ICSIIT 2017

2017 International Conference on Soft Computing, Intelligent System and Information Technology

26-29 September 2017 • Denpasar, Bali, Indonesia

BUILDING INTELLIGENCE THROUGH IOT AND BIG DATA

Organized by

INFORMATICS DEPARTMENT
Petra Christian University

CONTINUING
EDUCATION
CENTRE

Supported by

tokopedia

PT. Catalyst Solusi Integrasi
IT Security Integrator

ORACLE
ACADEMY

edugate

Telkom Indonesia
the world in your hand

In Collaboration with

NTUST

INDUSTRIAL ENGINEERING
Petra Christian University

CONFERENCE INFORMATION

PAPERS BY SESSION

PAPERS BY AUTHOR

GETTING STARTED

TRADEMARKS

SEARCH

2017 International Conference on Soft Computing, Intelligent System and Information Technology

ICSIIT 2017

Table of Contents

Classification and Correlation Techniques

Feature Extraction and Image Recognition Methods

Algorithms for Intelligent Computation

Distributed Systems and Computer Networks

Mobile and Pervasive IoT Applications

Assessments of Integrated IS/IT

Simulation and Virtual Reality Applications

Smart Assistive Technologies

Smart Mobile Applications

Case Studies of Knowledge Discovery and Management

Linux PAM to LDAP Authentication Migration

Justinus Andjarwirawan, Henry Novianus Palit, Julio Christian Salim
Department of Informatics
Petra Christian University
Surabaya, Indonesia
justin@petra.ac.id, hnpalit@petra.ac.id, julio.christiansalim@hotmail.com

Abstract—**Authentication of Linux system users are maintained by PAM (Plugable Authentication Module). LDAP (Lightweight Directory Access Protocol) is used to replace the local user account authentication. Our existing environment uses PAM through Dovecot POP3 and RADIUS for authentication.This research focuses on the implementation of LDAP as an authentication service and migration of accounts and applications. Implementation and migration using account data that is using P AM. An application is also developed to search and modify existing account, and creating new account. Based on test performed, the account migration is successful. Testing the migrated accounts were performed. Also response time that is needed by LDAP to authenticate is shorter than the existing system which uses POP3 and RADIUS.**

Keywords—Migration; LDAP; PAM; Linux; authentication.

I. Introduction

Legacy Linux systems use PAM (Pluggable Authentication Module) for system users to be able to authenticate and gain access to the shell or any other service that needs authentication before they can be used. PAM uses local system users shadow passwords on Linux, it is the regular user account on Linux systems.

If the authentication system with local user remains in use, the operating system used for the authentication server will be difficult to update, potentially causing security issues, compatibility issues with newer authentication methods. Because PAM / local user on Linux also does not store much information about the user. PAM can only store information about the name of the user, while information such as phone numbers and addresses of users cannot be stored. Also, it is inconvenient in user management because there is no search user function provided. PAM also has a weakness in terms of security in the authentication, all messages are stored in plaintext. Local users information of Linux are stored in /etc/passwd and the encrypted password in /etc/shadow.

On that condition, it requires an authentication system that can store more complete information about user and the migration must be successful, seamless, compatible and future-proof.

With LDAP (Lightweight Directory Access Protocol), user migration is still possible to implement. Because LDAP provides a way to export users, it is possible to move from an existing LDAP server to another LDAP server [1]. The LDAP server is also not tied to the operating system used. For example, the LDAP server running on Debian Linux and Ubuntu has no difference in the configuration and installation, but the most important thing is the version of LDAP used. In LDAP the stored user information is also flexible, since the attributes in the user can be made as needed. LDAP also has a library for the PHP programming language that makes it easy to create authentication for web applications in the future. LDAP itself by default sends an authentication message with plaintext format, but LDAP has facilities where messages sent can be encrypted using TLS (Transport Layer Security) / SSL under the name LDAPS [2]. LDAP is also a Neutral Platform which means LDAP is not attached to a particular operating system. LDAP server can run on various operating systems.

II. Research Overview

A. The Approach

There are some problems background need to be formulated for this migration, they are:

- Configuring LDAP server and make a copy of user database from the Linux system users to LDAP.
- Check the existing third party applications or clients which use the authentication, making sure LDAP authentication works and compatible.
- Performing LDAP usage analysis for web apps authentication, as most applications support web services.

B. Scope of Research

The migration is delivered from an older Debian Linux distribution version 6 to version 8, OpenLDAP version 2.4.40, running on virtual machine VMware ESXi. The LDAP protocol used is version 3.

The success criteria of this research is users from third party applications and local system users must be able to authenticate seamlessly without any notice of migration from the backend. Passwords from /etc/shadow [3] file are properly copied to LDAP database, and PAM redirects the authentication to LDAP service.

III. Theories

A. PAM (Pluggable Authentication Module)

PAM is the integration mechanism of some low-level authentication schemes to high-level Application Programming Interface (API). On Debian Linux [4], PAM [5] uses

978-1-4673-9899-2 /17 $31.00 © 2017 IEEE
DOI 10.1109/ICSIIT.2017.66

local user data as the user database. Local user credentials in Debian Linux is stored in /etc/passwd and encrypted passwords in /etc/shadow. With PAM, applications can authenticate to local user data in both /etc/passwd and /etc/shadow.

In PAM settings, there are 4 configurable contexts, namely auth, account, password, session. Each context has different functions. Whereas in the context of those contexts there are 6 control flags that can be used in accordance with the PAM requirement i.e. required, requisite, sufficient, optional, include, substack. Control flag also has a different function, but the feedback obtained from the control flag is not issued as is. For example, if the result of control flag is "PAM_ACCT_EXPIRED" then PAM is only considered "failure".

According to Geisshirt [5], PAM reduces the complexity of authentication, since system administrators can use the same user database for all Logins on the system. Commonly used services in Linux such as SSH (Secure Shell), FTP (File Transfer Protocol), TELNET can directly use local user data in the operating system.

Basically, PAM can be modified to authenticate to LDAP. So LDAP and PAM can run simultaneously. But to make modifications need to make changes to some system files owned by PAM.

B. LDAP (Lightweight Directory Access Protocol)

LDAP is a network protocol that uses directories to store user data. LDAP is created as a general purpose directory server which means LDAP is not created to store certain data, but rather the data stored more flexible and can be tailored to the needs. To distinguish similar user data in the directory, LDAP uses a DN (Distinguished Name) example: "ou = example, dc = example, dc = com".

Directory and general purpose database are often considered the same, but actually the directory is a database that has several characteristics that distinguish from relational database. One of the characteristics of a directory is that directories are more frequently accessed for read and search than updates. Directory stores relatively static data, which does not change often. Most databases use the Structured Query Language (SQL) access method, while the directory uses a simpler access method.

Fig. 1 shows the structure of DIT (Directory Information Tree). The Directory Information Tree has data entries written in DN (Distinguished Name). The DN itself has an RDN (Relative Distinguished Name) separated by a comma. The written RDN affects the branch or branch of the data entry directory.

- LDAP has an additional library for PHP. With libraries from LDAP, applications created with PHP can access LDAP-owned user data, authenticate, make changes to existing user data.
- Benefits of using LDAP:
- LDAP uses TCP/IP protocol and can run on SSL (Secure Sockets Layer).
- LDAP can be a central store of information for members of an organization.
- LDAP can be used as an authentication center.
- LDAP has been implemented in some applications.

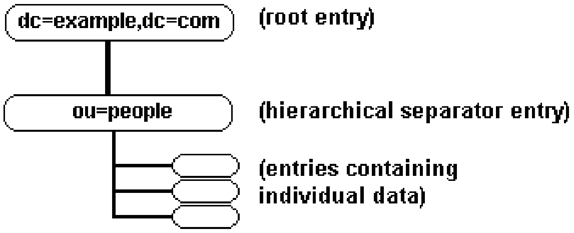
dc=example,dc=com
(root entry)
ou=people
(hierarchical separator entry)
(entries containing individual data)

Figure 1. DIT (Directory Information Tree) structure

Some implementations such as "Windows Active Directory" only run on Windows operating systems, while other implementations such as OpenLDAP run on Linux operating systems. OpenLDAP itself is an open source implementation. OpenLDAP has several parts i.e.:

- Slapd
- Libraries
- Tools

Slapd stands for "Stand-alone LDAP Daemon". Slapd listens for LDAP connections from a specified port. Generally, the ports used for LDAP connections are port 389, while port 636 is used for LDAPS (LDAP over SSL) connections.

The libraries used by OpenLDAP are called libldap. Libldap API supports OpenLDAP functions that use TCP, SSL and IPC protocols [7]. To use some tools already provided, it needs to have an account on the LDAP server.

The interaction between the LDAP client and the LDAP server is done in several stages:

- Client creates session with LDAP server. It is also known as creating bindings with servers.
- Client authenticates with LDAP server using username and password. The LDAP server allows authentication without a username and password.
- Client performs operations on existing data on the server. The operations include making changes to existing data on the server and read existing data.
- When the operation is done, the client closes the session.
- Connections to LDAP can use TLS that encrypt connections. The use of TLS connection can only be done when using LDAP protocol version 3.
- LDAP has a LDIF file format LDIF (LDAP Data Interchange Format) to simplify data changes in the LDAP directory.
- For LDAP servers that do not use the ldap.conf file for server settings but instead use the configuration in the cn = config folder, the LDIF file is used for configuration changes on the LDAP server. But there are some things that his configuration is stored not on cn = config, so still do the manual changes.

Interactions between LDAP clients and LDAP servers for connections using SSL / TLS through the following steps:

- Client asks for SSL / TLS session opening.
- The server sends a certificate containing the server's own private key, data about the owner of the certificate, the name of the certificate maker, and the expiration limit of the certificate.
- The client requests the server to prove its true identity that the server is indeed the actual certificate sender. This is done to ensure the certificate is not sent by another server.
- The server sends a message containing an encrypted message digest using a private key owned by the server.
- Client compares decrypted message digest using public key obtained from server and compared with message digest obtained by trying to make message digest from message sent by server. If the result is the same then the identity of the server is correct.
- Server and client create secret key for data encryption sent between client and server. The secret key is symmetric. An encrypted message with a public key owned by the client can only be decrypted by the private key owned by the server.
- The client encrypts the secret key with the public key obtained from the server. The secret key is sent to the server.
- The secret key medication server sent by the client using the server's private key.
- The server sends an encrypted test message with the secret key to prove the secret key safely.
- Client decrypts a test message that the server sends using a secret key.

In order to store passwords from users on the user Password attribute, OpenLDAP accepts some form of password storage method with hashes that are MD5, SMD5, Crypt, SHA, and SSHA. According to the OpenLDAP creator, the SSHA hash method is the safest one supported by slapd (OpenLDAP Foundation,). OpenLDAP can also store passwords without going through a hash process so the saved passwords are plaintext. OpenLDAP encodes passwords that have been through the hash process to base64 in its database storage.

The LDAP backend system can use one of three database options: BDB (Berkeley Database), HDB, and MDB (memory-mapped database). BDB uses Oracle Berkeley Database to store data, HDB is an improvement of BDB. MDB is created to replace BDB and HDB. MDB uses OpenLDAP's library called LMDB (Lightning Memory-mapped Database) to store data. LMDB supports indexing such as BDB and HDB but does not require caching and tuning to provide the best search performance [1].

C. Schema

In the data storage of user entries in LDAP, there are some related things i.e. schema, objectClasses, attributes, and entries. Fig. 2 shows the relation between schema, objectClasses, attributes, and entries. Data entry must follow an existing schema. Entries are created based on the schema rules specified by creating objectClasses. Schema determines the existing objectClasses.

ObjectClasses must already be specified in the schema to be used in data entry, objectClasses specifies the attributes that must exist and the attributes that may be present in the entry of the user. There are various objectClasses provided from the LDAP server application (such as openLDAP). When a new attribute is created and has not been previously defined in objectClasses, a new objectClasses must be created first. When the attribute is specified in objectClasses, then the attribute can be created. The attribute itself can have more than 1 value. It is specified in objectClasses. LDAP rejects the creation of a user that has an attribute that is not in the specified objectClasses in the schema. ObjectClass is created by invoking ldapmodify command with an LDIF file that uses the new Object class creation format.

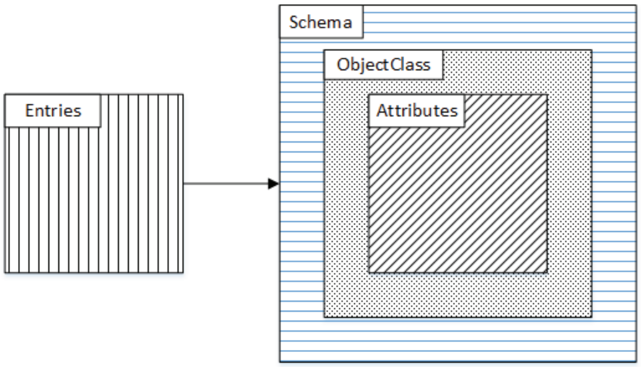

Figure 2. LDAP Schema Structure

OID is used as an object identification number. Attributetype and objectclass use OID for identification. OID is hierarchy, so an OID (example: 2.25.1234567) can have its own branches. OID is unique and should not be the same as other OID. To get a new OID one must register to the organization that sets the OID, registration to IANA (Internet Assigned Numbers Authority) is an example. There is one OID branch that can be used for experimental OID with the prefix "2.25." Followed by UUID behind it [6].

ObjectClass stores information about the attributes that can and should be used. In addition, objectclass also stores objectclass type, description, name, objectclass superior. ObjectClass has a unique OID (Object Identifier), it should not be the same as another OID. The OID should actually be registered to the organization that governs the OID. For example IANA for America. In creating ObjectClass, it takes several attributes. The attributes required and can be used in the creation of the objectclass. There is one OID branch that can be used freely without OID registration. That is an OID beginning with 2.25 and forwarded with UUID obtained from http://www.itu.int/en/ITU-T/asn1/Pages/UUID/uuids.aspx

D. AC (Access Control)

The AC in OpenLDAP serves as a permission setting to an existing account. AC needs to define its order and contents. OpenLDAP reads the AC in the order of smallest (top order) to largest (lowest order), and when it finds what the user wants to do, OpenLDAP will stop reading from the AC and make AC access restricted. Access levels are related to each other. When the level of "read" is given, then access to "search" with lower level is also given.

E. PAM migration to LDAP

In migration, user data is retrieved from the local user in the /etc/passwd and /etc/shadow files. From file /etc/passwd the username and full name are retrieved. The password of the username is obtained from the /etc/shadow file. The password format in the /etc/shadow file is a hash instead of plaintext.

Migrations from the /etc/shadow and /etc/passwd files are done using a tool called migrationtools. Migrationtools itself is created using perl language. Migrationtools has several modules. All modules are connected to a file called migrate_common.ph. The migrate_common.ph file stores the information data about the "dc" of the destination LDAP. Migrated users have their passwords in MD5 hash processed with salt.

IV. DESIGN AND IMPLEMENTATION

A new server has been setup using Debian Linux on top of a hypervisor virtual machine by VMware ESXi. This new server is fully prepared intended for authentication service using LDAP. The packages that must be installed for LDAP server are: slapd, ldaputils, migrationtools and phpldapadmin.

The main important thing about this migration is the copy of /etc/passwd entries and /etc/shadow hashes from old server to LDAP. Entries of user accounts will be stored in LDIF (LDAP Data Interchange Format). An example of one entry in LDIF is shown below:

```
dn: uid=justin,ou=subdomain,dc=domain
uid: justin
cn: Justinus Andjarwirawan
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: Petra
objectClass: shadowAccount
userPassword: ********************************
shadowLastChange: 15159
shadowMax: 99999
shadowWarning: 7
loginShell: /bin/sh
uidNumber: 1000
NomorTelepon: 0
NamaDepartemen: 0
gidNumber: 1000
homeDirectory: /home/justin
gecos: Justinus Andjarwirawan,,,
```

A web based application is developed in order to copy current passwords from existing applications which is using the previous authentication system PAM. The web based application and the existing applications are based on PHP. Basically it runs the same function as creating a new LDAP user entry. In PHP the password generator is using a built-in function to generate salted SHA1 hash:

```
$salt = uniqid(openssl_random_pseudo_bytes(8),
        true);
$hash = "{SSHA}" . base64_encode(hash('sha1',
        $password. $salt, true) . $salt);
ldap_mod_replace ($ds, $username1,
        array('userPassword' => $hash));
```

Some of the legacy applications are using POP3 and IMAP for authentication. It is possible to redirect authentication check from PAM to LDAP by modifying the POP3 and IMAP service configuration. In this case Dovecot server is used to run POP3 and IMAP servers. Changes in the Dovecot configuration in dovecot-ldap.conf file are:

```
dn = cn=admin,dc=petra,dc=ac,dc=id
dnpass = <password root LDAP>
ldap_version = 3
base = ou=subdomain,dc=domain
scope = subtree
user_attrs = homeDirectory=home, uidNumber=uid,
             gidNumber=gid
user_filter = (&(uid=%u))
pass_attrs = uid=user,userPassword=password
pass_filter = (&(uid=%u))
```

And the modification in dovecot.conf is:

```
passdb ldap {
  # Path for LDAP configuration file
  args = /etc/dovecot/dovecot-ldap.conf
}
userdb ldap {
  # Path for LDAP configuration file
  args = /etc/dovecot/dovecot-ldap.conf
}
```

This will allow Dovecot to authenticate by PAM first and when it fails Dovecot will authenticate to the configured LDAP service. By doing this the migration will run seamlessly. When all users have authenticated through LDAP within time then the PAM authentication can be turned off.

V. EVALUATION

Upon several successful authentication attempts from users and applications, the LDAP server is evaluated for its performance and security. For the performance test, Apache Bench is used as one of the tools to determine LDAP server’s response time; and the other tests are CPU utilization, and memory usage.

The first test is LDAP authentication performance from 1 to 100 concurrent connections with TLS (Transport Layer Security) support. The results are shown in Table I.

TABLE I. LDAP AUTHENTICATION PERFORMANCE WITH TLS

Number of Connection(s)	Response Time (in seconds)	CPU Load
1	0.008749723	6.90%
10	0.064053321	10.00%
100	0.064456030	57.00%

It is shown that there is a significant performance gap from 1 to 10 concurrent users, but very little gap between 10 and 100 concurrent users.

The second test will perform an LDAP authentication performance test without TLS support. TLS is a layer of process within the authentication so it will affect the overall performance but increasing the security risk. Table II shows the test results.

TABLE II. LDAP AUTHENTICATION PERFORMANCE WITHOUT TLS

Number of Connection(s)	Response Time (in seconds)	CPU Load
1	0.001240325	1.60%
10	0.002037811	4.00%
100	0.007709187	20.00%

Eliminating the TLS will give a very significant performance raise from the TLS supported LDAP and also there is no significant difference in performance from 1 to 100 concurrent connections.

To make the evaluation complete, the legacy performance of PAM authentication is also tested. The PAM test is using the POP3 and the result is shown in Table III.

TABLE III. POP3 AUTHENTICATION PERFORMANCE TEST

Number of Connection(s)	Response Time (in seconds)	CPU Load
1	0.014850640	2%
10	0.202573860	4%
100	2.581857831	12%

The legacy PAM authentication, in this case is POP3, have much lower performance in have load of 100 concurrent users. This performance result indicates the POP3 authentication needs the amount of time to read the /etc/passwd and /etc/shadow to perform password hash checking; compared to LDAP which is using the database entries, in this case MySQL.

VI. CONCLUSION

A successful migration is when the users are able to authenticate and change passwords without awareness of the migration behind. After all users' passwords collected through a transition attempts in the Dovecot configuration, the POP3 authentication is turned off and Dovecot directly attempts the authentication to LDAP. The LDAP service is now running on a dedicated server for authentication.

As a supplement, a password change module is developed in a PHP web based application, which is using a built-in LDAP function to hash the plain text password. It is placed along with LDAP user administration portal with access levels of security.

ACKNOWLEDGMENT

Thank you to the ICSIIT committees for the chance to write this paper as part of the 2017 ICSIIT conference in Bali, Indonesia. It is an honor for the writer to be part of the conference and build a network of researchers in the same area of interest. Hopefully the next ICSIIT event will gather more partners to gain participants.

REFERENCES

[1] OpenLDAP Project, "OpenLDAP Software 2.4 Administrator's Guide: Backends," ch. 11, Jun. 1, 2017. [Online] Available: http://www.openldap.org/doc/admin24/backends.html.

[2] OpenLDAP Project, "OpenLDAP Software 2.4 Administrator's Guide: Security Considerations," ch. 14, Jun. 1, 2017. [Online] Available: http://www.openldap.org/doc/admin24/security.html.

[3] S. Frampton, "Linux Administration Made Easy: Linux Password & Shadow File Formats," subch. 6.6. [Online] Available: http://www.tldp.org/LDP/lame/LAME/linux-admin-made-easy/shadow-file-formats.html.

[4] R. Herztog and R. Mas, The Debian Administrator's Handbook (Debian Wheezy), Freexian SARL, 2013.

[5] K. Geisshirt, Pluggable Authentication Modules: The Definitive Guide to PAM for Linux SysAdmins and C Developers, Birmingham (UK): Packt Publishing, 2007.

[6] ITU, "Universally Unique Identifiers (UUIDs)." [Online] Available: http://www.itu.int/en/ITU-T/asn1/Pages/UUID/uuids.aspx.

[7] R. Winston, "Using libldap, the LDAP Library Client," O'Reilly ONLamp.com, Aug. 14, 2003. [Online] Available: http://www.linuxdevcenter.com/pub/a/linux/2003/08/14/libldap.html.