HAAGA-HELIA
University of Applied Sciences

Integrating an Open Source Identity Management System into Access Management Software

Samu Viitanen

Bachelor's Thesis
Degree Programme in
Information and
Communications technology
2015

HAAGA-HELIA
University of Applied Sciences

Abstract

4 May 2015

Author(s)
Samu Viitanen

Degree programme
Information and Communications Technology

Report/thesis title
Integrating an Open Source Identity Management System into Access Management Software

Number of pages and appendix pages
32 + 2

Identity management is an important aspect of enterprise data security. Companies use identity management to protect their digital property from internal and external threats. Identity management is comprised of different elements like password synchronization, directory integration, identity lifecycle management and device on-boarding.

Trusteq Oy, a data security consulting business, started a project in the spring of 2015 to study and integrate one of the available open source identity management systems into their access management software. Trusteq wished to see if open source identity management solutions were advanced enough for their needs. Their study came into conclusion that Evolveum midPoint displayed enough potential to be integrated.

This thesis describes the process of creating a proof of concept implementation of integrating the Evolveum midPoint identity management system. MidPoint was integrated next to an access management application. Customization was required due to the data consistency and –security demands of Trusteq. The default identity provisioning behaviour was changed to make midPoint master of data.

This thesis concludes that an open source identity management system can offer the required amount of features and customizability for a company like Trusteq. As a result of this thesis Trusteq received a proof of concept identity management system that will later be developed further.

Keywords
identity provisioning, identity management, open source, integration, access management

HAAGA-HELIA
University of Applied Sciences

Tiivistelmä

4.5.2015

Tekijä(t)
Samu Viitanen

Koulutusohjelma
Tietojenkäsittelyn koulutusohjelma

Opinnäytetyön otsikko
Avoimen lähdekoodin identiteetinhallintajärjestelmän yhdistäminen pääsynhallintasovellukseen

Sivu- ja liitesivumäärä
32 + 2

Opinnäytetyön otsikko englanniksi
Integrating an Open Source Identity Management System into Access Management Software

Identiteetinhallinta on tärkeä osa yritysten tietoturvaa. Yritykset käyttävät identiteetinhallintaa suojellakseen digitaalista omaisuuttaan ulkoisilta ja sisäisiltä uhilta. Identiteetinhallinta koostuu erilaisista elementeistä kuten hakemistojen yhdistäminen, salasanojen ja identiteettien elinkaarten hallinta sekä laitteiden on/off-boarding.

Tietoturva-alan konsultointiyritys Trusteq Oy aloitti keväällä 2015 projektin jonka tavoite on yhdistää heidän nykyinen pääsynhallintasovelluksensa yhteen saatavilla olevista avoimen lähdekoodin identiteetinhallintajärjestelmistä. Trusteq halusi nähdä tarjoavatko avoimen lähdekoodin identiteetinhallintaratkaisut riittävän edistyneitä ominaisuuksia heidän tarpeilleen. Trusteq tuli siihen tulokseen että Evolveum midPoint osoitti riittävästi potentiaalia integroimiseen.

Tämä opinnäytetyö kuvaa koetoteutuksen Evolveum midPoint identiteetinhallintajärjestelmän yhdistämisestä pääsynhallintasovellukseen. Trusteq:n antamat tietoturva- ja tietojen eheysvaatimukset johtivat monenlaisiin muokkauksiin sekä pääsynhallintasovelluksessa, että identiteetinhallintajärjestelmässä. Identiteettien provisiointiprosessin oletuskäyttäytymistä oli muutettava niin, että midPoint pitää yllä tietojen oikeellisuutta.

Opinnäytetyössä tullaan siihen tulokseen että avoimen lähdekoodin identiteetinhallintajärjestelmä voi tarjota riittävästi ominaisuuksia ja muokkausmahdollisuuksia Trusteq Oy:n kaltaiselle yritykselle. Tämän opinnäytetyön tuloksena Trusteq sai prototyypin automatisoidusta identiteetinhallintajärjestelmästä, jonka jatkokehitystä on jo suunniteltu.

Avainsanat
identiteettien provisiointi, identiteetinhallinta, avoin lähdekoodi, yhdistää, pääsynhallinta

Table of contents

1 Introduction

Identity Management defines what physical users can do on a network under different circumstances. It is really important for physical users on the Internet to know about their identities in different resources. A physical user can be a person, an organization, a network or even a country. Identity management is used to increase data security, user productivity and data consistency. (Jøsang & Pope 2005, 1; Rouse & Mathias 2013.)

1.1 Background

A data security consulting business called Trusteq Oy started a project in the spring of 2015 to study two of the available open source Identity Management (IDM) systems on the Internet and to integrate the an IDM system next to their Access Management (AM) software. The project team of Trusteq thinks that including identity management in their AM software could really increase their leverage against competing companies.

Identity management is a broad subject, but Trusteq was mainly interested in identity provisioning. Provisioning is an identity management operation what coordinates the creation of user accounts, email authorizations and other tasks (Sullivan, 112-113). There is only a handful of open source identity management systems with provisioning capabilities, but an open source identity management system would offer an inexpensive increase in data consistency (Sullivan, 112-113).

1.2 Project Goals and Scope

The purpose of this thesis was to study and compare two different open source IDM systems and to create a Proof-of-Concept (PoC) of integrating an IDM system next to an AM application. These systems were Evolveum midPoint and Apache Syncope. The subjects of the study were selected based on the ideas of the lead architect of Trusteq. He has previous experience and knowledge about IAM and IDM software. He thought that based on perfunctory investigation, these two systems offered enough features and customizability for Trusteq Oy.

The comparison was performed to help the Trusteq project team decide which system should be implemented. The comparison was done based on pre-determined comparison criteria. The comparison process is described in chapter 3. The results of the comparison describe why the selected IDM system was integrated. In addition to comparing the sys-

tems based on criteria, the following research questions had been prepared to help shape out the research problem of this thesis:

1. Can Trusteq improve data consistency and data security with identity provisioning?
2. Can open source identity management systems offer enough customizability and features to function next to the access management software of Trusteq Oy?

As a result of this thesis, Trusteq has a Proof-of-Concept (PoC) of an open source Identity Management System (IDMS) implementation. The PoC is basically an example implementation that is used to prove that an integration like this is technically possible and cost efficient. The implementation process is described in chapter 4. The PoC implementation will later be used for further development after this project has finished.

Trusteq Oy was mainly interested in identity provisioning. This thesis is restricted strictly to identity provisioning. Only subjects related to identity provisioning are discussed. Things like password synchronization, directory integration, device on/off-boarding and other irrelevant IDM related features are left out. The scope was limited like this to reduce the amount of indifferent information without omitting important theory.

1.3 Work Methods and Tools

The IDMS comparison was performed in a research-type manner based on a set of predetermined criteria. The criteria was gathered together by consulting the project team of Trusteq Oy and by studying an existing IDMS comparison. The consulted Trusteq employees together have over 20 years of experience in identity and access management therefore their opinions can be considered as definite information. Evolveum had also conducted an IDMS comparison in May 2015. The comparison of Evolveum was taken into account when the criteria set was constructed.

The implementation project was started using Scrum as the development method. Scrum is a way for development teams to work together. It is an efficient “thinking framework” that brings a development team together by encouraging communication. Scrum teams are self-organizing and cross-functional teams that execute a set of tasks over a Sprint. Sprint is a time frame usually spanning around 2-4 weeks. The contents of a Sprint are planned in sprint planning, where the whole Scrum team gathers together to shape out the goals to be achieved during the next sprint. Scrum really suitable and efficient development method for software development. (Scrum.org and ScrumInc 2014.)

After using Scrum for a while, Trusteq Oy decided to switch to the Kanban development method. Kanban is different from Scrum. It focuses more on goals over a time frame (Peterson 2009). In Kanban, software development could be described as a pipeline where feature requests come in and improved software comes out (Peterson 2009). Kanban work method is simple. A big physical or digital board is populated with “sticky notes” to describe what features are being worked on and what are finished. The board is divided to different parts of development. These sections have been limited to a certain amount of sticky notes by a number above the section. (Peterson 2009.) Figure 1 explains how a Kanban board can be organized.

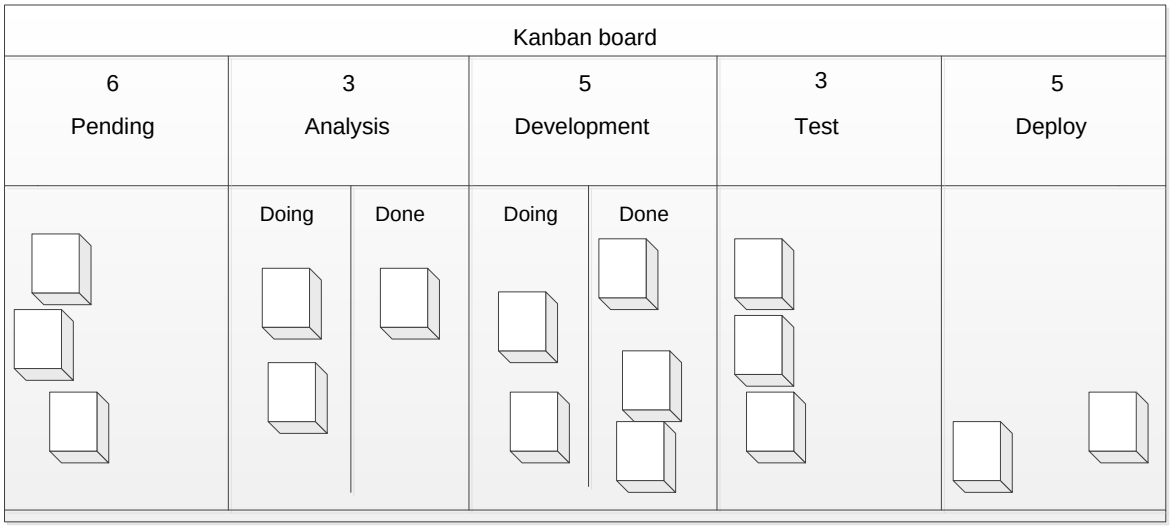

Figure 1. Example Kanban board (Peterson 2009)

The testing part in the example has been limited to 3 sticky notes. When testers are done with testing a certain feature, they move the note and free up a slot to test something else. Kanban is more of an idea or a guideline that can be modified. It is up to developers to make it reasonable. (Peterson 2009.)

2 Identity Management

The core of this thesis is identity provisioning but provisioning is part of a bigger concept; identity management. The following chapter helps readers gain more theoretical and functional understanding about identity management in the scope of this thesis.

2.1 Identities in Cyberspace

A digital identity is a unique set of information that identifies a user in an information system. Identities are always unique per system. The system cannot contain two instances of the same identity. An identity must contain an attribute or a combination of attributes that can be used to identify the user. (Abelson, Lessig, Covell, Hochberger, Kovacs, Krikorian & Schneck 1998.) Figure 2 describes an example identity structure in a network.

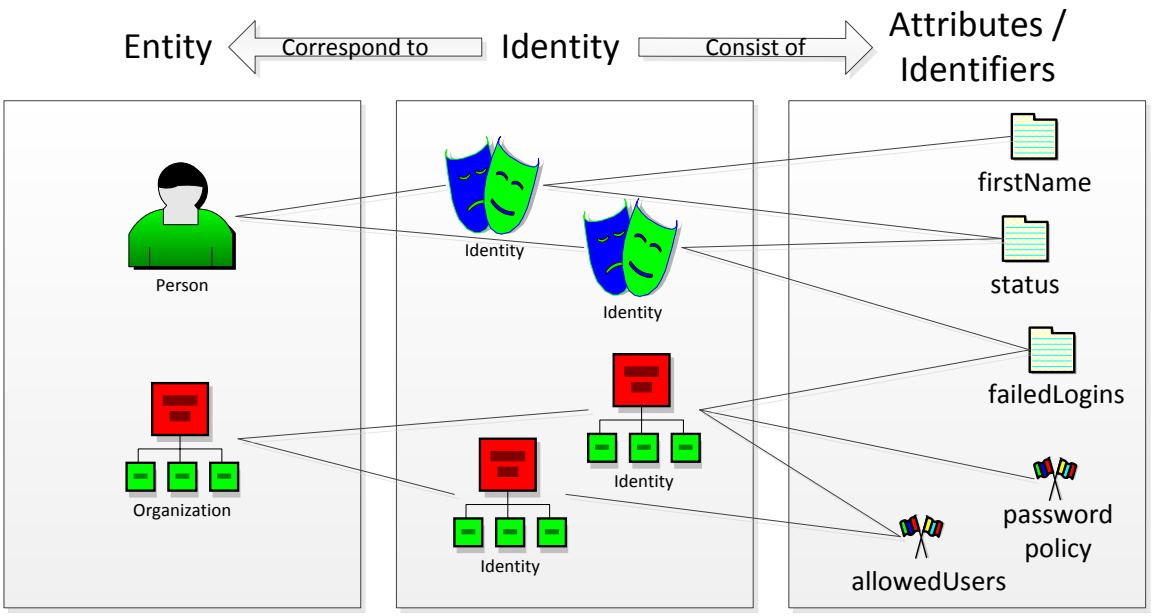

Figure 2. Identities in an example network (Azstrel Tech-Soft Pvt Ltd. 2013)

An identity always corresponds to a person or another entity like an organization or a group. Identities always consist of attributes that may or may not be unique. There must be a way to make sure that an identity can be linked to the user it belongs to. Before a person can use an identity in a network, they have to be authenticated against the identity. The simplest way of authentication is a password, but it is not very reliable since passwords can easily be shared or distributed. (Azstrel Tech-Soft Pvt Ltd. 2013; Abelson, etc. 1998.)

2.2 Identity Management

In enterprise setting, identity management is about managing the roles and access privileges of individual network users. Companies use identity management to protect their digital assets. Well managed identities translate directly in to smaller risk of external or internal attacks. A typical identity management system contains the following elements:

- identity repository that is used to store the identities of individual users,
- identity lifecycle management that manages the creation, modification and deletion of the identity data,
- user access regulation with access management,
- auditing and reporting system to verify what happens within the system. (Waters 2004.)

User access is regulated with different authentication methods like passwords, digital certificates, tokens and smart cards. An efficient authentication method is two-factor authentication where something you know, like a password, is combined with something you have, like a hardware token or a one-time password in a mobile device. An example of a two-factor authentication is a Wireless Public Key Infrastructure (WPKI) authentication where the authentication server prompts the user for a pin code through their mobile phone using the cellular network. Two-factor authentication offers a secure way of identifying the user. (Waters 2004; RaulWalter 2015.)

Identity management allows companies to extend access to their information beyond the company staff without compromising security. However, centralizing operations into one place may offer tempting targets to hackers and crackers. If a hacker gained access to the centre of the operation structure, he or she could create identities with powerful privileges in other systems. (Waters 2004.)

Identity management consists of many features; password management, security policy applications, reporting and monitoring apps and identity repositories. Currently a big thing in identity management is “identity lifecycle management” which encapsulates the concepts of identity provisioning, de-provisioning and managing and synchronizing digital identities. (Waters 2004.)

2.3 Identity Provisioning

Identity provisioning aims to make sure that business applications and services are protected against unauthorized access and the data across the network is consistent. Provisioning can operate manually or automatically, usually the latter. Manual identity provi-

sioning often relies on human interaction. Automatic provisioning is centralized and uses prepared workflows or configuration scripts to run. (Casassa Mont, Baldwin & Shiu 2009.)

Identity provisioning requires dealing with two core phases. When users are created, modified or deleted, it is the approval phase. The approval is usually completed by a person, usually a manager or a supervisor. An example of the approval phase is when an administrator creates a new user in an application. After the user has been created, it has to be reflected into target resources connected with the application. This phase is called the deployment and configuration phase. In the deployment and configuration phase, the newly created user will be provisioned in multiple resources across the resource network. Deployment and configuration phase can be manual or automatic. (Casassa, etc. 2009.)

Identity provisioning makes creation, management and deactivation of login identifiers, directories, mail folders, security entitlements and other related items cheaper and reliable. Provisioning can automatically aggregate and correlate data from Human Resources (HR), Customer Relationship Management (CRM), email systems and other identity repositories. Provisioning also reduces the complexity of administration by automating operations that normally would have to be performed by a person (Hitachi ID Systems, Inc. 2015; Gartner, Inc. and/or its Affiliates. 2013.)

Many organizations choose to manage identities manually instead of automating the process. Automating the identity management process reduces the cost of manual work to manage identities, cost of access right management, security related costs, audit costs etc. It also reduces helpdesk overload and reduces the amount of unused user accounts. Many application and resource licences are user-based so reducing the amount of users saves money. (Semancik 2014b.) Figure 3 describes the current state of many identity management systems.

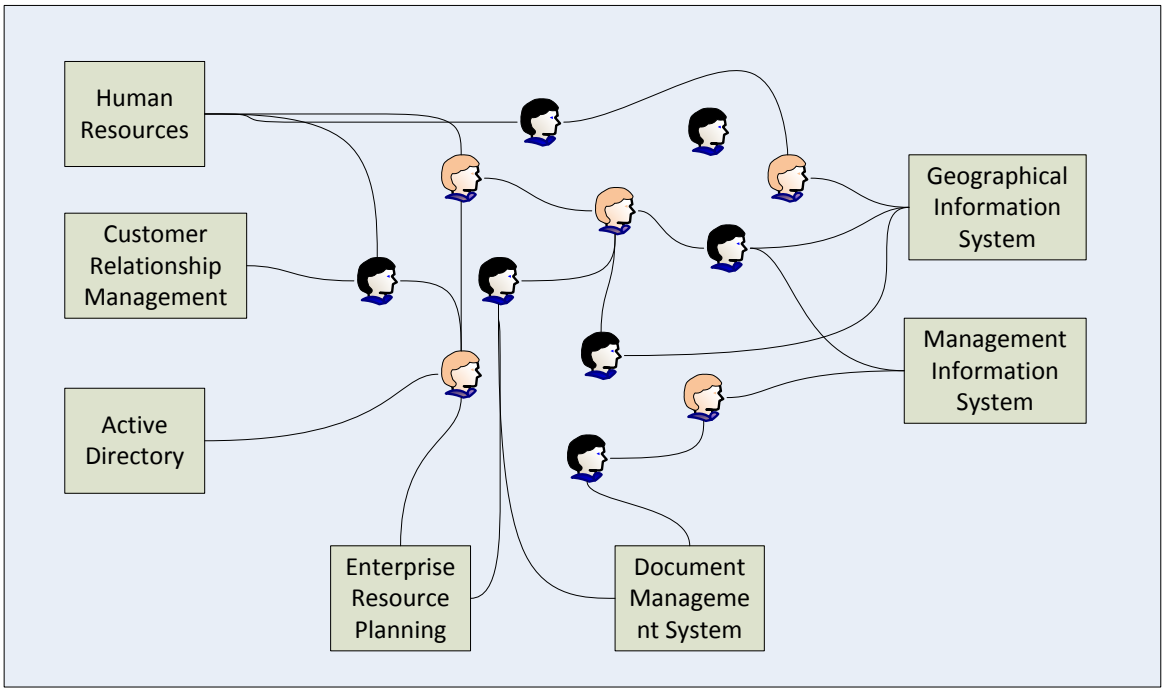

Figure 3. Current state identity management systems in many organizations (Semancik 2014b)

In a manual IDM system like in figure 3, people are contacting each other for information change requests and reassignments, accessing different resources through different people and so on. The process is very slow, unreliable and slightly confusing. The processes are likely to cause problems and costs within the organization. Majority of these processes and actions are routine, which means they can be automated. Computers are designed to do these things faster and more efficiently. Figure 4 describes how an identity provisioning system affects the structure of these processes. (Semancik 2014b.)

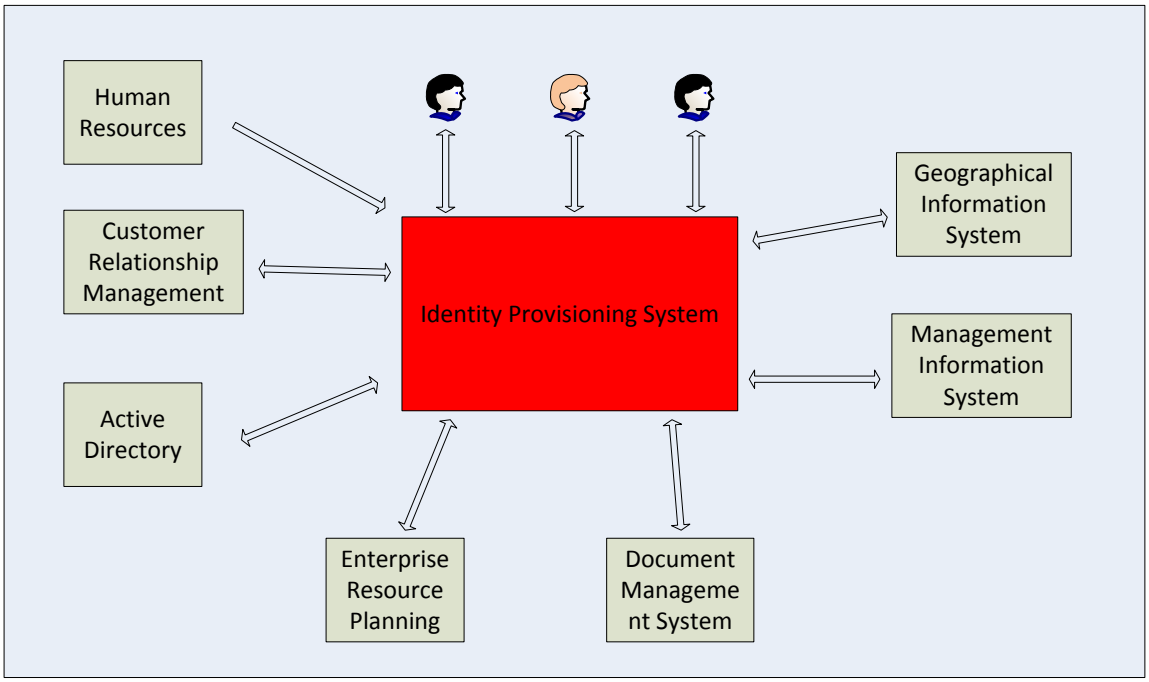

Figure 4. How an identity provisioning system affects the identity management processes (Semancik 2014b.)

In Figure 4, the identity management system automates most of the processes involved. However, some of the operations like approval and initial user creation still have to be performed by a person. Human interaction cannot be avoided, but automating the provisioning process will make the work of people much more efficient. (Semancik 2014b.)

3 Comparing Provisioning Software

Before integration, Trusteq wanted to compare two similar open source identity management solutions: Apache Syncope and Evolveum midPoint. They are both products that have been derived from the original OpenIDM system which is currently managed by ForgeRock. According to the lead architect of Trusteq the development of the original OpenIDM system is not going the way Trusteq desires. ForgeRock has changed the OpenIDM license and they also seem to have reworked some major features and functionalities from the original OpenIDM code. It would be safer and easier for Trusteq to work on Syncope or midPoint.

Before midPoint and Syncope were compared, a set of comparison criteria had to be constructed. The criteria explains what kind of features and qualities Trusteq needed in an identity management system. A similar comparison was also conducted by Evolveum in May 2015. The comparison criteria in that study was taken into account in addition to the criteria constructed from the requirements of the employees of Trusteq Oy. The criteria has been divided into chapters where the two different solutions are compared based on the criteria.

3.1 Identity Connectors

One important criteria was that the selected identity management system has to support various identity connectors. Identity connectors are the heart of automated identity management systems. They act as a bridge between the system and the connected identity resources. Identity resources store identity information in different ways. An identity connector handles the transformation of identity data between the resources. (Oracle 2015.)

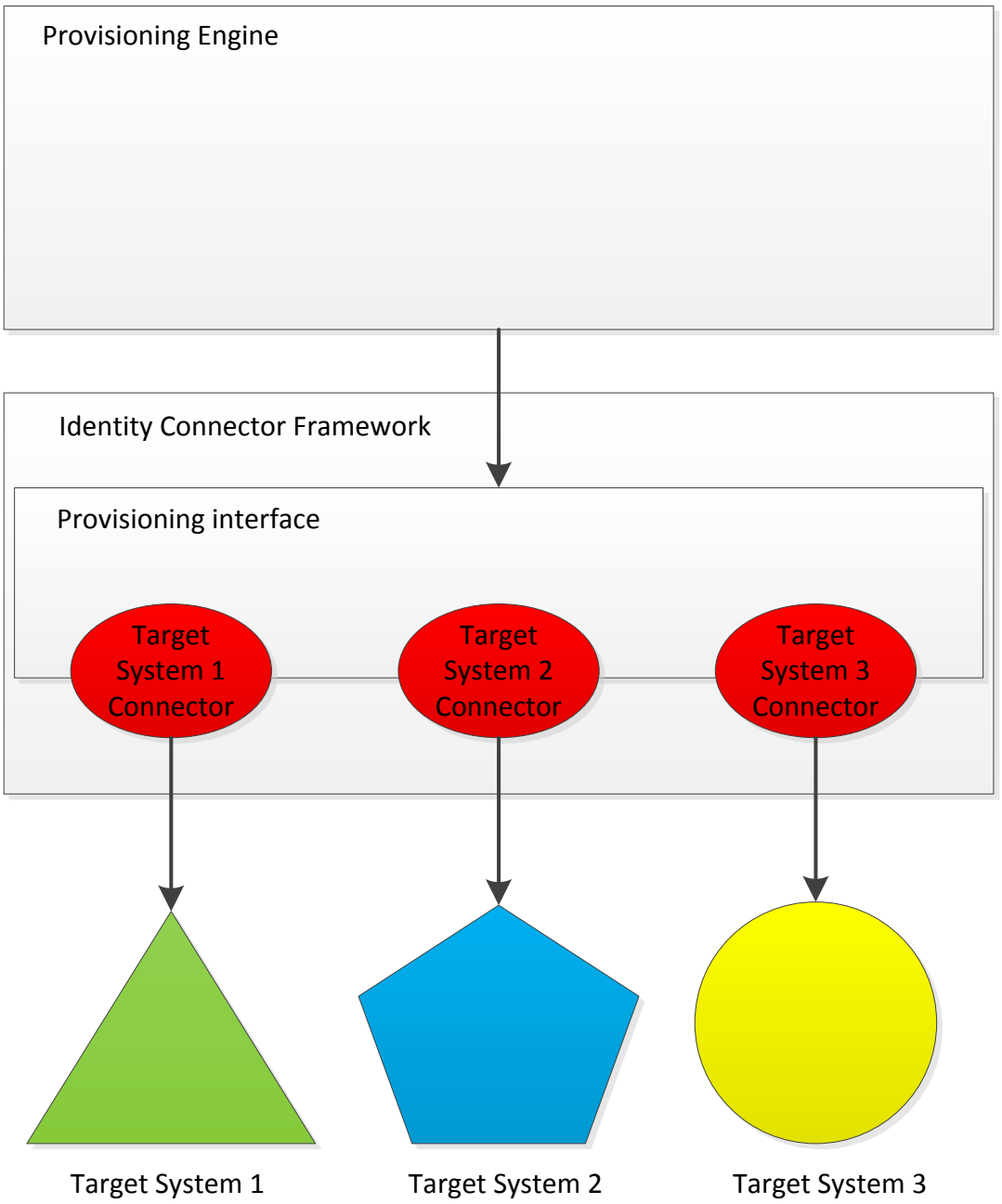

Figure 5. Identity connectors in provisioning (Oracle 2015)

Figure 5 describes how the provisioning engine gains access to different resources through the identity connectors in the Identity Connector Framework (ICF). The identity connectors are designed to act as modules that can be removed or added as pleased. Most identity connector frameworks also allow developers to make their own identity connectors. (Oracle 2012.)

Trusteq is likely to implement various connectors in their IDM system but initially Trusteq only required support for the Microsoft Azure Active Directory (AD) connector. AD is similar to any other directory service but a normal Lightweight Directory Access Protocol

(LDAP) connector is not sufficient enough for AD. AD requires a connector that is customized just for Active Directory.

In addition to the AD connector, Trusteq required a possibility to make a customized identity connector that allows the IDMS to efficiently communicate with a complicated Hibernate database. A normal database identity connector was not sufficient enough because the requirements stated that the connector should do exactly what the AM application of Trusteq did in terms of data persistence.

Syncope uses the Connectors for Identity Management (ConnId) framework. ConnId is an open source ICF managed by the Apache Syncope team. Evolveum and the ForgeRock development teams are also contributing in the ConnId project. ConnId was recently merged with the OpenICF. The two different frameworks offered essentially same features and interfaces so the development teams decided to join them to avoid confusion. (E-mail discussion 2013; Chicchiriccò 2015a.)

It is not perfectly clear what identity connectors are currently supported by the recent version of the ConnId framework, but Syncope documentation claims that Syncope supports at least the connectors made with the original ConnId framework. Syncope has stable releases for the following ConnId identity connectors:

- Database connector
- Active Directory (AD) connector
- Command Line (CMD) connector
- Comma Seperated Values (CSV) connector
- Lightweight Data Access Protocol (LDAP) connector
- Simple Object Access Protocol (SOAP) connector

(SOURCE.)

The list of available identity connectors offered by Apache Syncope is not very large, but it is safe to say that Syncope offers enough supported connectors for the initial requirements of Trusteq.

Evolveum midPoint also uses the ConnId framework. Evolveum has started a project called Polygon which is an effort to join all currently available ConnId and OpenICF connectors in one place by modifying, documenting and testing them. Due to the Polygon project, midPoint theoretically supports all available identity connectors. MidPoint has stable releases of the following connectors:

- DatabaseTable Connector (recommended version: 1.4.0.49 - Modified by Evolveum)

- CSVFile Connector (recommended version: 1.4.0.49 - Modified by Evolveum)
- LDAP Connector (recommended version: 1.4.0.49 - Modified by Evolveum)
- Active Directory Connector (recommended version: 1.4.1.20257 - Evolveum fixes)
- ScriptedSQL Connector (recommended version: 1.1.2.0.em3
  (Semancik 2015a.)

The identity connectors described before are only the stable releases of connectors. They have been fully tested and documented by Evolveum. MidPoint also has different levels of support for other available ConnId and OpenICF connectors. For a list of other connectors, see Appendix 2.

Syncope and midPoint both use ConnId as the identity connector framework. ConnId has a comprehensive API for custom made identity connectors. The ConnId API includes Service Provider Interface (SPI) operations that can be implemented into the identity connector as preferred. The most relevant SPI operations are: authenticate, create, delete, search, sync, test and update.

3.2 Customizability

Another important criteria for Trusteq was that the chosen identity management system has to be customizable enough to change the default provisioning behaviour. The system has to offer a way to invoke the internal functions of the system from a third party application. It has to be possible to create new users remotely. Apache Syncope offers third party applications access by a Representational State Transfer (REST) interface. REST is a simple alternative to typical SOAP and Web Service Description Language (WSDL) based services. (SMARTBEAR; Apache Syncope 2012, 8.)

MidPoint has multiple ways to access the internal functions of the system. MidPoint offers third party applications access on different levels. The highest level of access is offered by their IDM Model Interface, which can be implemented as a client Java application. The IDM Model interface offers essentially all the same functions that are used within midPoint itself. The second highest level of accessibility is offered by the IDM Model Web Service Interface that can be invoked using SOAP envelopes. The web service interface offers almost all the functions the IDM Model interface does but it can be accessed remotely without a Java client. The last and the lowest level of access is offered by the midPoint REST Application Programming Interface (API), which has limited amount of usable functions. (Mederly 2014; Semancik 2015b; Semancik 2015c.)

3.3 Configuration and Deployment

Trusteq does not intend to use the Graphical User Interface (GUI) of the chosen IDM system. The system has to offer tools to configure and deploy from the command line. In a perfect world, necessary customizations and configurations could all be included in one script that would install and deploy the IdM solution.

The Syncope administration console uses the same RESTful interface that offers access to third party applications. It means that everything that can be done from the administration console, can also be done remotely. The RESTful interface should theoretically allow Trusteq to make necessary configurations and deploy the system without accessing the GUI. (Apache Syncope 2012, 8.)

MidPoint offers similar solutions for remote access. Either the midPoint REST API or the IDM Model Web Service can be used for necessary command line invocations.

3.4 Scalability

Trusteq required that their identity management system should be able to sustain at least 200 000 users in the system. This is a baseline requirement criteria that is likely to change in the future. The vast amount of users creates a need for connection pooling. Connection pools are groups of usable database connections. They are often managed in the memory of the database and can be reused. Connection pools are used to increase the performance of executing database commands. (IBM Corp. 2013, 593.)

In addition to connection pooling Trusteq desired the possibility to divide the workload between multiple instances of the IDM system. This process is called server clustering. Clustering in this context means that multiple servers are connected to act as one. If one cluster node stops functioning, a failover process moves all the unfinished processes and workloads from the broken node to a running server. This allows the service to be always available, increasing consistency and security. (Microsoft 2003.)

Since both Syncope and midPoint use ConnId as their identity connector framework, they both support connection pooling. ConnId has an interface called PoolableConnector which makes an identity connector eligible to be pooled. Pooled identity connectors are similar to pooled database connections where multiple instances of a connector are cached and kept alive for different ICF operations. (ForgeRock AS 2010.)

Syncope and midPoint are both eligible for clustering. Syncope clustering depends heavily on the Java Enterprise Edition (JEE) container used. Syncope documentation only includes a guide about setting up Syncope clustering in the Tomcat 7 container.

MidPoint also recommends a standard Tomcat container if the service is to be clustered. The instances of the midPoint cluster are called nodes. Theoretically there are no limits in how many nodes can operate in one cluster, but a large cluster has never been tested by Evolveum and experimentations are encouraged. (Chicchiriccò 2012; Mederly 2013.)

3.5 Documentation and Reliability

Documentation is a very important criteria for Trusteq. It is crucial that any software developer can look at the documentation of the selected IDM system and understand the architecture and functions. Documentation is often overlooked because it does not provide direct benefit for the writer and it also requires writing skills to write informative documentation (Yeates 2005).

Apache Syncope documentation is confusing. They have a confluence Wiki page that has some information, but not nearly enough. They also have three different websites to search information from. Well organized documentation offers a certain amount of reliability and certainty that the system is actually going to work.

On the other hand, midPoint has more comprehensive documentation. Their confluence page has almost all information required for the implementation of midPoint. Some deficiencies and unfinished articles also exist, but the midPoint documentation looks truly circumambient compared to Syncope.

Also worth mentioning is that Evolveum has a professional attitude towards developing midPoint. They focus on efficient deployment and functionality instead of esthetics. They want to make sure that midPoint works perfectly before they start working on appearances and cosmetic features. One of the mentalities of Evolveum is that focusing on donations and subscriptions instead of commercializing the product is the way to go. They think that an ecosystem of smaller flexible companies offer more than one giant company. They have absolutely no interest in competing with their partners. (Evolveum s.r.o 2014a; Evolveum s.r.o. 2014b; Evolveum s.r.o. 2014c.)

3.6 Data Consistency

Data consistency for Trusteq means that all data is as consistent as possible across the whole service network. Full transactional consistency is an unrealistic goal in identity management systems, but the system that offers better features for data consistency will be taken into account.

Syncope does not have many mentions of data consistency in their documentation. According to Semancik (2015d) “Yet another issue is a provisioning consistency. Similarly to Sun IDM the Syncope seems to have just a basic retry mechanism to handle provisioning failures. This is likely to become a major issue in deployments with large number of resources.”

Many traditional identity management systems map data from one resource to another, which is fine for simple identity management systems. However, even medium sized enterprises can cover tens of different systems. Mapping information from system to system will result in an uncontrollable pile of data transformation rules. What midPoint is trying to create is a common data model where all identity data is mapped to a midPoint common data model before mapping it to another system. (Semancik 2012.) Figure 6 below describes the difference.

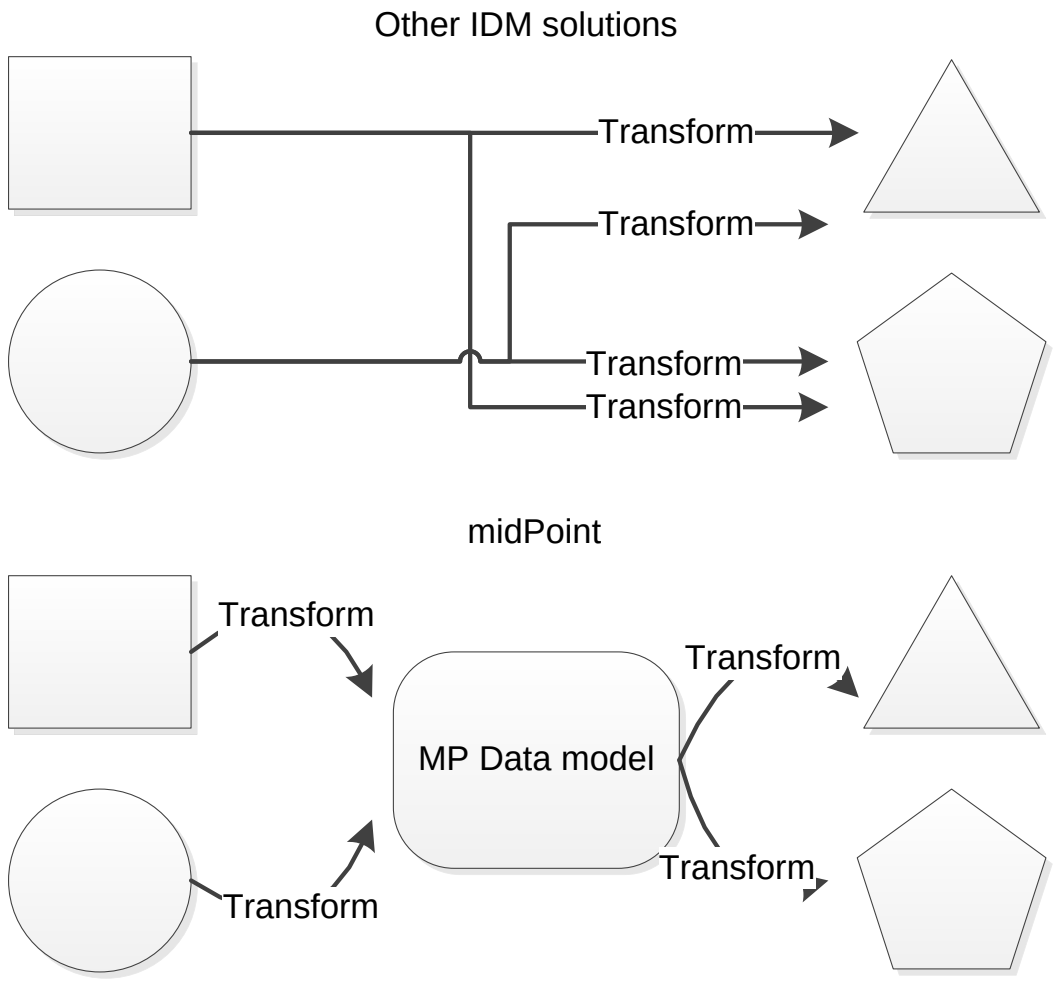

Figure 6. MidPoint data mapping compared to other solutions (Semancik 2012)

The midPoint common data model basically eliminates the need to write transformation rules from system to system. All data is transformed into a midPoint model object which keeps the amount of transformation rules relatively small. (Semancik 2012.)

In addition to the common data model, midPoint uses relative changes instead of absolute changes. Many traditional IDM systems replace the whole user if it has changed, whereas midPoint only replaces the changed attributes within the user. Relative changes significantly reduce the need for resource locking. Instead of locking the whole user when updating, only the changed attributes of the user are locked for the time being. (Mederly 2011.)

MidPoint follows a weak consistency model (LDAP consistency model). It cannot guarantee that all data is consistent across the network. This makes midPoint non-transactional. However, midPoint has a built in consistency system that is constantly checking for conflicts. If a conflict is found, midPoint will read its configuration to decide what to do. A well

configured instance of midPoint will not crash if something fatal happens, it will just try to ignore the problem and log it. (Mederly 2011.)

3.7 Other mentions

One important feature that Trusteq desires is the support for multi-tenancy. Multi-tenancy is an architecture where one instance of a software application serves multiple customers. These customers are called tenants. The tenants could possess privileges to change certain elements in the application but not the actual code. A multi-tenant application offers better maintenance and cost efficiency compared to single-tenant applications. Multi-tenant application provider only has to update the application once, whereas single-tenant provider has to update every single application. (Rouse 2014.)

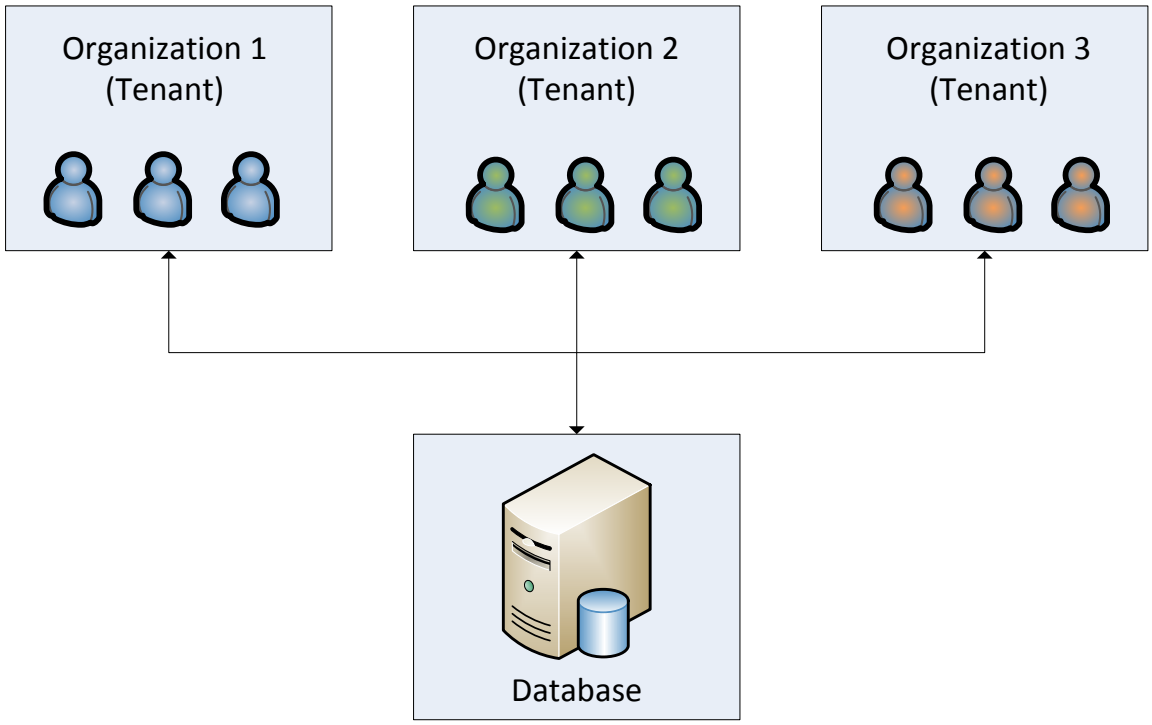

Figure 7. Fine-grained multi-tenancy (Shalom 2010)

Figure 7 depicts fine-grained multi-tenancy as a concept. Fine-grained multi-tenancy means that essentially all users in different tenants share the same database. This means that there has to be a way to identify the user in the tenant itself in addition to identifying them across the network. (Shalom 2010.)

Apache Syncope supports multi-tenancy in a form of “realms” where a realm could basically be an organization that may even have sub organizations. The concept of realms is a new feature that was introduced in Q1 of 2015. It was released after this project was started. (Chicchiriccò 2015b.)

Evolveum midPoint has full support for multi-tenancy. Tenants in midPoint are called “Organizational Units”. The organizational tree structure in midPoint offers flexible tools to create simple or very complex organizational structures. (Semancik 2014c.)

3.8 Summary

Based on the data gathered before, a summary and a conclusion has been constructed. The summary is described in Table 1 below.

Table 1. Comparison summary

	Apache Syncope	*Evolveum MidPoint*
Identity Connectors	has an AD connector and supports customized connectors	has an AD connector and supports customized connectors
Customizability	has one way for remote access	has multiple ways for remote access
Configuration and Deployment	has one way to invoke functions remotely	has multiple ways to invoke functions remotely
Scalability	supports connection pooling and clustering	supports connection pooling and clustering
Documentation and Reliability	messy and confusing documentation, not many mentions about development agenda	relatively comprehensive, a lot of information about development methods and agenda
Data Consistency	basic failover retry mechanism	common IDM data model, relative changes, consistency mechanism
Multi-tenancy	new feature to support multi-tenancy in the form of “realms”	full support for multi-tenancy

Table 1 was presented to the development steering group of Trusteq. After a brief discussion about the results of the study, it was decided that midPoint was the superior system considering the requirements of Trusteq. Both systems met the requirements, but midPoint displayed more potential due to comprehensive documentation, special data consistency features, multiple remote access points and the extensive support for various identity connectors and multi-tenancy.

4 Integration

After midPoint was chosen as the system to be implemented, the following business case was constructed: “MidPoint must persist a specific set of user attributes into the database of the Access Management (AM) and into Active Directory. AM must persist the rest of the data without corrupting the data that originated from midPoint.” The business case was then divided into different tasks that are displayed in Table 2 below.

Table 2. midPoint implementation tasks

MidPoint must persist a specific set of user attributes into the database of the Access Management (AM) and into Active Directory. AM must persist the rest of the data without corrupting the data that originated from midPoint.	
TASK-1	AM must propagate user information immediately into midPoint when user is saved and tracked information is changed in the AM application.
TASK-2	Midpoint must synchronize all available information with AD.
TASK-3	Midpoint must persist information into the AM database.
TASK-4	After information has been persisted into the database from midPoint, AM application must poll its database for changes in update dates or versioning.
TASK-5	When AM knows about the completed propagation, it must merge information that originated from midPoint with the rest of the user information and persist the complete data into the database of the AM application and update its session user.

The following chapters describe the integration process in the scope of this thesis. It has to be noted that the resulting implementation is a proof of concept, so all of the specified criteria might not be relevant at this point. The implementation is explained with the research problem in mind.

4.1 Installing Identity Connectors

A standard AD connector was installed into midPoint to allow synchronization from AD. Installing and configuring the AD connector was enough to complete TASK-2. In addition to the AD connector, a custom Hibernate database connector was created to complete TASK-3. The connector allowed midPoint to persist data into a database like the access management application. The custom connector reduces the amount of customization required in the existing AM application. It allows the use of persistence functionalities similar to the AM application thus removing the need to modify the database.

The Hibernate connector was written using the ConnId framework. All of the ConnId SPI operations were not necessary. Trusteq only intended to create or update users in the database and midPoint would get all up-to-date information from somewhere else beforehand. That way the custom connector would not have to pull changes from the complex database since it already had the most recent data. Custom operation like that dramatically increases performance since no unnecessary database queries need to be performed.

MidPoint initializes identity connectors based on a resource reference. The resource reference contains a set of configuration properties. The identity connector has to override a method called getConfiguration to allow configuration properties to be specified in midPoint. MidPoint might need reference to multiple instances of the same database on different servers for example. Table 3 describes the configuration properties in the custom Hibernate identity connector.

Table 3. Hibernate connector configuration

Configuration property name	Description
username	The username for the database connection
password	The password for the database connection
jdbcUrl	The Java Database Connectivity (JDBC) URL for the database connection
driverClass	The JDBC driver class that is used with the database connection (org.postgresql.Driver for example)
hibernateDialect	Hibernate can be set to optimize Structured Query Language (SQL) queries for a specific database
pool_size	The size of the connection pool
changeLogColumn	The database column name that is used to store update dates of identity data

An important part of the custom connector property was a change log column. The change log column is basically the name of a database column that stores the last update time of the identity, preferably even few previous update times. Trusteq already had a database column that logs identity update times which was more than enough for logging change times. However, implementing the custom connector alone was not enough. Since the connector only uses create and update SPI operations, there is no way midPoint would ever retrieve the user data from the database. The solution to this issue is explained in the next chapter.

4.2 Customizing the Provisioning Process

Trusteq decided not to synchronize identity data from the database, so they needed to modify the default provisioning process. Figure 8 describes the normal provisioning process in IDM systems.

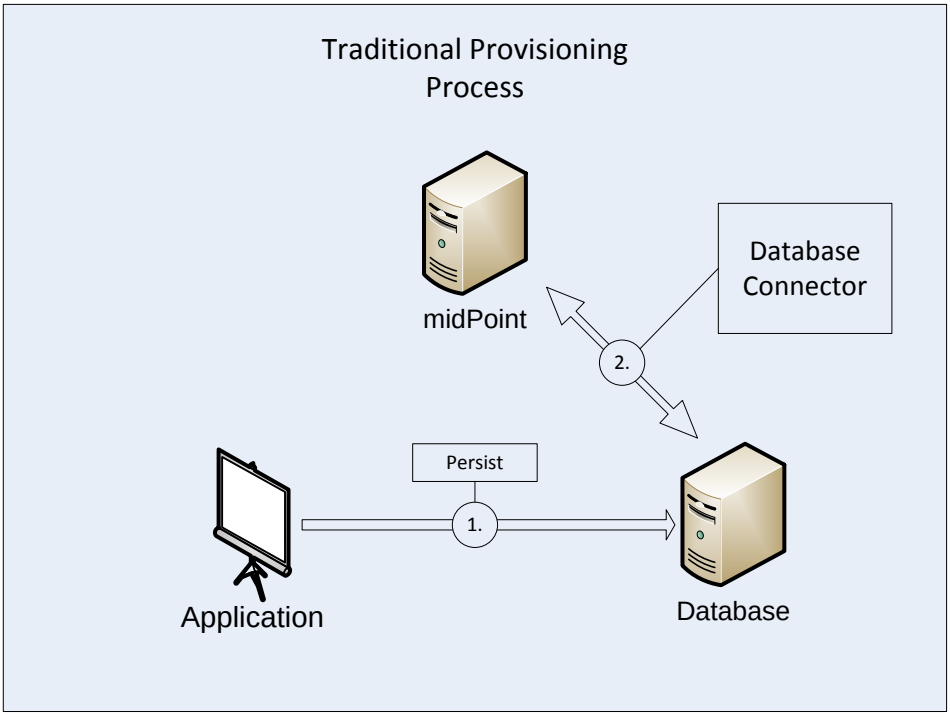

Figure 8. Traditional provisioning process (Semancik 2014b)

In the model displayed in Figure 8, the application persists user information directly into the database, leaving the rest to midPoint. However, TASK-1 required that the identity data would go to midPoint before the database to guarantee that midPoint has the latest data before it is reflected anywhere else. Figure 9 explains how Trusteq wanted to change the synchronization behaviour.

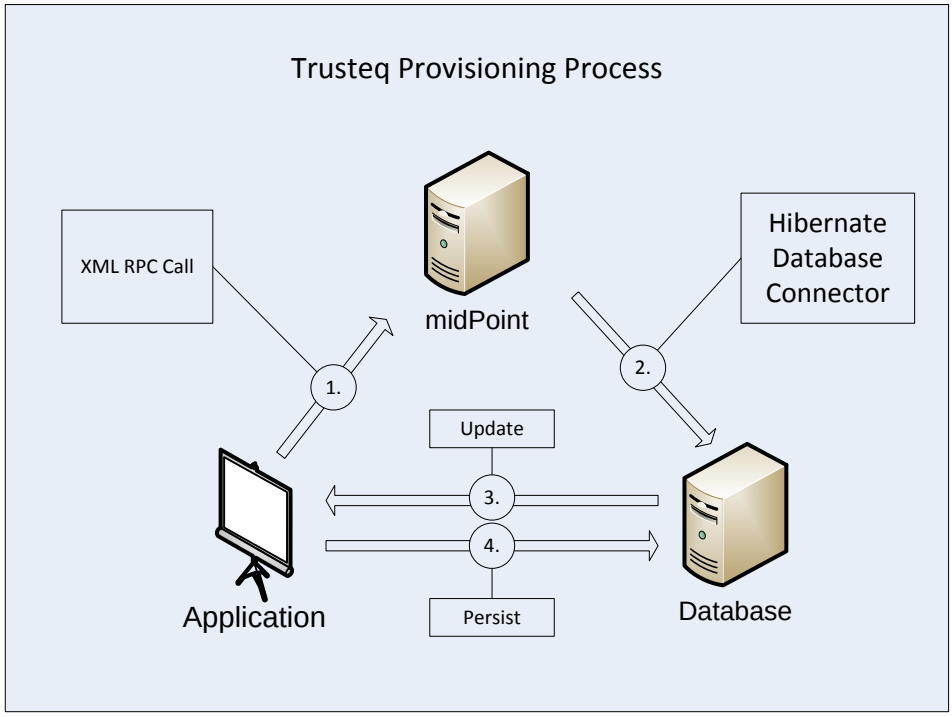

Figure 9. Customized provisioning process

The AM application sends a Remote Procedure Call (RPC) to midPoint. The RPC contains an Extensible Markup Language (XML) representation of the user data that will be stored by midPoint. MidPoint has an interface called IDM Model Web Service interface. Developers can invoke almost all midPoint operations through the interface by sending XML RPC messages to midPoint via Hypertext Transfer Protocol (HTTP) or HTTP Secure (HTTPs). (Bray, Paoli, Sperberg-McQueen, Maler, Yergeau 2008; Semancik 2014a.)

An XML RPC message is a HTTP POST request that uses XML as its body. (Winer 1999). In this context, the XML RPC messages are SOAP envelopes that contain a security header and an XML representation of the operation to be performed in midPoint. See Appendix 1 for an example SOAP envelope. The SOAP envelope allows developers to create new users, organizations or groups in midPoint from basically anywhere.

The RPC messages are sent to an endpoint URL. The URL has been specified in midPoint documentation. Any data that is sent to the endpoint URL is passed to the IDM Model Web Service interface. The web service interprets the received information and performs actions accordingly. The actions that are performed by the web service are described in the web service description. Web service descriptions are written in Web Service Description Language (WSDL) (Christensen, Curbera, Meredith, Weerawarana 2001).

The web service description is basically an XML schema that defines the functions of the service (Christensen, etc. 2001). The web service description can be downloaded from the endpoint URL. Like any XML schema, the web service description can also be unmarshalled into Java objects with something like Java Architecture for XML Binding (JAXB). JAXB is an architecture that helps developers create Java classes from XML documents.

Trusteq used a tool called wsimport in the Java Standard Edition (SE) to import the midPoint WSDL file. Wsimport uses Java API for XML Web Services (JAX-WS) to create Java classes from the schema types in a web service description (Oracle 2011). The import operation was included in a build file to allow importing of Java classes at runtime instead of importing the entire web service into version management. Importing the web service description at runtime is efficient because the midPoint web service description contains over 800 different entities which are converted into Java classes.

The web service description could not be imported dynamically from an URL because midPoint generates the web service description namespaces on the fly. The namespaces in the description do not work if they are imported from the midPoint instance so they had to be downloaded from midPoint version management.

4.3 Configuration and Deployment

Some configuration was required in midPoint in addition to the customization described before. MidPoint had a default password policy that defined the required structure of user passwords in the system. The AM application of Trusteq already had existing password policy rules, so the midPoint default password policy had to be removed to avoid conflicts between the password policies.

New physical users in midPoint are generally created by external resources as a result of identity synchronization. If midPoint finds a new unlinked account in AD for example, it will attempt to create that user in midPoint. New users are rarely created directly in midpoint. If a user is created directly in midPoint, it will not be synchronized. To allow the synchronization process to notice new users that have been created remotely by RPCs, the new users needed to be linked to a resource. MidPoint creates new users based on a user template. The template defines how new users should be created. The default user template had to be changed to allow all new users to also be created in the database resource. The new user template forces midPoint to create a new account in the database every time a new user is created.

The identity connectors that had been imported to midPoint were not usable by default. The database connector and AD connector had to be defined to act as a resource with a resource definition file. The resource definition is an XML file that describes how different user attributes and configuration are used in the connector to perform SPI operations.

4.4 Scaling the System

Server clustering was not relevant in this POC. It was a feature that will be addressed in the future when the implementation is developed further. In addition to server clustering, Trusteq also required support for connection pooling. The PoolableConnector interface was implemented in the custom Hibernate database connector. The interface basically just allowed the Hibernate connector to be pooled leaving multiple instances of the connector ready to use by midPoint thus increasing the efficiency of the database queries.

4.5 Enforcing Data Consistency

All of the user information was not tracked by midPoint. The rest of the user information had to be persisted to the database traditionally after midPoint had done its persistence operations. However, it was not possible to just persist the rest of the information because of the generic functionality of the AM application. Instead, the information updated by midPoint had to be retrieved from the database, merged with the rest of the data and then persisted again into the database. That way it would be certain that the persisted user is up to date.

The change log in the Hibernate connector helped to determine if the data in the database was really changed by midPoint. The Hibernate connector changes the last update time in the database which allows the AM application to compare the update time of the user in the session to the update time of the user that came from the database as it was specified in TASK-4. If the update time has changed, the AM application knows that the data has been updated by midPoint. When the AM application knows about the update performed by midPoint, TASK-5 states that the current session user has to be updated and persisted to complete the update. This feature can be developed further by saving multiple update times in the database. If there are multiple update times, midPoint could restore a user to a certain date.

4.6 Summary

All of the tasks specified by the business case were completed. TASK-1 was completed by changing how users are created in midPoint. New users were created by SOAP messages that originated from the Trusteq AM application. The default user template was changed to force midPoint to create new users into the database resource allowing midPoint identity synchronization to notice them. TASK-2 was completed by installing and configuring a default AD identity connector into midPoint.

TASK-3 was completed by creating a new customized identity connector that allows midPoint to execute persistence operations similar to the AM application of Trusteq. The custom identity connector was configured only to use create and update SPI operations to prevent unnecessary database queries in the database of Trusteq. The connector was implemented as a PooledConnector for efficient database communication.

TASK-4 and TASK-5 were completed by including a change log attribute in the custom database connector. The change log attribute caused the connector to store update dates in the database to allow the AM application to poll the database for updates. The AM application was modified to wait for midPoint updated before persisting user data.

Some of the compare criteria were not addressed in the implementation. They were not relevant in the PoC. The documentation and reliability criterion was not a functional requirement. It was only necessary to help study which solution offered better tools for implementation. It was important to know that Evolveum takes developing their product seriously. Evolveum offered a great deal of reliability in terms of documentation and development agenda. The midPoint documentation alone was enough to complete this implementation. However, studying the midPoint documentation takes time, so everything required for configuring and deploying the implementation was also documented in the Wiki page of Trusteq Oy.

The implementation is only a proof of concept. It is not a fully functional implementation by any means. The PoC was presented to the Trusteq software steering group as proof that an implementation like this is possible. Further actions regarding this PoC were left for the steering group to decide. Some ideas of further development was discussed with the steering group.

5 Conclusions

As a result of this thesis Trusteq received a PoC of a midPoint IDMS implementation. The PoC was an effective way of showing that an IDMS like midPoint can be integrated into the access management software of Trusteq. The implementation is an important asset for Trusteq Oy. The implementation can be used in customer demonstration situations and as a starting point for production development.

The required steps of the implementation were documented in the Wiki page of Trusteq. The documentation will back up the development process when it continues. When the time comes to continue developing the midPoint implementation, Trusteq has perfect tools for the implementation. The faster the implementation can be finished and tested, the faster it affects the company income.

MidPoint proved to be customizable enough for the integration. It was important to know that midPoint offers enough features and functions for the needs of Trusteq. Trusteq can count on midPoint when they implement a production IDM system. MidPoint is open source and the license allows customizing, sharing and selling the product. It makes mid-Point really cost efficient.

Trusteq Oy was able to improve their data security with the midPoint Proof-of-Concept. Data security is a very important concept for Trusteq. They want to make sure that their products are as secure as possible. The customers of Trusteq rely on the fact that the products of Trusteq are data secure. Lack of security reduces credibility thus affecting the amount of customers.

Trusteq Oy was also able to improve data consistency with midPoint. Previously inconsistent data had direct influence in the expenses of Trusteq because it created a possibility for unused user accounts in their network. MidPoint also removed the need for individually update identity information in AD thus removing the chance for a former employee to have access in AD for example.

The compare criteria was discussed with the project team of Trusteq after the project had finished. The project team decided to leave some of the initial requirements to be implemented when the development of the PoC is continued in the near future. Those requirements simply were not relevant in the PoC implementation.

Future development ideas were discussed with the Trusteq project team. Some of the initial requirements will be taken into account when the IDMS is developed further. Implementing multi-tenancy was one of the future development ideas. The PoC implementation did not address multi-tenancy because it was not relevant in terms of demonstrating the functionality between the AM, midPoint and the database. However, multi-tenancy will be important when the implementation is developed further. A customer most likely needs individual tenants for their own customers. The tenants can then have their own directory resources, database resources, HR resources, etc. Those resources can then be managed by the provider. Also, in the future there might be a case where a tenant is allowed to even add their own resources, then multi-tenancy is used to its full extent.

With multiple tenants and increasing numbers of users in the system, server clustering will become important. Server clustering was specified as a desired feature, but the POC implementation does not implement server clustering. To guarantee that the service is always available, multiple instances of midPoint will have to be deployed. All of the instances will use the same resources, but in case one instance fails, the active processes are moved to another instance. Server clustering should be implemented and tested in the future.

All in all, the PoC was successfully implemented with midPoint documentation. MidPoint had enough customizability for Trusteq. MidPoint increased data security and data consistency in the network of Trusteq. Future development tasks were approved and documented.

References

Abelson, H., Lessig, L., Covell, P., Gordon, S., Hochberger, A., Kovacs, J., Krikorian, R., Schneck, M. 1998. Digital Identity in Cyberspace. URL: http://groups.csail.mit.edu/mac/classes/6.805/student-papers/fall98-papers/identity/linked-white-paper.html. Accessed: 12 April 2015.

Apache Syncope. 2010. Apache Syncope. URL: http://syncope.apache.org/. Accessed: 4 March 2015.

Apache Syncope. 2012. OpenSource IdM Managing Identities in Enterprise Environments. URL: http://syncope.tirasa.net/assets/content/assets/syncope_en/apache-syncope-opensource-idm.pdf. Accessed: 21 April 2015.

Azstrel Tech-Soft Pvt Ltd. 2013. Identity. URL: http://www.azstrel.com/identity.php. Accessed: 13 April 2015.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., Yergeau, F. 2008. Extensible Markup Language (XML) 1.0. Fifth Edition. URL: http://www.w3.org/TR/REC-xml/. Accessed: 15 April 2015.

Casassa Mont, M., Baldwin, A., Shiu, S. 2009. Identity Analytics - "User Provisioning" Case Study: Using Modelling and Simulation for Policy Decision Support. URL: http://www.hpl.hp.com/techreports/2009/HPL-2009-57.pdf. Accessed: 13 April 2015.

Chicchiriccò, F. 2012. Setup a Syncope Cluster. URL: https://cwiki.apache.org/confluence/display/SYNCOPE/Setup+a+Syncope+cluster. Accessed: 21 April 2015.

Chicchiriccò, F. 2015a. Upgrade to ConnID 1.4.0.0. URL: https://issues.apache.org/jira/browse/SYNCOPE-537. Accessed: 6 April 2015.

Chicchiriccò, F. 2015b. [DISCUSS] Realms. URL: https://cwiki.apache.org/confluence/display/SYNCOPE/%5BDISCUSS%5D+Realms. Accessed: 5 April 2015.

Christensen, E., Curbera, F., Meredith, G., Weerawarana, S. 2001. Web Services Description Language (WSDL) 1.1. URL: http://www.w3.org/TR/wsdl. Accessed: 3 May 2015.

E-mail discussion. 2013. [DISCUSS] ConnId restructuring (including full OpenICF compatibility). URL: https://groups.google.com/forum/#!msg/connid-dev/mKBhLZaKR1Q/NFQio1ZJzt8J. Accessed: 20 April 2015.

Evolveum s.r.o. 2014a. Approach. URL: https://www.evolveum.com/approach/. Accessed: 6 April 2015.

Evolveum s.r.o. 2014c. Pragmatism. URL: https://www.evolveum.com/pragmatism/. Accessed: 6 April 2015.

Evolveum s.r.o. 2014b. midPoint. URL: https://www.evolveum.com/midpoint/. Accessed: 6 April 2015.

ForgeRock AS. 2010. Interface PoolableConnector. URL: http://openicf.forgerock.org/connector-framework/apidocs/org/identityconnectors/framework/spi/PoolableConnector.html. Accessed: 10 April 2015.

Gartner, Inc. and/or its Affiliates. 2013. User provisioning. URL: http://www.gartner.com/it-glossary/user-provisioning. Accessed: 13 April 2015.

Hitachi ID Systems, Inc. 2015. User Provisioning Best Practices. URL: http://hitachi-id.com/identity-manager/docs/user-provisioning-best-practices.pdf. Accessed: 13 April 2015.

IBM Corp. 2013. Application Programming Guide and Reference for Java. Eight edition. DB2 10 for z/OS.

Jøsang, A. and Pope, S. 2005. User Centric Identity Management. URL: http://www.researchgate.net/profile/Audun_Josang/publication/228623015_User_centric_identity_management/links/53d759aa0cf228d363eaf44e.pdf. Accessed: 5 March 2015.

Mederly, P. 2011. Consistency Model. URL: https://wiki.evolveum.com/display/midPoint/Consistency+Model. Accessed: 21 April 2015.

Mederly, P. 2013. High Availability and Load Balancing. URL: https://wiki.evolveum.com/display/midPoint/High+Availability+and+Load+Balancing. Accessed: 21 April 2015.

Mederly, P. 2014. REST API. URL: https://wiki.evolveum.com/display/midPoint/REST+API. Accessed 21 April 2015.

Microsoft. 2003. Evaluationg the Benefits of Clustering. URL: https://technet.microsoft.com/en-us/library/cc778629%28v=ws.10%29.aspx. Accessed: 21 April 2015.

Oracle. 2011. wsimport - Java™ API for XML Web Services (JAX-WS) 2.0. URL: https://docs.oracle.com/javase/6/docs/technotes/tools/share/wsimport.html. Accessed: 4 May 2015.

Oracle. 2012. Identity Connectors Framework. URL: https://wikis.oracle.com/display/IdentityConnectors/Identity+Connectors+Framework. Accessed: 20 April 2015.

Oracle. 2015. Understanding the Identity Connector Framework. URL: http://docs.oracle.com/cd/E21764_01/doc.1111/e14309/icf.htm#OMDEV3332. Accessed: 10 April 2015.

Peterson, D. 2009. What is Kanban?. URL: http://kanbanblog.com/explained/. Accessed: 13 April 2015.

RaulWalter. 2015. MobileID (WPKI). URL: https://www.raulwalter.com/government/mobile-id-wpki/. Accessed: 5 May 2015.

Rouse, M. August 2014. Multi-tenancy. URL: http://whatis.techtarget.com/definition/multi-tenancy. Accessed: 3 April 2015.

Rouse, M. & Mathias, C. October 2013. Identity management (ID management). URL: http://searchsecurity.techtarget.com/definition/identity-management-ID-management. Accessed: 30 May 2015.

Scrum.org and ScrumInc. 2014. The Scrum Guide. URL: http://www.scrumguides.org/scrum-guide.html. Accessed: 8 April 2015.

Semancik, R. 2012. Common Data Model. URL: https://wiki.evolveum.com/display/midPoint/Common+Data+Model. Accessed: 6 April 2015.

Semancik, R. 2014a. IDM Model Web Service Interface Examples. URL: https://wiki.evolveum.com/display/midPoint/IDM+Model+Web+Service+Interface+Examples. Accessed: 15 April 2015.

Semancik, R. 2014b. Business Benefits of Identity Provisioning. URL: https://wiki.evolveum.com/display/midPoint/Business+Benefits+of+Identity+Provisioning. Accessed: 19 April 2015.

Semancik, R. 2014c. Organizational Structure. URL: https://wiki.evolveum.com/display/midPoint/Organizational+Structure. Accessed: 27 April 2015.

Semancik, R. 2015a. Identity Connectors. URL: https://wiki.evolveum.com/display/midPoint/Identity+Connectors. Accessed: 10 April 2015.

Semancik, R. 2015b. IDM Model Web Service Interface. URL: https://wiki.evolveum.com/display/midPoint/IDM+Model+Web+Service+Interface. Accessed: 21 April 2015.

Semancik, R. 2015c. IDM Model Interface. URL: https://wiki.evolveum.com/display/midPoint/IDM+Model+Interface. Accessed: 21 April 2015.

Semancik, R. 2015d. Apache Syncope. URL: https://compare.evolveum.com/details-syncope.html. Accessed: 21 April 2015.

Shalom, N. 2010. Multi-tenancy: does it have to be that hard? URL: http://natishalom.typepad.com/nati_shaloms_blog/2010/03/multitenancy-does-it-have-to-be-that-hard.html. Accessed: 21 April 2015.

SMARTBEAR. SOAP vs REST Challenges. URL: http://www.soapui.org/testing-dojo/world-of-api-testing/soap-vs--rest-challenges.html. Accessed: 21 April 2015.

Sullivan, D. Chapter 5: Identity and Access Management. URL: http://cdn.ttgtmedia.com/searchSecurity/downloads/Sullivan_Ch5.pdf. Accessed: 30 May 2015.

The Apache Software Foundation. January 2004. Apache License, Version 2.0. URL: https://www.apache.org/licenses/LICENSE-2.0. Accessed: 5 April 2015.

The Apache Software Foundation. 2010a. Apache Syncope. URL: http://syncope.apache.org/architecture.html. Accessed: 5 April 2015.

Waters, J. 2004. The ABCs of Identity Management. URL: http://www.csoonline.com/article/2120384/identity-management/the-abcs-of-identity-management.html. Accessed: 12 April 2015.

Winer, D. 1999. XML-RPC Specification. URL: http://xmlrpc.scripting.com/spec.html. Accessed: 15 April 2015.

Yeates, S. 2005. Documentation Issues in Open Source. URL: http://oss-watch.ac.uk/resources/archived/documentation. Accessed 5 April 2015.

Appendices

Appendix 1. Example SOAP message in midPoint IDM Model Web Service

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
        <wsse:Security
            xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
            xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
            soap:mustUnderstand="1">
            <wsse:UsernameToken
                wsu:Id="UsernameToken-4028eae9-3f47-447f-82f8-8e88becd71d5">
                <wsse:Username>administrator</wsse:Username>
                <wsse:Password
                    Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss
                    -username-token-profile-1.0#PasswordDigest">PimHMwl8PR5ixPiH5Zggkm/LBa8=</wsse:Password>
                <wsse:Nonce
                    EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss
                    -soap-message-security-1.0#Base64Binary">6QDpLmFtpQZ+qzEZg+RX8w==</wsse:Nonce>
                <wsu:Created>2014-11-03T19:03:00.792Z</wsu:Created>
            </wsse:UsernameToken>
        </wsse:Security>
    </SOAP-ENV:Header>
    <soap:Body>
        <m:getObject
            xmlns:m="http://midpoint.evolveum.com/xml/ns/public/model/model-3"
            xmlns:c="http://midpoint.evolveum.com/xml/ns/public/common/common-3">
            <m:objectType>c:UserType</m:objectType>
            <m:oid>620237a0-6393-11e4-8e1b-3c970e467874</m:oid>
            <m:options />
        </m:getObject>
    </soap:Body>
</soap:Envelope>
```

Appendix 2. Theoretically supported connectors in midPoint

Reasonably supported connectors:

- Solaris Connector (Synchronization not supported)
- Exchange Connector (Synchronization not tested)
- Office365 (Synchronization not tested)
- GitLab (Developed by Evolveum)
- Lotus Notes Connector (Developed by Evolveum, does not support synchronization)

Not tested, theoretically functional connectors:

- AD (JNDI) Connector (ConnId)
- SOAP Connector (ConnId)
- CMD Connector (ConnId)
- CSV Directory Connector (ConnId)
- Google Apps Connector (ConnId)
- OpenAM Connector (ConnId)
- UNIX Connector (ConnId)
- DB2 Connector
- MySQL User Connector
- Oracle Connector
- Flat File Connector
- XML Connector
- VMS Connector
- OpenPortal Connector
- SPML Connector
- SAS Connector