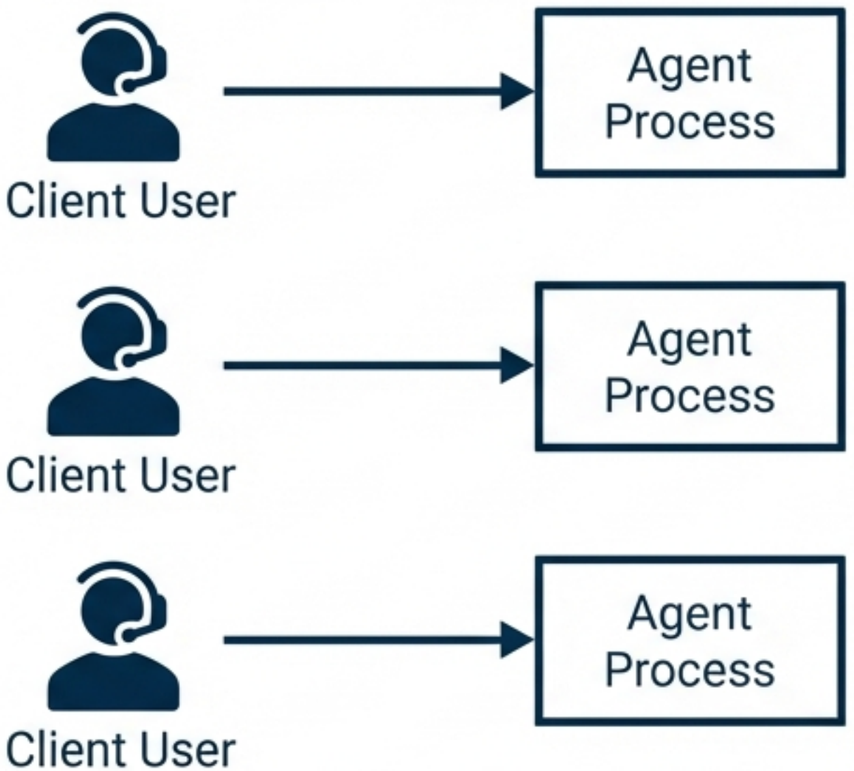
```
* (Wildcard)
```

→ **Action: Map to ReadOnly_Agent**

The Broker scans in **ascending order**. Local users match Rule 1 and **stop**. Remote users **fail** Rule 1 and **fall through** to Rule 2.

Resource Management: Optimizing Agent Reuse

The Problem



Standard:
1 Client = 1 Agent

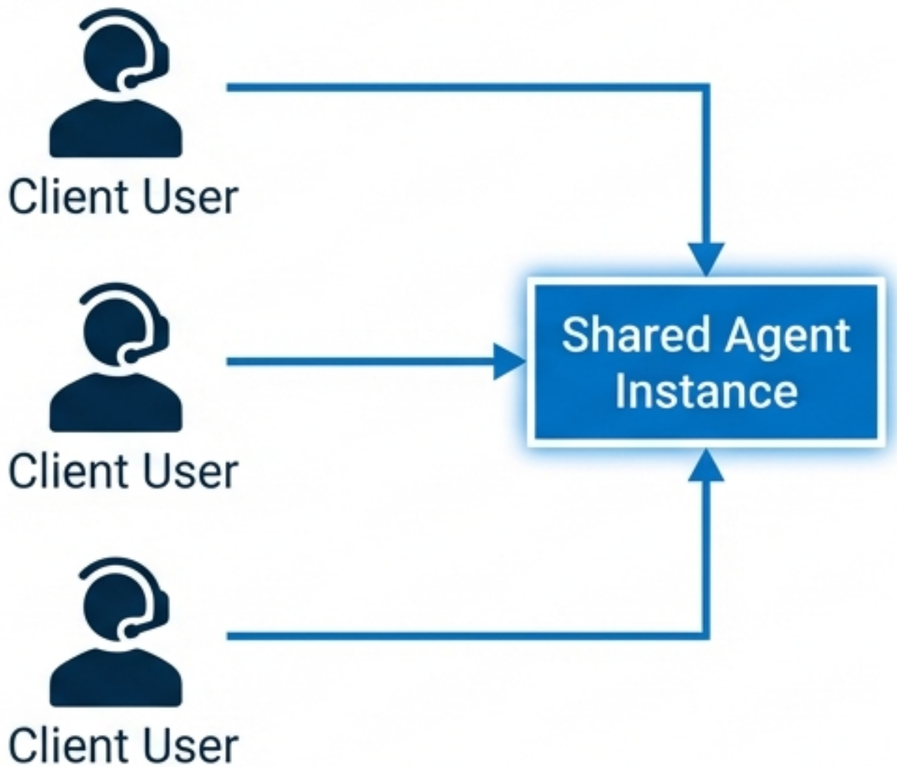
High RAM Usage.

Configuration:

Set `'ReUse = Always'` in the Agent Template.

A new instance is only spawned if the current one is busy processing a request.

The Solution: ReUse = Always

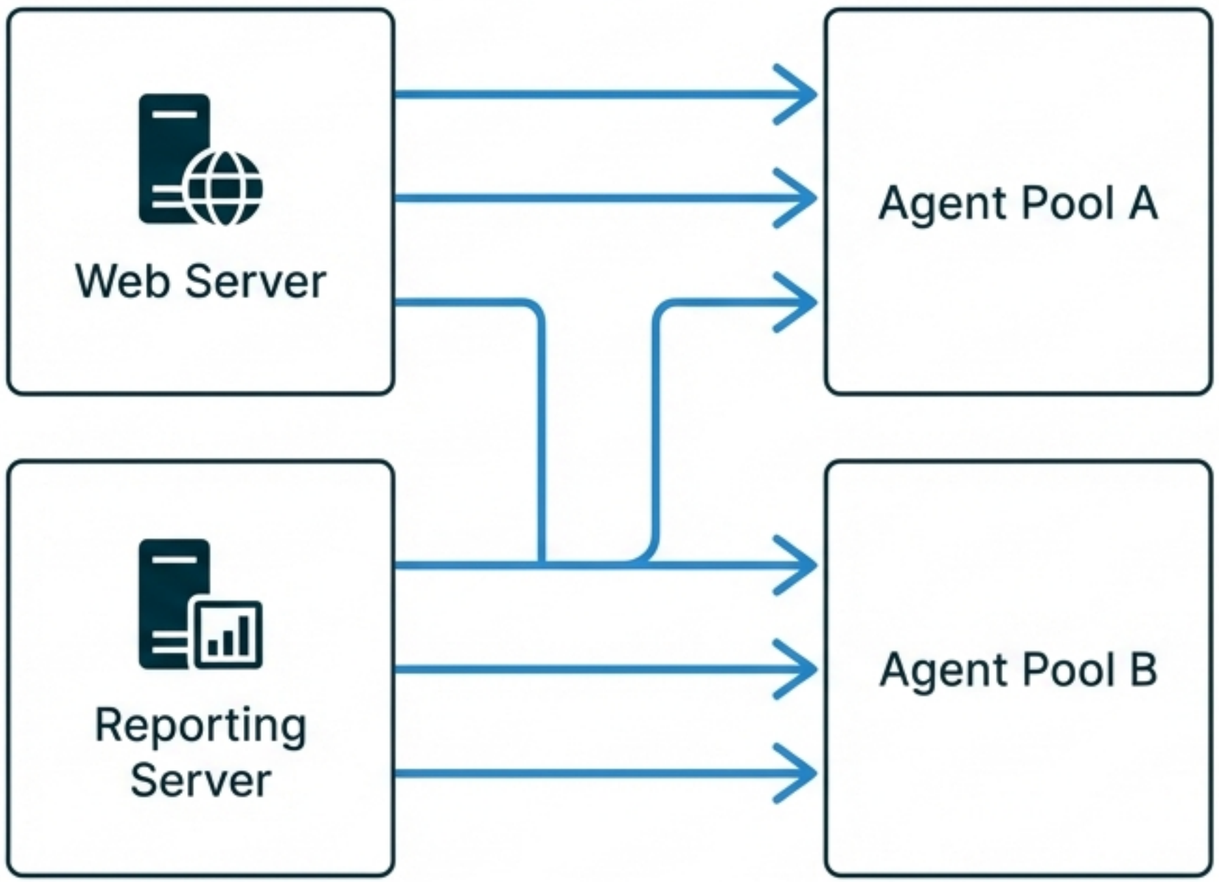


Optimized:
Many Clients = 1 Agent

Maximum Efficiency.

Advanced Tuning: Conditional Pooling

Balancing isolation with efficiency.



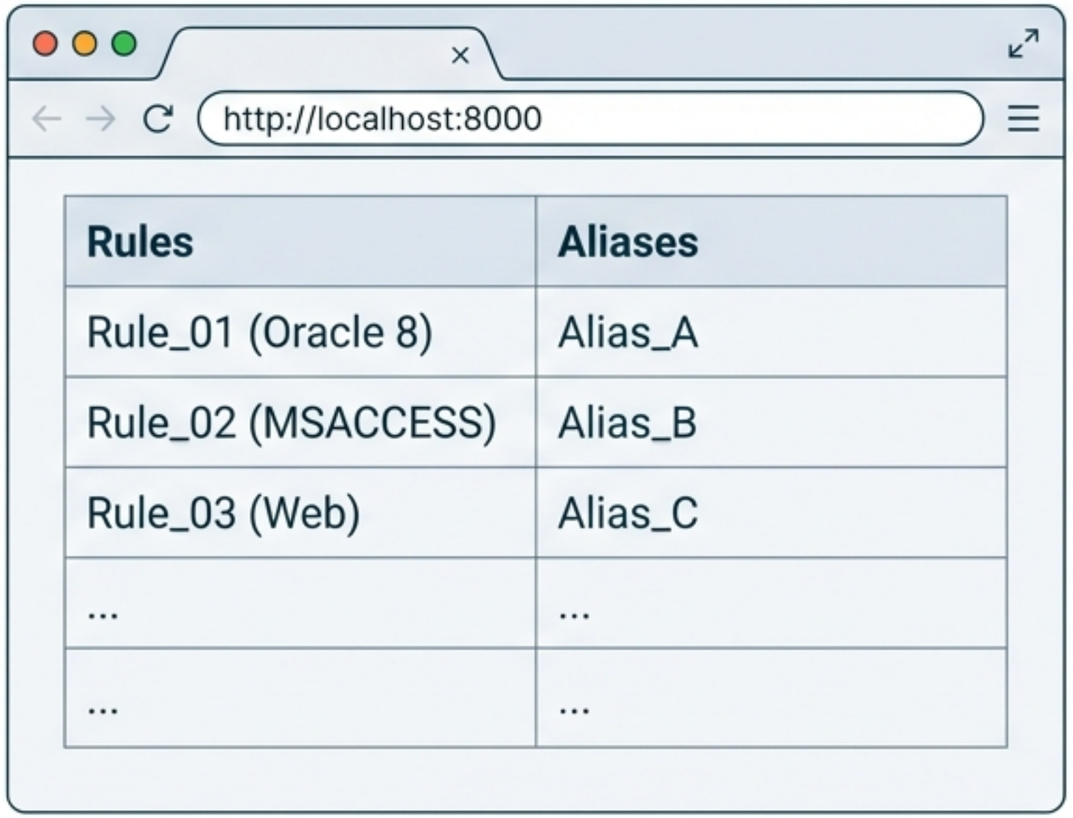
Configuration: Select 'Conditionally' and check 'When originating from the same machine'.

⚠ Prevents a heavy load from the Reporting Server (Machine B) from impacting the performance of the Web Server (Machine A).

This ensures that resource-intensive tasks on one machine do not degrade the service quality of critical applications on another.

Administration & Troubleshooting

The Interface



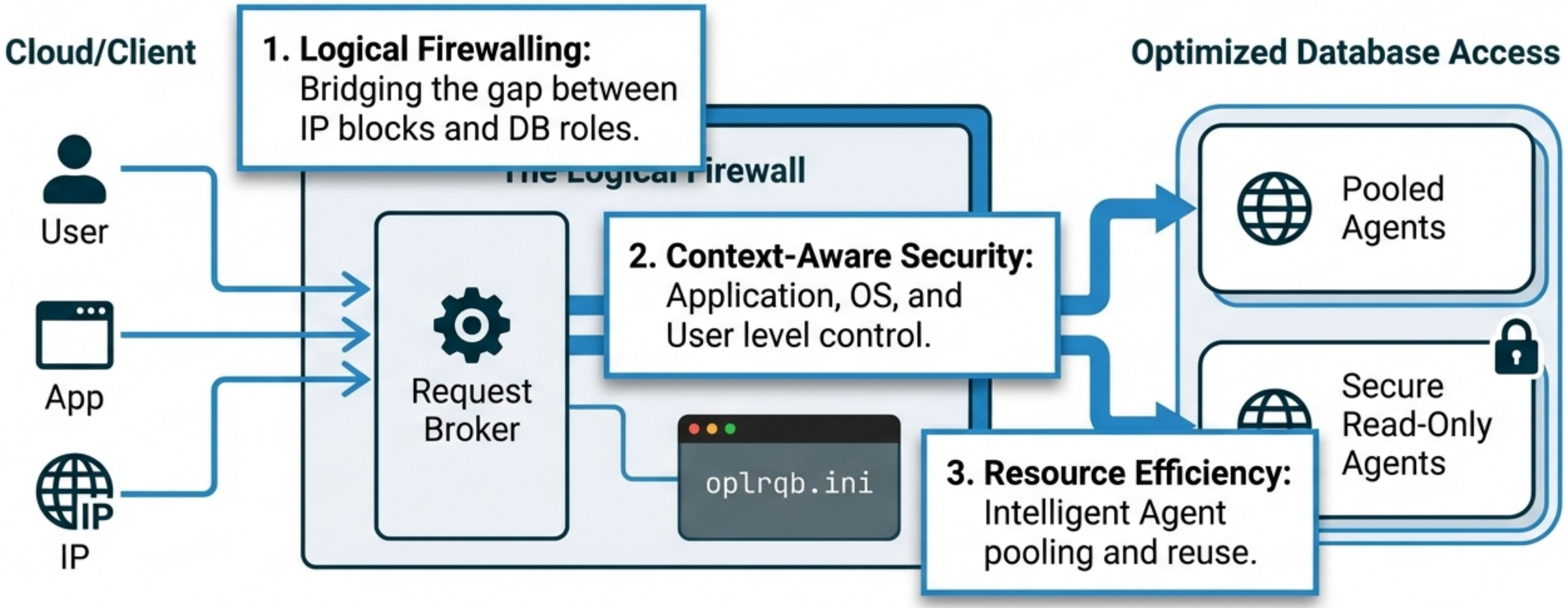
Rules	Aliases
Rule_01 (Oracle 8)	Alias_A
Rule_02 (MSACCESS)	Alias_B
Rule_03 (Web)	Alias_C
...	...
...	...

Debug Mode

```
oplrqb: client-machine.YourDomain called
(123.123.123.100)
oplrqb: request: domain=Oracle 8
oplrqb: opsys=win32 readonly=0
oplrqb: application=MSACCESS
```

Use Debug Mode to capture exact attribute values (e.g., ‘win32’) to ensure your Regex matches reality.

Total Infrastructure Control



In an era of hyper-connectivity, OpenLink Session Rules provide the granularity required to secure and optimize the Hybrid Cloud.