# Tech Brief: Leveraging Graph Databases for Access Management

## Introduction

Access Management (AM) is concerned with authenticating users and determining whether they have permission to access requested resources. Core to the ForgeRock Identity Platform™ is the design and implementation of access control policies. In its most common form, an access control rule is specified by subjects, objects, permissions, and conditions; and lets the AM system determine whether a user (subject) is able to perform an operation on a resource (object) under the current condition.

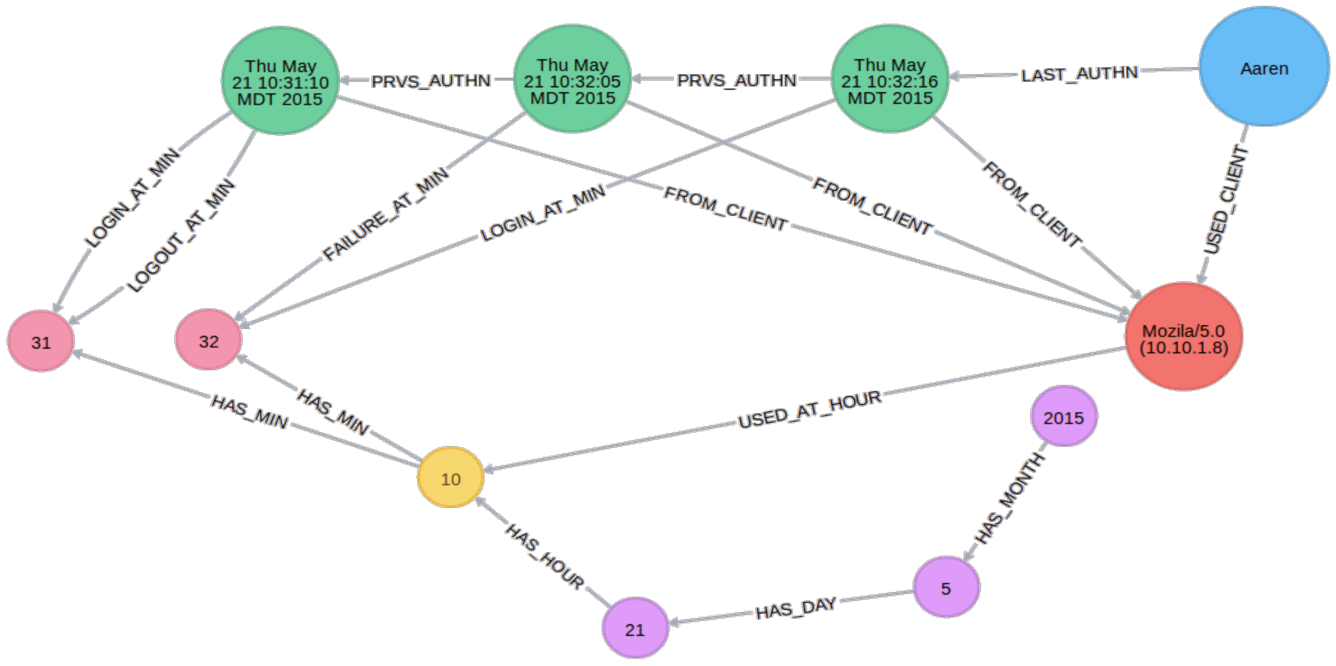
## Client-Based Access Control - A Graph Database Implementation

Nulli has developed a simple access management use case whereby ForgeRock OpenAM is used to protect resources. Users' authentication history is utilized to control access to those resources. To access the service, a user is required to be authenticated to OpenAM. However, basic authentication may not be sufficient. Depending on the client type, the user may or may not be granted access and additionally, further steps may be required.

We approach a solution through an enhancement that makes use of the Neo4j™ graph database to store and retrieve historical authentication data. Our OpenAM extension consists of two plugins: a post-authentication plugin (PAP) and a policy condition plugin. The former stores users' authentication history and the latter uses this data to make informed decisions on granting access to protected resources. The following provides a high level description of the two pieces.

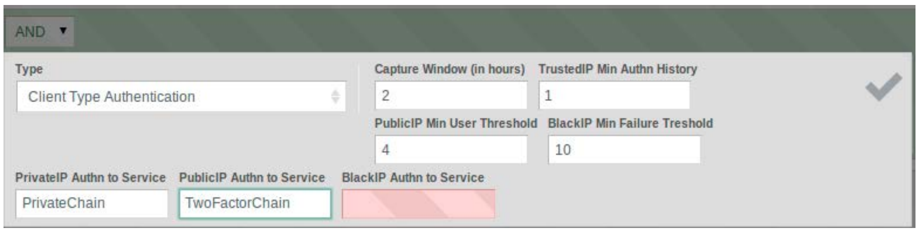
### *Updating Authentication History with PAP*

OpenAM PAP is invoked after the user completes an authentication process. We have provided a simple plugin that updates the authentication history of a user based on their authentication result which can be a failure, a successful login, or a logout. The authentication history graph is bootstrapped with OpenAM registered users and a calendar. Upon a (successful or failed) login event, a new authentication node is created and is located between the user and the past authentication nodes. The new node is also linked to the calendar as well as a client node which embodies client information such as IP, agent, etc. Upon a logout event, the relative authentication node (created at login) will be updated. Below is an illustration from the graph database of *Aaren's* recent authentication events.



## *Access Decision Making via Policy Condition*

We have also implemented a condition plugin which queries the graph database to return policy advice based on the user's client which can be of type: *"trusted"*, *"private"*, *"public"*, or *"adversarial"*. The client type as well as the corresponding decision are determined using administrator configurable values from the OpenAM interface shown below.


AND

| | |
|---|---|
| Type | Client Type Authentication |
| Capture Window (in hours) | 2 |
| TrustedIP Min Authn History | 1 |
| PublicIP Min User Threshold | 4 |
| BlackIP Min Failure Treshold | 10 |
| PrivateIP Authn to Service | PrivateChain |
| PublicIP Authn to Service | TwoFactorChain |
| BlackIP Authn to Service | |

## Potential

Access control policies are naturally seen as graphs that connect users, resources, and conditions through permission decisions. Graph databases therefore provide a unique opportunity for expressing and evaluating access policies. There are two obvious advantages to using graph. Firstly, graph databases excel at querying connected or adjacent data from known starting points. In the access management realm the starting point and the resource being requested is also known. Graph theory allows the intermediary relationships (in this case policy) to be traversed and evaluated quickly even in complex scenarios. Graph interfaces provide a unique way of viewing data; they allow for a very high level of visual inspection. Given the appropriate graphical interface, this will help security administrators better comprehend access control policies, check consistency in policy updates, and avoid possible conflicts.

The second significant advantage of graph databases in access management is their effectiveness in recording and retrieving users' access data. The advent of the Internet of Things (IoT) has given users the capability to access resources from a multitude of different devices and locations. The variety of connecting devices and the sheer number of resources create a broad target upon which adversarial parties can attempt exploits. Increasingly complex security mechanisms and access control policies to ensure appropriate access to resources will be required. These rigorous policies will be based on a multitude of factors and will rely on intricate relationships to make sense of the policies. Multiple identities, multiple devices and multiple contexts lead to a mass proliferation of relationships that need to be managed and evaluated. This is the domain of native graph databases. One interesting opportunity that graphs provide is in the realm of adaptive access. By utilizing a graph database to track and evaluate authentication attempts, real time decisions can be made very effectively about the context of subsequent attempts.

We have presented the "client-base access control" solution as a use case which shows the effectiveness of graph databases for implementing complex access control policies in OpenAM. This is to be viewed as a starting point with various interesting directions. The same post authentication data, for instance, can be used to enforce policies which combine users' authentication times and locations to capture suspicious behaviors, e.g., when a user switches between locations which are geographically far apart or when access times do not follow the normal pattern for the provided location.

There are other AM areas which can rely on graph database storage of authentication and/or authorization data. The topics of location-based access control, DoS and dictionary attack defense mechanisms, and auditing are only a few examples of such potential.

## Interesting! Want to know more?

**Drop by the Nulli Booth.**
**Attend ForgeRock and the Graph, A Match Made in IRM; Dave Bennett; Thursday 3:30 pm.**