



IceWall SSO

Version 10.0

セッション ID 生成ルーチン 開発者マニュアル

日本ヒューレット・パッカード株式会社
2010 年 8 月

Printed in Japan

HP Part No. B2873-97015

Rev.100817A

ご注意

本書に記載されました内容は、予告なしに変更することがあります。

本書は内容について細心の注意をもって作成いたしましたが、万一ご不審な点や誤り、記載もれなど、お気づきの点がございましたら当社までお知らせください。

当社は、本書に記載されている誤り、あるいは備品、パフォーマンス、または本書の使用に関連して発生する偶然または必然的な損傷について責任は負いかねます。

本書の著作権は Hewlett-Packard Development Company, L.P. に帰属します。本書の一部あるいは全部についての無断複製、転載を禁じます。

本製品パッケージとして提供した本マニュアル、CD-ROM 等の媒体は本製品用だけにお使いください。

IceWall はヒューレット・パッカー社社の商標です。

Adobe は Adobe Systems Incorporated の米国及びその他の国における登録商標または商標です。

Java は、米国 Sun Microsystems, Inc. の米国およびその他の国における商標または登録商標です。

Microsoft および Windows は、米国 Microsoft Corporation の、米国、日本およびその他の国における登録商標または商標です。

Oracle は、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における登録商標です。

— 目次 —

1 はじめに	1
1.1 文中におけるバージョン表示記号について	1
2 セッション ID 生成ルーチンとは	2
3 開発の手順	3
3.1 仕様の決定	3
3.2 コーディング・ビルド	3
3.3 テスト・デバッグ	3
4 セッション ID 生成ルーチン仕様	5
IW_ExCreateSid	6
4.1 セッション ID 生成後の動作	8
5 注意事項	9
5.1 セッション ID 生成ルーチンの影響範囲	9
5.2 パフォーマンスの影響	9
5.3 メモリリーク	9
5.4 63 バイト長のセッション ID を生成する場合 10.0	9
5.5 サポートについて	9
6 制限事項	10
6.1 リンクする標準ライブラリのバージョンについて	10
6.2 互換性について	10
6.3 スレッドについて	10
6.4 Linux 版について	10
7 サンプルソース	11
7.1 ユーザー ID をセッション ID とする場合	11
8 参考	12
8.1 セッション ID 生成ルーチン開発キット	12
8.1.1 スケルトンソース (iwcreatesid.c)	12
8.1.2 ヘッダーファイル (iwcreatesid.h)	12
8.1.3 メイクファイル (HP-UX : Makefile)	13
8.1.4 メイクファイル (Linux : Makefile)	13

1 はじめに

本書では、認証モジュールに組み込むセッション ID 生成ルーチンの開発に必要な情報が記述してあります。

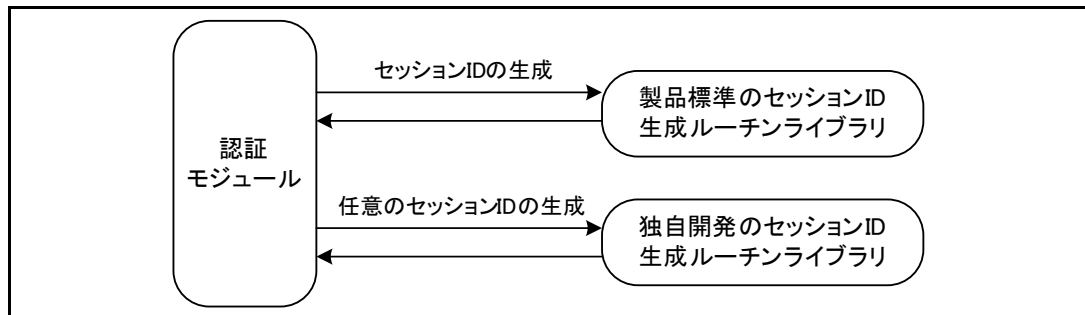
1.1 文中におけるバージョン表示記号について

記述中に付加されているバージョン表示記号は以下の意味を表します。

表示記号	意味
10.0	四角で囲まれているバージョンで追加された項目です。この場合、10.0 にて追加されたことを表します。
10.0	楕円で囲まれているバージョンで仕様変更もしくは機能追加された項目です。この場合、10.0 にて仕様変更または機能追加されたことを表します。

2 セッション ID 生成ルーチンとは

セッション ID 生成ルーチンとは、認証モジュールが認証時にセッション ID を生成する部分をライブラリ化し、任意のセッション ID の生成を可能とするものです。



セッション ID 生成ルーチンは Shared Library として提供されています。セッション ID の生成アルゴリズムを変更する場合は、開発の手順にしたがって、新たに Shared Library を作成する必要があります。

セッション ID 生成ルーチンでは以下の処理を行うことができます。

- ・任意のセッション ID の生成が可能

セッション ID に意味を持たせることや、他のセキュリティ製品や Web アプリケーションで有効となるセッション ID を生成することが可能です。

3 開発の手順

セッション ID 生成ルーチンの開発を行うための一般的な手順をご説明いたします。

3.1 仕様の決定

「5 注意事項」にしたがい、仕様を決定してください。

3.2 コーディング・ビルド

決定した仕様に基づき、セッション ID 生成ルーチンのソースコードをコーディングします。コーディング後に Shared Library としてビルドします。

コーディングは、開発キットに含まれるスケルトンソースを利用してください。
ビルドは開発キットに含まれる Makefile で行います。

```
$ cd /opt/icewall-ss0/developkit/certd/SidExit  
$ make -f Makefile
```

3.3 テスト・デバッグ

テストプログラム等により、作成された Shared Library のテストおよびデバッグを行います。テストが終了した後に、ライブラリを認証モジュールへ組み込み、動作テストを行います。

インストールは以下の手順で行います。

- (1) 認証モジュールが動作中の場合は、認証モジュールを停止コマンドで停止します。

```
$ /opt/icewall-ss0/certd/bin/end-cert
```

- (2) 標準でインストールされるセッション ID 生成ルーチンをバックアップします。

```
$ cd /opt/icewall-ss0/lib  
$ cp -p libCreateSidExit.sl libCreateSidExit.sl.bak
```

- (3) 作成したセッション ID 生成ルーチンをコピーします。

```
$ cp -p /opt/icewall-ss0/developkit/certd/SidExit/libCreateSidExit.sl /opt/icewall-ss0/lib
```

- (4) 認証モジュールを起動コマンドで起動します。

```
$ /opt/icewall-ss0/certd/bin/start-cert
```

4 セッション ID 生成ルーチン仕様

本章ではセッション ID 生成ルーチンの仕様を解説いたします。

- ・ 認証モジュールよりコールされるインターフェイス関数
- ・ セッション ID 生成後の動作

IW_ExCreateSid

インターフェイス関数の仕様は以下のとおりです。

書式 **int IW_ExCreateSid(EX_CS_REQ* req, char* sessionid, char* userid, char* password)**

引数 **req** : リクエスト構造体です。この構造体に認証モジュールの情報が格納されています。

sessionid : 認証モジュールで生成された標準のセッション ID (31 文字または 63 文字) が格納されます。この引数に本関数内で生成したセッション ID を格納します。認証が成功した場合は、ログアウトされるまで有効なセッション ID となります。セッション ID は **31 文字または 63 文字** で生成してください。 (10.0)

userid : ブラウザーよりログイン時に入力されたユーザー ID が格納されます。この引数は編集できません。

password : ブラウザーよりログイン時に入力されたパスワードが格納されます。この引数は編集できません。

戻り値 認証モジュールにセッション ID の生成結果を返します。

0 : セッション ID の生成に成功

0 以外 : セッション ID の生成に失敗

制限事項 本関数で渡される引数は sessionid 以外編集できません。これ以外の引数を編集した場合の動作は保証されません。

セッション ID は次の制約の範囲内で生成してください。

- ユニーク性の高いアルゴリズムであること
- セッション ID が 31 文字のときの使用可能文字は、0-9 (0x30 ~ 0x39)、A-Z (0x41 ~ 0x5a)、a-z (0x61 ~ 0x7a) のみです。63 文字のときの使用可能文字は、0-9 (0x30 ~ 0x39)、A-Z (0x41 ~ 0x5a)、a-z (0x61 ~ 0x7a)、- (マイナス) (0x2d)、_ (アンダースコア) (0x5f) のみです。記号や制御コードを使用した場合の動作は保証されません。 (10.0)

セッション ID がブラウザーへ通知される際には、認証モジュール識別キー 1 文字が追加されて 32 文字または 64 文字になります。追加される管理コードは、認証モジュール設定ファイル (cert.conf) の CERTUNIQUEKEY 項目で設定できます。また、生

成するセッション ID のバイト長は、認証モジュール設定ファイル (cert.conf) の SESSIONIDLEN の設定とあわせる必要があります。 **10.0**

備考 生成したセッション ID がシステム内でユニークではない場合は、認証モジュールは本関数を再度コールします。このとき、引数 `sessionid` には新たに生成した標準のセッション ID が格納されます。

認証モジュールで生成されたセッション ID の長さは `strlen(*sessionid)` で確認できます。

4.1 セッション ID 生成後の動作

セッション ID 生成ルーチンの処理結果によるセッション ID 生成後の動作を解説いたします。

- (1) インターフェイス関数の戻り値を 0 とした場合

生成されたセッション ID を使用して認証モジュールの認証処理を継続します。

- (2) インターフェイス関数の戻り値を 0 以外とした場合

認証モジュールのエラーログに次のメッセージが出力され、ブラウザーにシステムエラー画面が表示されます。

```
Fatal: IW_ExCreateSid Error. UserID=[ ユーザー ID] Result=[ インターフェイス関数の戻り値 ] [EC31002-20097]
```

- (3) 引数 sessionid に 31 文字または 63 文字以外のセッション ID を格納した場合

認証モジュールのエラーログに次のメッセージが出力され、ブラウザーにシステムエラー画面が表示されます。 (10.0)

```
Fatal: Bad SessionID. UserID=[ ユーザーID] IW_INFO=[ 生成したセッションID] [EC31003-20098]
```

- (4) 生成したセッション ID が重複した場合

認証モジュールのアクセスログに次のメッセージが出力され、再度セッション ID 生成ルーチンがコールされます。COOKIERETRY 設定項目の回数分セッション ID 生成をおこなってもユニークなセッション ID が生成できなかった場合、ブラウザーにシステムエラー画面が表示されます。

```
Fatal: Ununique Cookie. UserID=[ ユーザー ID] IW_INFO=[ 生成したセッションID] [AC31004-25111]
```

- (5) 生成したセッション ID が認証モジュールで生成するセッション ID のバイト長と一致しない場合

セッション ID 生成ルーチン内で生成したセッション ID が認証モジュールで生成するセッション ID のバイト長と一致しない場合は、次のメッセージが出力され、ブラウザーにセッション ID 生成エラー画面が表示されます。

```
Fatal: Bad SessionID. UserID=[ ユーザーID] IW_INFO=[ 生成したセッションID] [EC31003-20098]
```

5 注意事項

セッション ID 生成ルーチンを作成および導入する上で注意しなければならない点をご紹介します。

5.1 セッション ID 生成ルーチンの影響範囲

セッション ID 生成ルーチン内で実行される処理は、認証モジュールの内部処理と密接な関係にあるため、セッション ID 生成ルーチン内でシステム的なエラーが発生した場合、認証モジュールがダウンする可能性があります。作成されたライブラリのテストを十分に行った上で、認証モジュールへ組み込んでください。

また、セッション ID 生成ルーチン内で `exit()` をコールすると、認証モジュールの実行プロセスが終了してしまうため、記述しないでください。

5.2 パフォーマンスの影響

セッション ID 生成ルーチン内で処理を追加する場合、追加した処理を実行するため、パフォーマンスが低下します。パフォーマンスへの影響を考慮して生成するセッション ID の生成ロジックの決定とコーディングを行ってください。

5.3 メモリリーク

セッション ID 生成ルーチン内で独自に取得したメモリ領域については、必ず解放するようにしてください。解放しない場合はメモリリークの原因となります。

5.4 63 バイト長のセッション ID を生成する場合 10.0

63 バイト長でセッション ID を生成する場合、認証モジュールへのリクエストが ICP 2.0 を使用しており、認証モジュール設定ファイル (`cert.conf`) の `SESSIONIDLEN` 項目の設定を 64 にしている必要があります。

5.5 サポートについて

製品のサポートは、標準で添付されるセッション ID 生成ルーチンを使用した場合にのみ受けることができます。お客さまが作成されたセッション ID 生成ルーチンが導入されている状態ではサポートを受けることができません。標準添付のライブラリはサポート時に必要となりますので、必ずバックアップを行ってください。

6 制限事項

セッション ID 生成ルーチンを作成する場合には、以下の制限事項があることに注意してください。以下の制限内で作成されない場合には、認証モジュールが実行できない、または動作が不安定になる原因となります。

6.1 リンクする標準ライブラリのバージョンについて

認証モジュールはダイナミックにライブラリを使用して動作するように作成されています。

ただし、セッション ID 生成ルーチンにアーカイブライブラリをリンクしている場合には、セッション ID 生成ルーチンにリンクされているライブラリが利用されることとなるため、バージョンの違いによる不整合を招く可能性があります。(OS への Patch を適用しても解決しない等のトラブルにもなる可能性があります)

注) 特に Oracle 等、インストール環境によって生成されるライブラリ構成が変わる (libclntsh.sl の構成が変わる) ものをセッション ID 生成ルーチンで利用する場合にはリンクされているライブラリの状態について十分注意する必要があります。

6.2 互換性について

セッション ID 生成ルーチンを作成したバージョン以外のバージョンで使用する場合にはリビルドする必要があります。このため、IceWall SSO のバージョンアップを行った場合には、セッション ID 生成ルーチンをリビルドするようにしてください。

6.3 スレッドについて

認証モジュールは内部処理でスレッドを使用しており、セッション ID 生成ルーチンはスレッドよりコールされるため、標準ライブラリ関数は必ずスレッドセーフ版を使用するようにしてください。また、ユーザー関数を作成する場合は、スレッドセーフとして動作する必要があります。

6.4 Linux 版について

コンパイルオプションには必ず「-m64」を付加してください。

認証モジュールのスレッドは「Linux Threads」及び「Native POSIX Library Thread」が使用できます。

7 サンプルソース

以下にセッション ID 生成ルーチンのサンプルソースをご紹介します。

7.1 ユーザー ID をセッション ID とする場合

- ・ユーザー ID の 31 桁までをセッション ID とします。
- ・ユーザー ID が 31 桁より小さい場合、ゼロキャラクターでパディングします。

```
#include <string.h>

#include "iwcreatesid.h"

int IW_ExCreateSid ( req, sessionid, userid, password )
EX_CS_REQ *req;
char    *sessionid;
char    *userid;
char    *password;
{
    int length;

    strncpy( sessionid, userid, 31 );
    length = strlen( userid );

    if ( length < 31 ) {
        memset ( ( sessionid + length ), 0x30, ( 31- length ) );
    }

    return ( 0 );
}
```

8 参考

標準でインストールされるセッション ID 生成ルーチン開発キットの内容です。
開発キットの構成は以下のディレクトリおよびファイルとなっています。

ディレクトリおよびファイル					内容
/opt/icewall-ssso	/developkit	/certd	/SidExit	/iwcreatesid.c	セッション ID 生成ルーチン開発 キット
				/iwcreatesid.h	
				/Makefile	

なお、この開発キットで作成されるライブラリ名は標準でインストールされるライブラリ名と同じものになりますので、作成および導入する際には、必ずバックアップを行ってください。

8.1 セッション ID 生成ルーチン開発キット

標準でインストールされるセッション ID 生成ルーチン開発キットの各ファイルの内容をご紹介します。

8.1.1 スケルトンソース (iwcreatesid.c)

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

#include "iwcreatesid.h"

/*-----*/
int IW_ExCreateSid ( req, sessionid, userid, password )
EX_CS_REQ *req;
char *sessionid;
char *userid;
char *password;
{
    return( 0 );
}
```

8.1.2 ヘッダーファイル (iwcreatesid.h)

```
/*-----*/
/* Create Sid Exit */
/*-----*/
#ifndef IWCREATESID_H
#define IWCREATESID_H

/*-----*/
```

```
/* API Structure */
/*-----*/
typedef struct ex_cs_req EX_CS_REQ;

#endif /* #ifndef IWCREATESID_H */
```

8.1.3 メイクファイル (HP-UX : Makefile)

```
CC      = cc

CCFLAGS  = -D_UNIX -D_HPUX_SOURCE -D_POSIX_C_SOURCE=199506L -
Aa +e -D_FILE_OFFSET_BITS=64 -z +DD64 +z

LIBS     =

INCLUDE  = -I./

MAKEFILE = Makefile

OBJS     = iwcreatesid.o

PROGRAM  = CreateSidExit

SRCS     = iwcreatesid.c

all:      $(PROGRAM)

$(PROGRAM): $(OBJS)
            ld -b -z -o lib$(PROGRAM).sl $(OBJS) $(LIBS)

.c.o:
            $(CC) $(CCFLAGS) $(INCLUDE) -c $(SRCS)

clean:
            rm -f $(OBJS) lib$(PROGRAM).sl core
```

8.1.4 メイクファイル (Linux : Makefile)

```
CC      = gcc

CCFLAGS  = -m64 -fPIC -DLinux -D_FILE_OFFSET_BITS=64 -
D_LARGEFILE_SOURCE -D_GNU_SOURCE

LDFLAGS  = -m64 -fPIC

LIBS     =

INCLUDE  = -I./
```

```
MAKEFILE    = Makefile

OBJS        = iwcreatesid.o

PROGRAM     = CreateSidExit

SRCS        = iwcreatesid.c

all:                $(PROGRAM)

$(PROGRAM):        $(OBJS)
gcc -shared $(LDFLAGS) -o lib$(PROGRAM).sl $(OBJS)
$(LIBS)

.c.o:
$(CC) $(CCFLAGS) $(INCLUDE) -c $(SRCS)

clean:
rm -f $(OBJS) lib$(PROGRAM).sl core
```