令和 5 年度高エネルギー加速器研究機構技術研究会、2024 年 3 月 7～8 日
https://www2.kek.jp/engineer/tsukuba/giken/2023/

LDAP と Samba を組み合わせたファイル共有システムの作成

金城行真#
分子科学研究所 技術推進部

概要

分子科学研究所では、現在 LDAP と WebDAV を組み合わせたファイル共有システムを運用している。しかし、接続が安定しない等このシステムに対してユーザから若干の不満が上がっている。これに対し、LDAP に連携させる形で Samba を立ち上げ、導入コストを抑えながら、ユーザーフレンドリーかつ接続の安定したファイル共有システムを立ち上げようと試みた。結果的に実現には至らなかったものの、LDAP-Samba 間通信の暗号化やユーザアクセスの制御、自動履歴機能の実装等、様々な知見が得られた。今回の報告では、導入の背景にも触れつつ、それらの知見を共有する。

1. 背景

分子科学研究所では、現在 LDAP と Windows WebClient を組み合わせたファイル共有システムを運用している。具体的には、LDAP サーバに格納済みの認証情報を用いて、部門の職員が共同で編集する web サーバに Windows WebClient で接続するという体制を取っている。しかし、接続が安定しない等ユーザから若干の不満が上がっている。また、CarotDAV 等の WebDAV クライアントソフトの導入にはユーザ側の心理的な抵抗が存在する。これに対し、Windows ライクな UI を提供できる Samba[1]であれば、ユーザの心理的な抵抗を最小限にしつつ、Windows WebClient よりも安定した接続を提供できるのではないかと考えた。また認証情報やその他諸々のデータを LDAP サーバに格納し、Samba サーバがそれを取り出し利用する仕組みを整えることができれば、現行の体制をあまり変化させずに移行が可能となる。すなわち、管理者側が負担するイニシャル/ランニングコストを抑えつつシステムを改良できる。このような LDAP と Samba を連携させたファイル共有システムの作成を目標に、テスト環境を作成し、そこで様々な検証を行った。

2. 目標設定

今回作成するファイル共有システムでは、既存のものと同様 LDAP サーバに格納されたユーザ情報を Samba サーバでの認証に利用することを目指した。LDAP-Samba 間の通信は SSL/TLS 化、つまり LDAPS（エルダップエス）化するものとした。また、ユーザ毎のアクセス制御機能についても実装することとした。具体的には、ユーザ毎の個人フォルダと共同作業用フォルダを用意する。ユーザ A は自身の個人フォルダと共同作業用フォルダにはアクセスできるが、ユーザ B の個人フォルダにはアクセスできない、という状況を目指した。加えて、自動履歴機能についても実装を目指した。具体的には、ユーザの意思とは無関係に自動でファイルの履歴、つまりバックアップを作成し、必要時には現行のファイルをバックアップファイルに置き換えることができるような状態を目指した。目標とするシステムの概略図を以下に示す（図 1）。

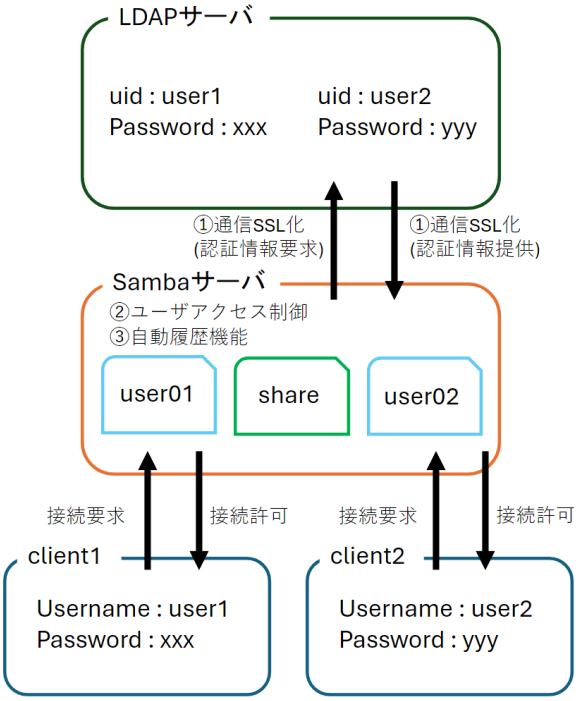

図 1. 目標とするシステムの概略図

特に実装を目指した機能については通し番号を記載した（①LDAP-Samba 間の通信の SSL 化、②ユーザアクセスの制御、③自動履歴機能）。特に②について、例えば user01 はフォルダ”user01”と”share”にはアクセスできるが、フォルダ”user02”にはアクセスできない。

3. 検証用環境の準備

3.1 検証用仮想環境の構築

Windows 11 搭載のデスクトップ PC（Z790-PLUS D4、ASUS 製）に VirtualBox 7.0.8 GUI[2]をインストールし、その中にゲスト OS を複数配置することで検証用の仮想環境を構築した。VirtualBox インストール時の設定は全てデフォルトとした。また、各ゲスト OS 間やホスト OS-ゲスト OS 間の通信にはブリッジ接続を用いた。インストールメディアには Rocky Linux 公式

kaneshiro@ims.ac.jp

ダウンロードページ[4]で配布されている Rocky-9.2-x86_64.minimal.iso を用いた。

3.2 LDAP サーバの作成

LDAP サーバのホスト名は”389ds”とした。これは、LDAP サーバ用ソフトウェアとして 389 Directory Server[5]（以下 389DS）を導入したためである。また、サーバ操作用のコマンドを使うために OpenLDAP 関連パッケージも同時に導入した。

```
[root@389ds ~]# dnf install 389-ds-base openldap-servers openldap-clients
```

導入された 389DS のバージョンは 2.3.6 であった。続いて、パッケージに含まれる dscreate コマンドを使い、対話形式で初期設定を行った。

```
[root@389ds ~]# dscreate interactve
（以下対話形式で設定）
```

上記コマンドによる初期設定の過程で自己署名証明書の作成を問われるため、作成するよう指示した。LDAPS の実装にはこの自己署名証明書を用いた。作成するインスタンス名は 389DS1、ベース DN は”dc=kaneshiro,dc=localnet”、管理者（ルート）DN は”cn=Manager,dc=localnet”と指定し、残りの項目にはデフォルトの設定値を用いた。初期設定が終了した後、以下に示す ldif ファイルをエディタで作成し、ldapadd コマンドで LDAP ツリーに追加した。

```
[root@389ds ~]# cat clients.ldif
dn: ou=clients,dc=kaneshiro,dc=localnet
objectClass: top
objectClass: posixGroup
objectClass: organizationalUnit
ou: clients
gidNumber: 1003
```

```
[root@389ds ~]# ldapadd -w （管理者 DN のパスワード） -D "cn=Manager,dc=localnet" -x -f clients.ldif
```

この時点での LDAP ツリー構造は以下のようになった（図 2）。

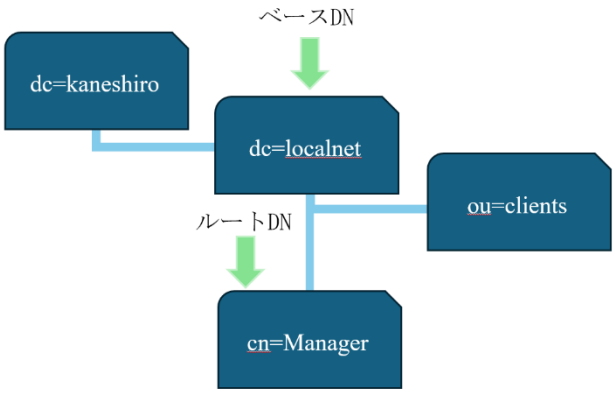

図 2. 作成された LDAP ツリー構造の概略図

3.3 Samba サーバの作成

Samba サーバのホスト名は”samba”とした。Samba サーバ用ソフト”samba”に加え、関連コマンドを含む”samba-client”を以下のコマンドにより導入した。

```
[root@samba ~]# dnf install samba samba-client
```

導入された samba のバージョンは 4.17.5 であった。また、LDAP サーバに登録されたユーザを Samba サーバに Samba ユーザとして認識させるために、sssd（system security service daemon）[6]を周辺パッケージと共に導入した。

```
[root@samba ~]# dnf install sssd sssd-client sssd-ldap sssd-utils sssd-dbus sssd-tools
```

導入された sssd のバージョンは 2.8.2 であった。

4. LDAP・Samba サーバ各種設定

4.1 通信を SSL 化した LDAP-Samba 連携

LDAP-Samba 間通信の SSL 化（LDAPS 化）を行うため、LDAP サーバの port636 を解放した。port636 は LDAPS が標準で使用するポート番号である。

```
[root@389ds ~]# firewall-cmd –add-port=636/tcp –permanent
[root@389ds ~]# firewall-cmd –reload
```

続いて、LDAP サーバにテスト用 LDAP ユーザ”testClient”の情報を追加するために、以下に示す ldif ファイルを作成した。

```
[root@389ds ~]# cat testClient.ldif
dn: uid=testClient,ou=clients,dc=kaneshiro,dc=localnet
objectClass: top
objectClass: posixAccount
objectClass: account
cn: testClient
uid: testClient
uidNumber: 1008
gidNumber: 1003
homeDirectory: /home/testClient
loginShell: /bin/bash
userPassword:: （ハッシュ化済みパスワード）
```

ldapadd コマンドを使い、このエントリを LDAP ツリーに追加した。その後、389DS を再起動した。

```
[root@389ds ~]# ldapadd -w (管理者 DN のパスワード) -x -D "cn=Manager,dc=localnet" -f testClient.ldif
[root@389ds ~]# systemctl restart dirsrv@389DS1.service
```

続いて Samba サーバへ移動し、/etc/samba/smb.conf を編集することで、Samba を LDAPS に対応させた[7, 8]。以下に LDAPS に対応させた smb.conf を掲載する。なお、192.168.56.0/24 はブリッジ接続された VirtualBox 上のローカルネットワークを、192.168.56.15 は LDAP サーバの IP アドレスを示す。

```
[root@samba ~]# cat /etc/samba/smb.conf
[global]
        server string = SAMBA
        netbios name = SAMBA
        security = user
        unix charset = UTF-8
        interfaces = 192.168.56.0/24 127.0.0.1
        ntlm auth = ntlmv2-only
        log level = 3
        log file = /var/log/samba/log.samba-ldap
```

```
        max log size = 1000
        ldap ssl = off
        passdb backend = ldapsam:ldaps://192.168.56.15
        ldap suffix = dc=kaneshiro,dc=localnet
        ldap group suffix = ou=clients
        ldap user suffix = uid=%S,ou=clients
        ldap admin dn = cn=Manager,dc=localnet
        ldap debug level = 1
        ldap passwd sync = yes
[homes]
        browsable = no
        writable = yes
        valid users = %U
[shared]
        path = /home/shared
        writable = yes
        valid users = @clients
        force group = clients
        force create mode = 664
        force directory mode = 775
```

また、Samba サーバの/etc/opnldao/ldap.conf 末尾に以下の文を加え、自己署名証明書による接続を許可した。

```
[root@samba ~]# tail -n 1 /etc/openldap/ldap.conf
TLS_REQCERT ALLOW
```

sssd の設定ファイル/etc/sssd/sssd.conf は導入時に自動作成されないため、管理者が手動で作る必要がある。以下に LDAPS に対応させた sssd.conf の雛型を掲載する[9]。

```
[root@samba ~]# cat /etc/sssd/sssd.conf
[sssd]
   debug_level = 0
   config_file_version = 2
   services = nss, sudo, pam, ssh
   domains = default
[domain/default]
   id_provider = ldap
   auth_provider = ldap
   ldap_uri = ldaps://192.168.56.15
   ldap_search_base = dc=kaneshiro,dc=localnet
   ldap_default_bind_dn = cn=Manager,dc=localnet
   ldap_default_authok = (平文パスワード)
   ldap_id_use_start_tls = True
   ldap_tls_reqcert = hard
   ldap_schema = rfc2307bis
   cache_credentials = True
   enumerate = True
   case_sensitive = false
[nss]
   filter_users = root
   filter_groups = root
[sudo]

[pam]
```

sssd.conf により規定される sssd が LDAP ツリー構造に接続するためにはバインドする dn（すなわち ldap_default_bind_dn）のパスワードが必要になる。ldap_default_authok の内容がそれに該当するが、平文パスワードを書き込むのはセキュリティ上不適切である。sssd-tools パッケージに含まれる sss_obfuscate コマンドを使い、このパスワードを難読化する。

```
[root@samba ~]# sss_obfuscate -d default
Enter password
Re-enter password:
```

これにより、難読化されたパスワードが sssd.conf に自動的に書き込まれる。

```
[root@samba ~]# grep “ldap_default_authok”
/etc/sssd/sssd.conf
ldap_default_authok_type = obfuscate_password
ldap_default_authok = （難読化されたパスワード）
```

sssd.conf のパーミッション、オーナーシップを適切に編集した後、authselect コマンドを実行し、認証に sssd を使うことを明示した。その後、設定を反映させるために各サービスを再起動した。

```
[root@samba ~]# chown root:root /etc/sssd/sssd.conf
[root@samba ~]# authselect select sssd –force
[root@samba ~]# systemctl restart smb sssd
```

この状態で pdbedit コマンドを実行し、登録された LDAP ユーザと同名のユーザ”testClient”を Samba サーバに登録する。

```
[root@samba ~]# pdbedit -a testClient
（中略）
new password:
retype new password:
```

pdbedit コマンドの実行に成功すると、LDAP サーバの”testClient”エントリに Samba 関連の属性が追加される。

```
dn: uid=testClient,ou=clients,dc=kaneshiro,dc=localnet
objectClass: top
objectClass: posixAccount
objectClass: account
objectClass: sambaSamAccount
cn: testClient
uid: testClient
uidNumber: 1008
gidNumber: 1003
homeDirectory: /home/testClient
loginShell: /bin/bash
userPassword:: （ハッシュ化されたパスワード）
sambaSID:S-1-5-21-3890852381-3112181165-
2220008927-1005
displayName: testClient
sambaNTPassword: （ハッシュ化されたパスワード）
（以下省略）
```

ここで、SambaSID とは Samba サーバ上でのユーザを一意に識別するためのセキュリティ識別子（SID）[10]のことであり、S-1-5-21-（数字 1）-（数字 2）-（数字 3）-（数字 4）といった決まった構造を取る。数字 1 から 3 はドメイン識別子であり、Samba サーバ毎に固有の値を取る。数字 4 はユーザ識別子であり、ユーザ毎に固有の値を取る。また、SambaNTPassword は

pdbedit コマンドにより入力されたパスワードを NTLM でハッシュ化した値である。

続いて、LDAP ツリー内にある”testClient”エントリの”homeDirectory”属性を確認し、Samba サーバ上の同じ Path に手動でホームディレクトリを作成した。その後、オーナーシップとパーミッションを設定した。

```
[root@samba ~]# mkdir /home/testClient
[root@samba ~]# chown testClient:clients
/home/testClient
[root@samba ~]# chmod 700 /home/testClient
```

これらの操作が完了したのち、作業内容を反映させるため、各サーバのサービスを再起動させた。

```
[root@samba ~]# systemctl restart
dirsrv@389DS1.service
[root@samba ~]# systemctl restart smb sssd
```

その後ホスト OS（Windows）からエクスプローラを起動し、ウィンドウ上部の窓（Path が記載される部分）に Samba サーバの IP アドレスを入力することで接続テストを行った。すると想定通り認証情報の入力を求められ、ユーザ名とパスワードを正しく入力することで、Samba サーバ上のフォルダへとアクセスすることができた。

4.2 ユーザのフォルダに対するアクセス制御

ユーザのアクセス制御を実現している箇所は、/etc/samba/smb.conf の以下の部分である[3]。

```
[root@samba ~]# cat /etc/samba/smb.conf
（中略）
（該当部分ここから）
[homes]
        browseable = no
        writable = yes
        valid uers = %U
[shared]
        path = /home/shared
        writable = yes
        valid users = @clients
        force group = clients
        force create mode = 664
        force directory mode = 775
（該当部分ここまで）
```

この状態でユーザ Samba サーバにアクセスすると、ユーザ自身のフォルダと共有フォルダのみが表示され、他のユーザのフォルダは表示されない。つまり、任意のユーザが自身のフォルダと共有フォルダ以外にはアクセスできない状況を実装できた。

4.3 自動履歴機能の実装

自動履歴機能を実装するには、履歴の元となるバックアップを取得しなくてはならない。これには様々なツールが利用できるが、今回は rsync[11]を利用することとした。Samba ユーザのホームディレクトリを/home/.snap 以下にバックアップし、30 日間でローテートするシェルスクリプト”shadow_copy.sh”を参考 URL[12]を基に作成し、Samba サーバの /usr/localn/samba/shadow_copy 以下に配置した。

```
[root@samba ~]# cd /usr/localn/samba/shadow_copy
[root@samba shadow_copy]# cat shadow_copy.sh
#!/bin/bash

#変数の定義
SambaDir=/home/testClient
BackupDir=/home/.snap
SnapDir=$(TZ=GMT                                    date
+@GMT-%Y.%m.%d-%H.%M.%S)
LastSnapFile=$BackupDir/last_snapshot
ROTATE_DATE=30
#バックアップ格納フォルダの作成
mkdir -p $BackupDir/$SnapDir

#バックアップの作成
if [ ! -e $LastSnapFile ]; then
   rsync -av $SAMBA_DIR $BackupDir/$SnapDir 1>
/dev/null
else
   LAST_SnapDir=`cat $LastSnapFile`
   rsync          -av          --delete          --link-
dest=$BackupDir/$LAST_SnapDir      $SAMBA_DIR
$BackupDir/$SnapDir 1> /dev/null
fi

echo $SnapDir > $LastSnapFile

#ローテート処理
LIST=`ls -1 $BackupDir | grep GMT`
DeleteSnapDirDay=`echo $s | cut -f2 -d"-" | tr -s "." "/"
`
DeleteSnapDirTime=`echo $s | cut -f3 -d"-" | tr -s "." ":"
`
DeleteSnapDir=`date       -d       "$DeleteSnapDirDay
$DeleteSnapDirTime" "+%s"`
Now=`date -d "$ROTATE_DATE days ago" "+%s"`
for s in $LIST
   do
        if [ $DeleteSnapDir -le $Now]; then
             rm -rf $BackupDir/$s
        fi
done
```

上記のシェルスクリプトを作成した後、/etc/samba/smb.conf を開き、以下の内容を追加した[12]。

```
[root@samba ~]# cat /etc/samba/smb.conf
[global]
（中略）
（追加箇所ここから）
vfs objects = shadow_copy2
shadow:snapdir = /home/.snap
shadow:basedir = /home
shadow:sort = desc
（追加箇所ここまで）
```

その後 samba を再起動し、シェルスクリプトを実行することでフォルダ/home/testClient のバックアップを作成した。ホスト OS（Windows 11）のエクスプ

ローラから Samba サーバ上の同フォルダへアクセスし、格納されているファイル”testClient.txt”について右クリックからプロパティを呼び出した。”以前のバージョン”タブを確認したところ、差分ファイルが作成されていることが確認できた。

5 実装に至らなかった理由

4 章 1 節において、以下のように Samba サーバで pdbedit コマンドを実行し、LDAP エントリ”testClient” に Samba 関連属性を登録したことを述べた。

```
# pdbedit -a testClient
（中略）
new password:
retype new password:
```

これにより登録される属性”SambaSID”は S-1-5-21-（数字 1）-（数字 2）-（数字 3）-（数字 4）といった構造を取り、数字 1 から 3 が Samba サーバを、数字 4 がユーザを一意に指定することも 4 章 1 節で既に述べた。この方式で困るのは、1 つの LDAP サーバに 2 つの Samba サーバを連携させた上で、両方の Samba サーバに接続したいユーザが存在する場合である（図 3）。

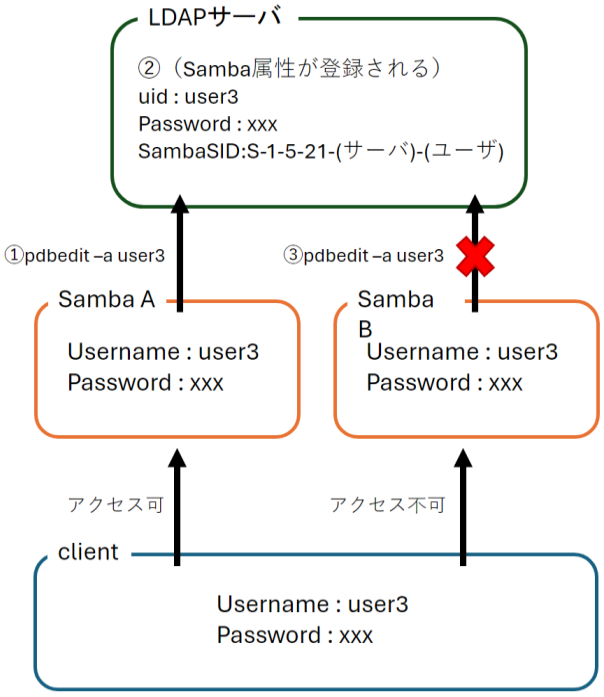

図 3. user3 が 2 つの Samba サーバ（Samba A と Samba B）に接続したい状況を表す概略図

SambaSID の値は LDAP エントリ毎に 1 つであるため、Samba A の SambaSID を先に登録してしまうと、Samba B の SambaSID は登録できない。

例えば、LDAP サーバに登録済みの user3 が、client 端末から Samba A と Samba B 両方に接続したいとする。まず Samba A で pdbedit -a を実行し（図 3 の①）、LDAP サーバに Samba A に関する Samba 属性を追加する。この時点で user3 の SambaSID には Samba A の値が登録される（②）。user3 は Samba B にも接続したいため、LDAP サーバに Samba B の SambaSID も登録する必要がある（③）。しかし、SambaSID の値は LDAP エントリ毎に 1 つであるため、それは不可能である。なお、検証環境で実際にこの状況を再現すると、コマンドの実行自体は問題なく行えるものの、SambaSID が上書きされることはなかった。

この状況を打破するにはどうすればよいかというと、LDAP サーバと連携せず、認証情報を Samba サーバに格納すればよい。それぞれの Samba サーバは独立しているため、複数の値を 1 つの属性に登録する状況は発生しない。複数の Samba にアクセスしたいユーザがいる場合は、LDAP との連携がかえって邪魔になってしまうわけである。

また、検証環境では LDAP サーバ側に認証情報を格納しているが、手元では pdbedit -a を実行しパスワードを入力するという、Samba サーバに認証情報を格納する場合と同じ操作を行っている。加えて、pdbedit -a コマンドの実行時 LDAP エントリに sambaNTPassword が追記されたことを 4 章 1 節で述べたが、Samba サーバ接続時の認証に用いるパスワードは実はこちらの値であり、LDAP の方で登録されたものではないことが後に判明した。つまり、情報の格納先が LDAP サーバであるだけであり、手元の操作、実際に行われている処理の両方において。Samba サーバに情報を置く場合と大差が無い。これでは、LDAP と Samba を連携させる旨味は感じられない。

このように、LDAP と Samba を連携させる意味が薄く、かえって不便になる場合も存在することが判明したため、今回検討したファイル共有システムは実装には至らなかった。

6 総括

現在分子研で運用している OpenLDAP と Windows WebClient を組み合わせたファイルシステムについてユーザから不満の声が上がっていることを受け、LDAP（389DS）と Samba による新しいファイルシステムの実装を試みた。実装を目指す中で、LDAPS 対応の Samba サーバの設定やユーザのアクセス制御、自動履歴機能の実装といった事柄について有益な知見が得られた。特定の状況で連携そのものが障害となることや、今回の検証内容では連携の意味そのものがないことが露見し、実装は見送られた。しかし、検討の過程で得られた LDAPS 化等のノウハウはどこかで役に立つものと考えている。

参考文献

[1] https://www.samba.org/
[2] https://www.virtualbox.org/wiki/Download_Old_Builds_7_0
[3] https://rockylinux.org/ja
[4] https://rockylinux.org/ja/download
[5] https://www.port389.org/
[6] https://access.redhat.com/documentation/jajp/red_hat_enterprise_linux/8/html/configuring_authentication_and_authorization_in_rhel/understanding-sssd-and-its-benefits_configuring-

authentication-and-authorization-in-rhel
[7] 高橋 基信､"サーバ構築の実例がわかる Samba [実践] 入門" 株式会社技術評論社、平成 28 年 4 月 5 日
[8] https://www.samba.gr.jp/project/translation/current/htmldocs/manpages/smb.conf.5.html
[9] https://qiita.com/mypaceshun/items/9c3b3f0ef9580c6d60ea
[10] https://learn.microsoft.com/ja-jp/windows-server/identity/ad-ds/manage/understand-security-identifiers
[11] https://rsync.samba.org
[12] https://postitx.blog.fc2.com/blog-entry-69.html