Solaris™ Security Toolkit 4.2
リファレンスマニュアル

Sun Microsystems, Inc.
www.sun.com

Part No. 819-3793-10
2005 年 7 月, Revision A

コメントの送付: http://www.sun.com/hwdocs/feedback

原典: *Solaris Security Toolkit 4.2 Reference Manual*
Part No: 819-1503-10
Revision A

目次

表目次

コード例

はじめに

この『Solaris™ Security Toolkit 4.2 リファレンスマニュアル』では、Solaris Security Toolkit ソフトウェアの特性を理解および使用するための情報を説明します。このマニュアルの主な対象読者は、Solaris™ Operating System (OS) バージョン 2.51 から 10 のセキュリティーを確保するために Solaris Security Toolkit ソフトウェアを使用するユーザーです。具体的には、管理者、コンサルタント、および Sun のシステムの新規配備または配備済みシステムのセキュリティー確保を担当するユーザーです。ここに記載されている説明は、このソフトウェアを JumpStart™ モードまたはスタンドアロンモードのいずれかで使用する場合に適用されます。

次に、このガイドで使用されている重要な用語を示します。

- 強化 – Solaris OS の構成を変更してシステムのセキュリティーを向上させること。
- 監査 – システムの構成が事前に設定されたセキュリティープロファイルに従っているかどうかを調べること。
- スコアリング – 監査の実行時に発見された障害の数を数えること。どのような障害も検出されていない場合、この数は 0 になります。Solaris Security Toolkit は、障害が検出されるたびにこの数 (脆弱性値とも言う) を 1 ずつ増やします。

お読みになる前に

このマニュアルの読者は、Solaris™ の サン認定システム管理者またはサン認定ネットワーク管理者であることが必要です。また、標準ネットワークプロトコルおよびトポロジについて理解していることも必要です。

このマニュアルは、セキュリティーについてさまざまなレベルの知識や経験を持つユーザーに役立つように作成されているため、それぞれの知識や経験に合わせて適宜活用してください。

マニュアルの構成

このマニュアルにはソフトウェアコンポーネントに関する参考情報が記載されており、次の章で構成されています。

第 **1** 章では、**Solaris 10 OS** で **Solaris Security Toolkit 4.2** ソフトウェアを使用する方法を説明します。

第 2 章では、フレームワーク関数の使用、追加、変更、および削除について説明します。フレームワーク関数を使用すると、ソースコードを変更せずに Solaris Security Toolkit ソフトウェアの動作を柔軟に変更することができます。

第 **3** 章では、Solaris Security Toolkit ソフトウェアに含まれているファイルテンプレートの使用、変更、およびカスタマイズについて説明します。

第 **4** 章では、ドライバの使用、追加、変更、および削除について説明します。この章で取り上げるドライバは、Solaris OS システムを強化、最小化、および監査するために、Solaris Security Toolkit ソフトウェアで使用されるドライバです。

第 **5** 章では、終了スクリプトの使用、追加、変更、および削除について説明します。この章で取り上げるスクリプトは、Solaris OS システムを強化および最小化するために Solaris Security Toolkit ソフトウェアで使用されるスクリプトです。

第 **6** 章では、監査スクリプトの使用、追加、変更、および削除について説明します。

第 **7** 章では、環境変数の使用について説明します。この章で取り上げる変数は、Solaris Security Toolkit ソフトウェアで使用されるすべての変数です。また、これらの変数値をカスタマイズする際のヒントとテクニックについても説明します。

UNIX コマンド

このマニュアルには、システムの停止、システムの起動、およびデバイスの構成などに使用する基本的な UNIX® コマンドと操作手順に関する説明は含まれていない可能性があります。これらについては、以下を参照してください。

- 使用しているシステムに付属のソフトウェアマニュアル
- 下記にある Solaris™ オペレーティングシステムのマニュアル
  `http://docs.sun.com`

シェルプロンプトについて

シェル	プロンプト
UNIX の C シェル	*machine_name*%
UNIX の Bourne シェルと Korn シェル	$
スーパーユーザー (シェルの種類を問わない)	#

書体と記号について

書体または記号[1]	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例。	.login ファイルを編集します。 ls -a を実行します。 % You have mail.
AaBbCc123	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して表します。	マシン名% **su** Password:
AaBbCc123	コマンド行の可変部分。実際の名前や値と置き換えてください。	rm *filename* と入力します。
『 』	参照する書名を示します。	『Solaris ユーザーマニュアル』
「 」	参照する章、節、または、強調する語を示します。	第 6 章「データの管理」を参照。 この操作ができるのは「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。	% **grep '^#define \ XV_VERSION_STRING'**

1 使用しているブラウザにより、これらの設定と異なって表示される場合があります。

ハードウェアモデル用の一般的な用語の使用法

Sun Fire™ ハイエンドシステムとは、次のモデル番号を指します。

- E25K
- E20K
- 15K
- 12K

Sun Fire ミッドレンジシステムとは、次のモデル番号を指します。

- E6900
- E4900
- 6800
- 4810
- 4800
- 3800

Sun Fire エントリレベルミッドレンジシステムとは、次のモデル番号を指します。

- E2900
- Netra 1280
- V1280
- V890
- V880
- V490
- V480

サポートされるハードウェアシステム

Solaris Security Toolkit 4.2 ソフトウェアは、SPARC® (64 ビットのみ)、および x86 システムをサポートします。

サポートされる Solaris OS のバージョン

Sun では、Solaris Security Toolkit ソフトウェアを Solaris 8、Solaris 9、および Solaris 10 オペレーティングシステムで使用する場合にのみサポートを提供しています。

注 – Solaris Security Toolkit 4.2 ソフトウェアの場合、Solaris 10 を使用できるのは Sun Fire ハイエンドシステムのドメイン上だけであり、システムコントローラ (SC) 上では使用できません。

Solaris Security Toolkit ソフトウェアは Solaris 2.5.1、Solaris 2.6、および Solaris 7 オペレーティングシステムで使用することもできますが、これらのオペレーティングシステムで使用する場合、サンではサポートを提供していません。

Solaris Security Toolkit ソフトウェアは、インストールされている Solaris オペレーティングシステムのバージョンを自動的に検出し、そのバージョンに合わせて適切なタスクを実行します。

このマニュアル全体の例では、スクリプトが OS のバージョンをチェックする場合、スクリプトは、Solaris OS のバージョンである 2.*x*、7、8、9、または 10 ではなく、SunOS™ のバージョンである 5.*x* をチェックします。表 P-1 に、SunOS のバージョンと Solaris OS のバージョンの関係を示します。

表 P-1 SunOS のバージョンと Solaris OS のバージョンの相関関係

SunOS のバージョン	Solaris OS のバージョン
5.5.1	2.5.1
5.6	2.6
5.7	7
5.8	8
5.9	9
5.10	10

サポートされる SMS のバージョン

System Management Services (SMS) を使用して Sun Fire ハイエンドシステム上のシステムコントローラ (SC) を稼働させている場合は、すべての Solaris 8 および Solaris 9 OS バージョンで SMS バージョン 1.4、1.4.1、および 1.5 と Solaris Security Toolkit 4.2 ソフトウェアの併用がサポートされます。Solaris 10 OS 上で Solaris Security Toolkit 4.2 ソフトウェアによりサポートされる SMS のバージョンはありません。

注 – Solaris Security Toolkit 4.2 ソフトウェアの場合、Solaris 10 を使用できるのはドメイン上だけであり、システムコントローラ (SC) 上では使用できません。

関連マニュアル

オンラインのマニュアルは次の URL で参照できます。

```
http://www.sun.com/products-n-solutions/hardware/docs/
Software/enterprise_computing/systems_management/sst/index.html
```

用途	タイトル	Part No	形式	場所
補正情報	『Solaris Security Toolkit 4.2 ご使用にあたって』	819-3796-10	PDF HTML	オンライン
管理者マニュアル	『Solaris Security Toolkit 4.2 管理マニュアル』	819-3789-10	PDF HTML	オンライン
マニュアルページ	『Solaris Security Toolkit 4.2 マニュアルページガイド』	819-3794-10	PDF	オンライン

マニュアル、サポート、およびトレーニング

Sun のサービス	URL	説明
マニュアル	http://jp.sun.com/documentation/	PDF と HTML マニュアルをダウンロードする、印刷マニュアルを注文する
サポートおよびトレーニング	http://jp.sun.com/supportraining/	テクニカルサポートを受ける、パッチをダウンロードする、Sun のコースについて情報を入手する

Sun 以外の Web サイト

このマニュアルで紹介する Sun 以外の Web サイトが使用可能かどうかについては、Sun は責任を負いません。このようなサイトやリソース上、またはこれらを経由して利用できるコンテンツ、広告、製品、またはその他の資料についても、Sun は保証しておらず、法的責任を負いません。また、このようなサイトやリソース上、またはこれらを経由して利用できるコンテンツ、商品、サービスの使用や、それらへの依存に関連して発生した実際の損害や損失、またはその申し立てについても、Sun は一切の責任を負いません。

コメントをお寄せください

マニュアルの品質改善のため、お客様からのご意見およびご要望をお待ちしております。コメントは下記よりお送りください。

http://www.sun.com/hwdocs/feedback

ご意見をお寄せいただく際には、下記のタイトルと Part No. を記載してください。

『Solaris Security Toolkit 4.2 リファレンスマニュアル』、Part No. 819-3793-10

第1章

Solaris 10 オペレーティングシステムのサポートの概要

Solaris Security Toolkit 4.2 ソフトウェアリリースの主な目的の 1 つは、Solaris 10 オペレーティングシステム (OS) をサポートすることです。Solaris Security Toolkit 4.2 ソフトウェアは、サービス管理機能 (SMF)、TCP ラッパー、IP フィルタなど、Solaris 10 OS の新しいセキュリティー機能に対応しています。新機能の詳細については、『Solaris Security Toolkit 4.2 ご使用にあたって』を参照してください。

Solaris Security Toolkit 4.2 ソフトウェアを使用して、以前のバージョンと同様の方法でシステムのセキュリティーを強化および監査できます。また、Solaris Security Toolkit 4.2 は以前のバージョンと同じように、JumpStart モードとスタンドアロンモードのいずれかで使用できます。

Solaris Security Toolkit ソフトウェアと Perl の併用

Solaris 10 OS には Perl (Practical Extraction and Report Language) が付属しています。Solaris 10 OS で使用するスクリプトを作成する場合、JumpStart モードでもスクリプトに Perl を使用できます。Solaris 10 OS よりも古いバージョンの Solaris OS では JumpStart 中には Perl が使えなかったり、Solaris OS のディストリビューションに Perl が含まれていなかったりする場合があります。Perl を必要とするスクリプトを書く前に、対象環境で Perl が使用できることを確認してください。セキュリティー意識の高いユーザーはシステムから Perl を削除していることが多いため、Perl が削除されている可能性があることも知っておく必要があります。

システムに Perl がインストールされている場合、Solaris Security Toolkit は `set-flexible-crypt.aud` スクリプト (218 ページの「set-flexible-crypt.aud」を参照) によって実行される監査中に Perl を使おうとします。システムに Perl がインストールされていない場合、スクリプトはエラーを出力します。

Solaris 10 OS 上の SMF とレガシーサービス

リストに入れて有効または無効にすることができるインターネットサービスデーモン (inetd) によって制御される一部のサービスは、サービス管理機能 (SMF) に変換され、障害管理リソース識別子 (FMRI) を使用しますが、inetd によって制御されるサービスにはサービス管理機能 (SMF) に変換されないものもあります。

- **SMF** 対応サービス – 有効または無効にする inetd によって制御される SMF 対応サービスのリストを作成する場合、JASS_SVCS_ENABLE または JASS_SVCS_DISABLE を使用します。JASS_SVCS_DISABLE スクリプトは、SMF 対応でシステムにインストールされ、リスト上にあるすべてのサービスを無効にします。表 1-1 に、SMF 対応の Solaris Security Toolkit スクリプトを示します。

注 – SMF 対応サービスのリストは Solaris 10 オペレーティングシステムに対してのみ有効です。

- レガシーサービス – 有効または無効にする inetd によって制御されるレガシー (つまり変換されていない) サービスのリストを作成する場合は、ツールキットの旧バージョンで使用していたのと同じ方法で、JASS_SVCS_ENABLE または JASS_SVCS_DISABLE を使用することができます。変換されないため、SMF がレガシーサービスと認識する Solaris Security Toolkit スクリプトを表 1-2 に示します。詳細は、278 ページの「JASS_SVCS_DISABLE」および 280 ページの「JASS_SVCS_ENABLE」を参照してください。

Solaris 10 オペレーティングシステムを使用している場合、JASS_SVCS_DISABLE スクリプトは、JASS_SVCS_DISABLE リスト上のすべてのサービスを (inetd.conf ファイル内にある場合) 無効にします。このため、あるサービスが inetd の下で Solaris 9 オペレーティングシステムに対しては有効であったが、Solaris 10 オペレーティングシステムに対して inetd.conf ファイルを使用しなくなった場合、JASS_SVCS_DISABLE 環境変数を変更してもそのサービスには何の変更も加えられません。

JASS_SVCS_ENABLE と JASS_SVCS_DISABLE のいずれかの環境変数にシステム上には存在しない FMRI または inetd サービス名が含まれている場合、Solaris Security Toolkit は警告メッセージを出します。

SMF 対応サービスインタフェースを使用するスクリプト

表 1-1 に、SMF 対応サービスを使用する Solaris Security Toolkit スクリプト、その Fault Management Resource Identifier (FMRI) および Solaris 9 OS に使用される起動/停止スクリプトを示します。

表 1-1 SMF 対応サービスインタフェースを使用する Solaris Security Toolkit スクリプト

スクリプト名	Fault Management Resource Identifier (FMRI)	Solaris 9 OS 用の起動/停止スクリプト
disable-apache2[1]	svc:/network/http:apache2	なし
disable-automount	svc:/system/filesystem/autofs:default	/etc/rc2.d/S74autofs
disable-dhcpd	svc:/network/dhcp-server:default	/etc/rc3.d/S24dhcp
disable-kdc	svc:/network/security/krb5kdc:default	/etc/rd3.d/S13kdc.master /etc/rd3.d/S14kdc
disable-ldap-client	svc:/network/ldap/client:default	/etc/rc2.d/S71dap.client
disable-lp	svc:/application/print/server:default svc:/application/print/ipp-listener:default svc:/application/print/rfc1179:default	/etc/rc2.d/S80lp
disable-named	svc:/network/dns/server:default	/etc/named.boot
disable-nfs-client	svc:/network/nfs/client:default svc:/network/nfs/status:default svc:/network/nfs/nlocmgr:default	/etc/rc2.d/S73nfs.client
disable-nfs-server	svc:/network/nfs/server:default	/etc/rc3.d/S15nfs
disable-power-mgmt	svc:/system/power:default	/etc/rc2.d/S85power
disable-rpc	svc:/network/rpc/bind:default svc:/network/rpc/keyserv:default	/etc/rc2.d/S71rpc
disable-sendmail	svc:/network/smtp/sendmail:default	/etc/rc2.d/S99sendmail
disable-slp	svc:/network/slp:default	/etc/rc2.d/S72slpd
disable-spc	svc:/application/print/cleanup:default	/etc/rc2.d/S80spc

表 1-1　SMF 対応サービスインタフェースを使用する Solaris Security Toolkit スクリプト (続き)

スクリプト名	Fault Management Resource Identifier (FMRI)	Solaris 9 OS 用の起動/停止スクリプト
disable-ssh-root-login	svc:/network/ssh:default	pkginfo -q -r SUNWsshdr を使用
disable-uucp	svc:/network/uucp:default	/etc/rc2.d/S70uucp
enable-ftpaccess	svc:/network/ftp:default	/etc/inet/inetd.conf
enable-inetd-syslog	svc:/network/inetd:default	/etc/default/inetd
enable-tcpwrappers	svc:/network/inetd:default	/etc/default/inetd
install-ftpusers	svc:/network/ftp:default	pkginfo -q -R SUNWftpr を使用
set-banner-ftpd	svc:/network/ftp:default	pkginfo -q -R SUNWsshdr を使用
set-banner-sshd	svc:/network/ssh:default	pkginfo -q -R SUNWftpr を使用
set-ftpd-unmask	svc:/network/ftp:default	pkginfo -q -r SUNWftpr を使用

1 Solaris 10 専用

SMF がレガシーサービスと認識するスクリプト

表 1-2 に、SMF 対応ではないが、SMF がレガシーサービスと認識する Solaris Security Toolkit スクリプトを示します。レガシーサービスは FMRI 形式で表すことができますが、SMF はそれらを有効/無効にすることはできません。

表 1-2　SMF がレガシーサービスと認識する Solaris Security Toolkit スクリプト

スクリプト名	Fault Management Resource Identifier (FMRI)
disable-apache	lrc:/etc/rc3_d/S50apache
disable-appserv	lrc:/etc/rc2_d/S84appserv
disable-autoinst	lrc:/etc/rc2_d/S72autoinstall
disable-directory	lrc:/etc/rc2_d/S72directory

表 1-2　SMF がレガシーサービスと認識する Solaris Security Toolkit スクリプト (続き)

スクリプト名	Fault Management Resource Identifier (FMRI)
disable-dmi	lrc:/etc/rc3_d/S77dmi
disable-dtlogin	lrc:/etc/rc2_d/S99dtlogin
disable-IIim	lrc:/etc/rc2_d/S95IIim
disable-mipagent	lrc:/etc/rc3_d/S80mipagent
disable-ppp	lrc:/etc/rc2_d/S47pppd
disable-preserve	lrc:/etc/rc2_d/S89PRESERVE
disable-samba	lrc:/etc/rc3_d/S90samba
disable-snmp	lrc:/etc/rc3_d/S76snmpdx
disable-uucp	lrc:/etc/rc2_d/S70uucp
disable-vold	lrc:/etc/rc3_d/S81volmgt
disable-wbem	lrc:/etc/rc2_d/S90wbem
set-banner-dtlogin	lrc:/etc/rc2_d/S99dtlogin

Solaris Security Toolkit 4.2 リリース用の新しいスクリプト

次に、Solaris Security Toolkit 4.2 リリース用の新しいスクリプトを示します。

- disable-apache2.{fin|aud}
- disable-appserv.{fin|aud}
- disable-IIim.{fin|aud}
- disable-routing.{fin|aud}
- enable-account-lockout.{fin|aud}
- enable-bart.{fin|aud}
- enable-ipfilter.{fin|aud}
- enable-password-history.{fin|aud}
- set-root-home-dir.{fin|aud}
- set-strict-password-checks.{fin|aud}

終了スクリプト (.fin) の機能は第 5 章で説明されています。監査スクリプト (.aud) の機能は第 6 章で説明されています。

Solaris 10 OS では使用されないスクリプト

表 1-3 に、Solaris 10 オペレーティングシステムを強化する際には使用されない Solaris Security Toolkit スクリプトの一覧を示します。

表 1-3 Solaris 10 OS では使用されない Solaris Security Toolkit スクリプト

スクリプト名	適用可能なオペレーティングシステム
`disable-ab2`	Solaris 2.5.1 ～ 8
`disable-aspp`	Solaris 2.5.1 ～ 8
`disable-picld`	Solaris 8 および 9
`install-fix-modes`	Solaris 2.5.1 ～ 9
`install-newaliases`	Solaris 2.5.1 ～ 8
`install-openssh`	Solaris 2.5.1 ～ 8
`install-sadmind-options`	Solaris 2.5.1 ～ 9
`install-strong-permissions`	Solaris 2.5.1 ～ 9
`remove-unneeded-accounts`	Solaris 2.5.1 ～ 9

Solaris 10 OS では使用されない環境変数

次の環境変数は、Solaris 10 オペレーティングシステムでは使用されません。

- `JASS_ISA_CAPABILITY` (*Solaris Security Toolkit 4.2* ソフトウェアから削除)
- `JASS_DISABLE_MODE`

Solaris 10 OS のゾーンの使用法

Solaris Security Toolkit 4.2 ソフトウェアは、Solaris 10 OS を使用するシステムのゾーン、つまり Sun Network One (N1) Grid コンテナの強化に使用できます。Solaris 10 のゾーンにおけるすべての Solaris Security Toolkit プロファイル (強化、監査、および元に戻す) 機能は、大部分において非ゾーンシステムと同じです。この節では、相違点を説明しています。

大域ゾーンおよび非大域ゾーンの強化における順序の重要性

非大域ゾーン (NGZ) がインストールされる前に大域ゾーンが強化されると、Solaris Security Toolkit 4.2 ソフトウェアによって行われた変更の一部が新しいゾーンに適用されますが、その他の多くの変更は適用されません。新しく作成されるゾーンが正しくセキュリティー保護されるように、ゾーンのインストール直後に、強化モードと監査モードの両方で Solaris Security Toolkit 4.2 ソフトウェアを適用する必要があります。非大域ゾーンがインストールされると、大域ゾーンにおける強化と強化解除は NGZ に影響せず、また逆に NGZ における強化と強化解除も大域ゾーンに影響しません。

非大域ゾーン内からの非大域ゾーンの強化

注意 – セキュリティー上のリスクがあるため、非大域ゾーンファイルシステムの外部から非大域ゾーンファイルシステムには決してアクセスしないでください。非大域ゾーン内では危険ではないパスが、大域ゾーンでは危険になる場合があります。たとえば、非大域ゾーンの管理者は `/etc/shadow` ファイルを `../../../shadow` ファイルにリンクできます。非大域ゾーンの内部では、このような操作は無害ですが、`/opt/testzone/etc/shadow` パスを使用しての大域ゾーンからのファイルに対する編集は、大域ゾーンの `/etc/passwd` ファイルを編集することになります。もう一度繰り返しますが、非大域ゾーンは (このゾーンにログインしていない限り) 決して強化したり、元に戻したり、クリーン化したり、監査したりしないでください。

Solaris Security Toolkit 4.2 のインストール場所が標準的な /opt/SUNWjass ディレクトリである場合、Solaris 10 OS の zlogin(1) コマンドを使用してゾーンにログイン、つまりゾーンに入って Solaris Security Toolkit を実行することで、そのゾーンを強化できます。

コード例 1-1　非大域ゾーンの強化

```
# zlogin myzone /opt/SUNWjass/bin/jass-execute -d my.driver
```

変数 *myzone* は非大域ゾーンで、変数 *my.driver* は使用するドライバの名前です。

非大域ゾーンに関連しない一部のスクリプト

/etc/system を使用してカーネルパラメタを変更するものなど、一部の Solaris Security Toolkit スクリプトは非大域ゾーンに関係しません。これらのスクリプトを非大域ゾーンで実行すると、スクリプトは、これらは [NOTE] としては非大域ゾーンには必要ないという事実を記録します。

独自のスクリプトを書く場合、logNotGlobalZone 関数 (28 ページの「logNotGlobalZone」を参照) を使用して標準的な方法でそのようなメッセージを発行することができます。ユーザーが、Solaris Security Toolkit スクリプトの非大域ゾーン内にいるかどうかをテストするには、Solaris Security Toolkit 4.2 の環境変数 JASS_ZONE_NAME をチェックして、その中に global が含まれているかどうかを確認します。Solaris 10 OS よりも古いバージョンの OS では、この変数は global に設定されています。この変数についての詳細は、258 ページの「JASS_ZONE_NAME」を参照してください。

非大域ゾーンの監査は大域ゾーンの監査からは分離、区別されている

実行中のプロセス、インストール済みソフトウェア、および非大域ゾーンの構成は、大域ゾーンからは独立して監査されます。たとえば、実行中の未認証のプロセスを検出した NGZ の監査は NGZ の監査障害を引き起こしますが、大域ゾーンの監査障害を引き起こすことはありません。同様に、大域ゾーンが監査され、セキュリティー違反が検出された場合、大域ゾーンのセキュリティー違反が生成され、NGZ の違反は生成されません。

大域ゾーンと非大域ゾーンの監査の間でオーバーラップが生じるのは、大域ゾーンの BART レビューの時点のみです。NGZ のファイルシステムは大域ゾーンにマウントされ、Solaris Security Toolkit に含まれる BART マニフェストファイルによりレビューされる場合があります。大域ゾーンから NGZ ファイルシステムをレビューす

る場合、NGZ に関連するセキュリティー違反が大域ゾーンで報告される場合があります。このような状況を回避するには、大域ゾーン上でマウントされる NGZ ファイルシステムが BART マニフェストファイルから除外されるようにします。

ゾーンに対応する終了および監査スクリプト

操作権限が不十分であるためゾーンでは実行されないツールキットスクリプトに関しては、それらが環境変数 JASS_ZONE_NAME (258 ページの「JASS_ZONE_NAME」を参照) を使用する global ゾーン内にあるかどうかを確認します。その Solaris Security Toolkit スクリプトが global ゾーンでは実行されていない場合、スクリプトは logNotGlobalZone 関数を使用してその情報を記録し、終了します。

表 1-4 に、ゾーンに対応する終了および監査スクリプトを示します。

表 1-4 Solaris Security Toolkit 4.2 のゾーンに対応する終了および監査スクリプト

ベーススクリプト名	ゾーン対応の理由	ゾーンの動作
disable-power-mgmt	ゾーンでは電源機能は使用できません。	ログ記録
enable-bsm	ゾーンは BSM を使用できますが、BSM を有効にすることはできません。NGZ で BSM を使用する機能を有効にする前に、まず大域ゾーンで BSM を使用する機能を有効にする必要があります。	ログ記録
enable-ipfilter	ゾーンは IP フィルタを変更できません。	ログ記録
enable-priv-ngs-ports	ゾーンは NFS サーバーになることはできません。	ログ記録
enable-rfc1948	ゾーンは /dev/ip スタックに影響を与えることはできません。	ログ記録
enable-stack-protection	ゾーンはカーネルパラメタを変更できません。	ログ記録
install-nddconfig	ゾーンは /dev/ip スタックに影響を与えることはできません。	ログ記録
install-security-mode	ゾーンは EEPROM にアクセスすることはできません。	ログ記録

一部のゾーン対応スクリプトは、非大域ゾーンで使用する前にアクションが必要

enable-bsm.fin など、ゾーンに対応する一部の Solaris Security Toolkit スクリプトは、非大域ゾーン内でフルに使用する前に、大域ゾーン内でアクションが必要な場合があります。こうしたアクションを取ることなくこのようなスクリプトを実行すると、プロンプトが表示され、これらの機能をフルに使用するために必要なアクション

を取るようにという指示が表示されます。言い換えると、一部のアクションではカーネルモジュールが動作する必要があります。このような場合、大域ゾーンからモジュールをロードする必要があります。そうすれば、非大域ゾーンでそれらのモジュールを使用できます。このような作業を行うまでは、アクションは実行されません。

ドライバに基づく rpcbind 無効化または有効化

Solaris 10 オペレーティングシステムには、Fault Manager Daemon (FMD)、ネットワーク情報サービス (NIS)、ネットワークファイルシステム (NFS) など rpcbind に依存するサービス、および共通デスクトップ環境 (CDE) や GNU Network Object Model Environment (GNOME) などのウィンドウマネージャがあります。Solaris Security Toolkit 4.2 ソフトウェアは、次のようにドライバに基づいて rpcbind を無効/有効にします。

- secure.driver: デフォルトで rpcbind は無効
- server-secure.driver: デフォルトで rpcbind は有効
- suncluster3x-secure.driver: デフォルトで rpcbind は有効
- sunfire_15k_sc-secure.driver: デフォルトで rpcbind は無効

システムの構成によっては、rpcbind を手動で起動するよう構成しなければならない場合があります。SMF の使用法の詳細については、Solaris 10 OS の管理マニュアルを参照してください。

Solaris 10 OS の rpcbind は TCP ラッパーを使用し、これら 2 つの使用は密接に関連しています。各ドライバが TCP ラッパーを自動構成する方法の詳細については、11 ページの「TCP ラッパーの使用法」を参照してください。

▼ rpcbind を有効にする

1. **システムの強化を解除します。**

2. pgrep コマンドを使用して rpcbind が実行中であることを確認します。

```
# pgrep rpcbind
process-id
```

子ゾーンのプロセスを受け取らないように、子ゾーンが付属する大域ゾーンがある Solaris 10 OS を実行しているシステムに対して、次の形式の pgrep コマンドを使用します。

```
# pgrep -z zone-name rpcbind
process-id
```

process-id を受け取れば、rpcbind が実行中であることがわかります。

3. secure.driver と hardening.driver を、それぞれ *new-secure.driver* と *new-hardening.driver* にコピーして名前を変更します。
4. *new-secure.driver* を編集して、hardening.driver への参照を *new-hardening.driver* に置き換えます。
5. *new-hardening.driver* から disable-rpc.fin スクリプトをコメントアウトします。
6. *new-secure.driver* とともに Solaris Security Toolkit を実行することで、カスタマイズされたコピードライバを使用して、強化を再度実行します。
7. システムを再起動します。

注意 – rpcbind サービスを有効にしたあと、追加のサービスが自動的に起動し、それに対応するポートが開く場合があります。Solaris Security Toolkit の監査は、これらの追加サービスを障害として警告します。

TCP ラッパーの使用法

Solaris 10 OS では、次の TCP ラッパー構成が次のドライバに使用されます。構成情報は、/etc/hosts.allow および /etc/hosts.deny ファイルにあります。

注 – これらの構成の引数では、大文字と小文字が区別されます。たとえば、**コード例 1-2** では、LOCAL と ALL はすべて大文字で入力する必要があり、localhost はすべて小文字で入力する必要があります。

TCP ラッパー構成 (secure.driver の場合)

コード例 1-2　Solaris 10 OS の secure.driver 用の TCP ラッパー構成

```
secure.driver: tcpwrappers enabled by default with the following:
        hosts.allow
                sshd:     LOCAL
                sendmail: localhost
        hosts.deny
                 ALL:       ALL
                 # rpcbind: ALL
```

TCP ラッパー構成 (server-secure.driver の場合)

コード例 1-3　Solaris 10 OS の server-secure.driver 用の TCP ラッパー構成

```
server-secure.driver: tcpwrappers enabled by default with the
following:
        hosts.allow
                 ALL: localhost
                sshd: ALL
        hosts.deny
                 ALL: ALL
```

TCP ラッパー構成 (suncluster3x-secure.driver の場合)

コード例 1-4　Solaris 10 OS の suncluster3x-secure.driver 用の TCP ラッパー構成

```
suncluster3x-secure.driver: tcpwrappers enabled by default with
the following:
        hosts.allow
                 <need to allow other cluster members access>
                 ALL: localhost
                sshd: ALL
        hosts.deny
                 ALL: ALL
        NOTE: need to warn if not configured properly by adding
        entries to hosts.allow
```

TCP ラッパー構成 (sunfire_15k_sc-secure.driver の場合)

コード例 1-5　TCP ラッパー構成
Solaris 10 OS の sunfire_15k_sc-secure.driver

```
sunfire_15k_sc-secure.driver: tcpwrappers enabled by default with
the following:
        hosts.allow
                <need to allow other SC sshd access>
                sendmail: localhost
        hosts.deny
                ALL: ALL
        NOTE: need to warn if not configured properly by adding
        entries to hosts.allow
```

環境変数の定義

ドライバ固有の環境変数が設定される順序に変更点があります。

Solaris Security Toolkit の旧バージョン

Solaris Security Toolkit の旧バージョンでは、環境変数が設定される順序は次のとおりです。

1. <*driver-name*>.driver
2. driver.init
   a. user.init
   b. finish.init
3. <*driver-name*>.driver (driver.init のあと)
4. フレームワーク変数 (ドライバファイル)
5. 終了スクリプトの変数の定義

Solaris Security Toolkit 4.2

Solaris Security Toolkit 4.2 ソフトウェアでは、環境変数が設定される順序は次のとおりです。

1. `jass-execute` コール

   a. `driver-init`

   b. `user-init`

   c. `finish.init`

   d. `*secure*`

      i. `driver.init`

     ii. `user.init`

    iii. `finish.init`

     iv. `*config*`

      v. `*hardening*`

手順 d では、一部の変数は手順 i の前または手順 iii のあとで設定できます。

注 – Solaris Security Toolkit 4.2 でドライバ固有の変数が設定される順序が変更されたにも関わらず、`user.init` を使用して優先指定を行うことができる点は旧バージョンから変更されていません。

第2章

フレームワーク関数

この章では、フレームワーク関数の使用、追加、変更、および削除について説明します。フレームワーク関数を使用すると、ソースコードを変更せずに Solaris Security Toolkit ソフトウェアの動作を柔軟に変更することができます。

フレームワーク関数は、新しい終了スクリプトと監査スクリプトの開発に必要なコーディング量を制限し、共通機能の一貫性を維持するときに使用します。たとえば、共通のログ関数を使用することで、新しいソースコードを開発したり、既存のソースコードを変更したりすることなく、レポートメカニズムを構成することができます。同様に、この共通関数のモジュール化コードを使用すれば、バグや拡張機能をより体系的に処理することもできます。

さらに、フレームワーク関数では元に戻すオプションも使用できます。たとえば、`cp` や `mv` コマンドの代わりにフレームワーク関数 `backup_file` を使用すると、元に戻す処理時に操作を復元できます。

この章では、以下の項目を説明します。

フレームワーク関数のカスタマイズ

Solaris Security Toolkit ソフトウェアはモジュラーフレームワークをベースにしており、このフレームワークを使用すると、組織のニーズに合ったさまざまな方法で機能を組み合わせることができます。しかし、時には Solaris Security Toolkit ソフトウェアが提供している標準機能が、実際の使用環境のニーズに合っていないこともあります。フレームワーク関数をカスタマイズすることによって標準機能を補完すれば、

Solaris Security Toolkit ソフトウェアが提供している機能を強化および拡張することができます。フレームワーク関数で行うのは、Solaris Security Toolkit ソフトウェアの実行方法の設定、使用する関数の定義、および環境変数の初期化です。

多くの場合、簡単に標準のフレームワーク関数のファイルとスクリプトをコピーして、用途に合わせて機能をカスタマイズすることができます。たとえば、user.run ファイルを使用すると、標準のフレームワーク関数を追加、変更、置換、または拡張できます。user.run ファイルは、user.init ファイルが環境変数の追加または変更に使用できることを除けば、user.init ファイルと目的は同じです。

場合によっては、新しいフレームワーク関数を作成する必要があります。その場合、類似したフレームワーク関数をコーディングの手引きやテンプレートとして利用し、このマニュアルに記載されている推奨事項に従ってください。フレームワーク関数の作成は、Solaris Security Toolkit ソフトウェアの設計および実装に精通しているユーザーだけが行うようにしてください。

注意 – 独自のフレームワーク関数を作成するときは、特に慎重に行なってください。誤ったプログラム作成を行うと、Solaris Security Toolkit ソフトウェアの、変更を正しく実装したり元に戻す機能、あるいはシステム構成の監査を行う機能に障害が生じることがあります。さらには、ソフトウェアに行った変更によって、ソフトウェアが実行されている対象プラットフォームに悪影響を及ぼす場合もあります。

標準のフレームワークをカスタマイズすることによって Solaris Security Toolkit の機能を拡張する方法を、コード例 2-1 に示します。この例では、JumpStart インストール時に開発者が追加ファイルシステムをマウントできるように mount_filesystems 関数を変更しています。mount_filesystems 関数は、driver_private.funcs スクリプトから user.run ファイルに直接コピーします。関数の変更は、行 8 と 9 で行なっています。

コード例 2-1　　フレームワークのカスタマイズによる機能の拡張

```
1   mount_filesystems()
2   {
3       if [ "${JASS_STANDALONE}" = "0" ]; then
4           mount_fs ${JASS_PACKAGE_MOUNT} ${JASS_ROOT_DIR} \
5               ${JASS_PACKAGE_DIR}
6           mount_fs ${JASS_PATCH_MOUNT} ${JASS_ROOT_DIR} \
7               ${JASS_PATCH_DIR}
8           mount_fs 192.168.0.1:/apps01/oracle \
9               ${JASS_ROOT_DIR}/tmp/apps-oracle
10      fi
11  }
```

コードを簡潔にするために、新しいファイルシステムのマウントに使用する変数を Solaris Security Toolkit 環境変数に変換していません。移植性と柔軟性を持たせるためには、環境変数を使用して実際の値を抽象化してください。これにより、製造、品質保証、開発などの要件が異なる環境にもソフトウェアが配備されるようになるため、変更に一貫性を持たせることができます。

注 – このマウントポイントを使用している終了スクリプト内に同じ機能を実装すれば、ファイルシステムのマウント、使用、アンマウント機能をすべてスクリプト内に埋め込むことは可能です。しかし、複数のスクリプトが単一ファイルシステムを使用している場合には、`mount_filesystems` を使用した方がより効果的かつ効率的にファイルシステムをマウントできます。

注意 – `mount_filesystems` を変更した場合のマイナス面は、Solaris Security Toolkit ソフトウェアのアップデート版のインストール時に `mount_filesystems` を再度変更する必要があることです。

共通のログ関数の使用

以下の関数は、すべてのログ関数とレポート関数を制御する関数で、`Drivers` ディレクトリの `common_log.funcs` ファイルに格納されています。ログ関数とレポート関数は、Solaris Security Toolkit ソフトウェアのすべての操作モードで使用されるため、共通関数と見なされます。このファイルには `logWarning` や `logError` などの共通関数が含まれています。

ここでは、以下の共通ログ関数について説明します。

logBanner

この関数は、バナーメッセージを表示します。通常、このメッセージは、ドライバ、終了スクリプト、監査スクリプトによる出力の前に表示されます。バナーメッセージは実行の開始時と終了時にも使用されます。ログ詳細レベルが 3 (完全モード) 以上の場合にのみ表示されます。詳細レベルについての詳細は、第 7 章を参照してください。

バナーメッセージは、次の 2 つのいずれかの書式になります。この関数に空の文字列を渡した場合は、一行区切り文字が 1 つ表示されます。多くの場合、この行は、表示される出力に強制的に「ブレーク」を入れるときに使用されます。1 つの文字列値を入力した場合は、一行区切り文字が表示されている 2 つの行の間に表示されます。バナーメッセージの例を、コード例 2-2 に示します。

コード例 2-2　　バナーメッセージ例

```
================================================================
Solaris Security Toolkit Version: 4.2
Node name:                        imbulu
Zone name:                        global
Host ID:                          8085816e
```

コード例 2-2　　バナーメッセージ例 (続き)

```
Host address:                          192.168.0.1
MAC address:                           0:0:80:85:81:6e
OS version:                            5.10
Date:                                  Fri Jul 1 22:27:15 EST 2005
================================================================
```

バナーメッセージの表示は、JASS_LOG_BANNER 環境変数を使用して制御できます。この環境変数についての詳細は、第 7 章を参照してください。

logDebug

この関数は、デバッグメッセージを表示します。デバッグメッセージには、[FAIL] や [PASS] などの型接頭辞がありません。詳細レベルが 4 (デバッグモード) 以上の場合にのみデバッグメッセージが表示されます。デフォルトでは、デバッグメッセージを出力しません。詳細レベルについての詳細は、第 7 章を参照してください。

引数:　　$1 - 出力する文字列

戻り値:　　なし

使用例 :

```
logDebug "Print first message for debugging."
```

logError

この関数は、エラーメッセージを表示します。エラーメッセージとは、文字列 [ERR] が含まれているメッセージのことです。

引数:　　$1 - エラーメッセージとして表示する文字列

戻り値:　　なし

使用例 :

```
logError "getScore: Score value is not defined."
```

出力例 :

```
[ERR ] getScore: Score value is not defined.
```

エラーメッセージの表示は、JASS_LOG_ERROR 環境変数を使用して制御できます。この環境変数についての詳細は、第 7 章を参照してください。

logFailure

この関数は、失敗メッセージを表示します。失敗メッセージとは、文字列 [FAIL] が含まれているメッセージのことです。

引数: $1 - 失敗メッセージとして表示する文字列

戻り値: なし

使用例 :

```
logFailure "Package SUNWatfsr is installed."
```

出力例 :

```
[FAIL] Package SUNWatfsr is installed.
```

失敗メッセージの表示は、JASS_LOG_FAILURE 環境変数を使用して制御できます。この環境変数についての詳細は、第 7 章を参照してください。

logFileContentsExist と logFileContentsNotExist

この 2 つの関数は、ファイル内容のチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_fileContentsExist 関数および check_fileContentsNotExist 関数で使用します。

引数: $1 - テストするファイル (文字列値)
$2 - 検索パターン (文字列値)
$3 - 脆弱性値 (0 以上の整数)
$4 - PASS または FAIL メッセージの次にユーザーに表示する関連情報 (オプション)

戻り値: 成功または失敗のメッセージ。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

使用例 :

```
logFileContentsExist /etc/default/inetinit "TCP_STRONG_ISS=2" 0
```

出力例 :

```
[PASS] File /etc/default/inetinit has content matching
TCP_STRONG_ISS=2.
```

logFileExists と logFileNotExists

この 2 つの関数は、ファイルのチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_fileExists 関数および check_fileNotExists 関数と一緒に使用します。

引数: $1 - テストするファイル (文字列値)
$2 - 脆弱性値 (0 以上の整数)。この引数に NULL 文字列値が渡された場合、この関数は logNotice 関数を使用して、通知形式で結果を報告します。引数が 0 である場合、logSuccess 関数を使用して結果を成功と報告し、それ以外の場合は logFailure 関数を使用して失敗と報告します。
$3 - PASS、FAIL、または NOTE メッセージの次にユーザーに表示する関連情報 (オプション)。

戻り値: 成功または失敗のメッセージ。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

使用例 :

```
logFileExists /etc/issue
```

出力例 :

```
[NOTE] File /etc/issue was found.
```

logFileGroupMatch と logFileGroupNoMatch

この 2 つの関数は、ファイルグループメンバーシップのチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_fileGroupMatch 関数および check_fileGroupNoMatch 関数で使用します。

引数: $1 - テストするファイル (文字列値)
$2 - チェックするグループ
$3 - 脆弱性値 (0 以上の整数)
$4 - PASS または FAIL メッセージの次にユーザーに表示する関連情報 (オプション)

戻り値: 成功または失敗のメッセージ。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

使用例 :

```
logFileGroupMatch /etc/motd sys 0
```

出力例 :

```
[PASS] File /etc/motd has group sys.
```

logFileModeMatch と logFileModeNoMatch

この 2 つの関数は、ファイルのアクセス権のチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_fileModeMatch 関数および check_fileModeNoMatch 関数で使用します。

次の引数を指定できます。

引数: $1 - テストするファイル (文字列値)
$2 - チェックするアクセス権
$3 - 脆弱性値 (0 以上の整数)
$4 - PASS または FAIL メッセージの次にユーザーに表示する関連情報 (オプション)

戻り値: 成功または失敗のメッセージ。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

使用例 :

```
logFileModeMatch /etc/motd 0644 0
```

出力例 :

```
[PASS] File /etc/motd has mode 0644.
```

logFileNotFound

この関数は、ファイル未検出メッセージを表示するときに使用します。この関数は、セキュリティー強化と監査の両方を実行する Solaris Security Toolkit コードで使用され、指定したファイルがシステム上で検出されなかったときに標準のメッセージを表示します。

次の引数を指定できます。

- 調べるファイルの名前を示す文字列値
- 脆弱性値を示す 0 以上の整数

  この引数に NULL 文字列値が渡された場合、この関数は logNotice 関数を使用して、通知形式で結果を報告します。そうでない場合は、logFailure 関数を使用して、結果を失敗と報告します。
- FAIL または NOTE メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)

使用例 :

```
logFileNotFound /etc/motd
```

出力例 :

```
[NOTE] File /etc/issue was not found.
```

通知メッセージと失敗メッセージの表示は、それぞれ JASS_LOG_NOTICE 環境変数と JASS_LOG_FAILURE 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

logFileOwnerMatch と logFileOwnerNoMatch

この 2 つの関数は、ファイルの所有権のチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_fileOwnerMatch 関数および check_fileOwnerNoMatch 関数で使用します。

次の引数を指定できます。

- 調べるファイルの名前を示す文字列値
- チェックする所有権を示す文字列値
- 脆弱性値を示す 0 以上の整数
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)

使用例 :

```
logFileOwnerMatch /etc/motd root 0
```

出力例 :

```
[PASS] File /etc/motd has owner root.
```

表示されるメッセージは、成功または失敗のいずれかです。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

logFileTypeMatch と logFileTypeNoMatch

この 2 つの関数は、ファイルタイプのチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_fileTypeMatch 関数および check_fileTypeNoMatch 関数で使用します。

次の引数を指定できます。

- 調べるファイルの名前を示す文字列値
- チェックするファイルタイプを示す文字列値

検出されるファイルタイプを、表 2-1 に示します。

表 2-1 check_fileTypeMatch 関数で検出されるファイルタイプ

ファイルタイプ	説明
b	ブロック型特殊ファイル
c	文字型特殊ファイル
d	ディレクトリ
D	door
f	通常ファイル
l	シンボリックリンク
p	名前付きパイプ (先入れ先出し)
s	ソケット

- 脆弱性値を示す 0 以上の整数
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)

使用例 :

```
logFileTypeMatch /etc/motd f 0
```

出力例 :

```
[PASS] File /etc/motd is a regular file.
```

表示されるメッセージは、成功または失敗のいずれかです。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

logFinding

この関数は、監査検出メッセージを表示します。この関数は、メッセージとして表示する単一文字列を引数として受け取ります。この関数に対して入力された値は、printPrettyPath 関数で処理されてから表示されます。また、詳細レベルが 2 (要約モード) 以上である場合には、オプションのタグがメッセージの先頭に付加されます。以下は、この関数を使用して付加できるオプションのタグです。

- 時刻表示 – デフォルトでは JASS_DISPLAY_TIMESTAMP は定義されていません。JASS_DISPLAY_TIMESTAMP 環境変数が 1 であり、JASS_VERBOSITY が 3 未満である場合、JASS_TIMESTAMP 環境変数で定義された時刻表示が検出メッセージの先頭に付加されます。
- 対象ホスト名 – デフォルトでは JASS_DISPLAY_HOSTNAME は定義されていません。JASS_DISPLAY_HOSTNAME 環境変数が 1 であり、JASS_VERBOSITY が 3 未満である場合、JASS_HOSTNAME 環境変数で定義された対象ホスト名が検出メッセージの先頭に付加されます。
- 現在のスクリプト名 – デフォルトでは JASS_DISPLAY_SCRIPTNAME は定義されていません。JASS_DISPLAY_SCRIPTNAME 環境変数が 1 であり、JASS_VERBOSITY が 3 未満である場合、現在の監査スクリプトの名前が検出メッセージの先頭に付加されます。

注 – driver.run スクリプトのフロー内など、監査スクリプト以外で検出が行われた場合には、ドライバの名前が使用されます。

これら 3 つの出力タグは、まとめて使用することも単独で使用することもできます。出力結果行での表示順は、入力行で指定した順序と同じになります。この関数と詳細レベルについての詳細は、第 7 章を参照してください。

使用例 :

```
logFinding "/etc/motd"
```

出力例 :

```
test-script /etc/motd
```

logFormattedMessage

この関数は、チェックするスクリプト名、チェックの目的、根拠などの情報を表示するフォーマット済み監査スクリプトヘッダーを作成するときに使用します。この関数は、単一文字列値を引数として受け取り、関数に渡されるこのメッセージをフォーマットします。

これらのメッセージは次のようにフォーマットされます。

- 最大文字数は 75 文字
- 前後に半角スペースが付いたシャープ (#) 記号が先頭に付加される
- 重複しているパス名のスラッシュが削除される

フォーマットされたメッセージは、詳細レベルが 3 (完全モード) 以上の場合にのみ表示されます。この関数と詳細レベルについての詳細は、第 7 章を参照してください。

使用例 :

```
logFormattedMessage "Check system controller secure shell
configuration."
```

出力例 :

```
# Check system controller secure shell configuration.
```

logInvalidDisableMode

この関数は、JASS_DISABLE_MODE 環境変数が無効な値に設定されているときに、エラーメッセージを表示します。このユーティリティー関数は、JASS_DISABLE_MODE 環境変数の状態を報告します。この環境変数についての詳細は、第 7 章を参照してください。

この関数は引数をとらず、以下のような出力を行います。

```
[ERR ] The JASS_DISABLE_MODE parameter has an invalid value: […]
[ERR ] value must either be “script” or “conf”.
```

logInvalidOSRevision

この関数は、check_os_revision 関数または check_os_min_revision 関数のいずれかがチェックに失敗したときに使用します。このユーティリティー関数は、適用されていない Solaris OS のバージョンで関数が呼び出されている場合に報告を行います。たとえば、Solaris 8 OS で Solaris 10 OS のスクリプトを使用してみるときなどに、この関数を使用します。

使用例 :

```
logInvalidOSRevision "5.10"
```

出力例 :

```
[NOTE] This script is only applicable for Solaris version 5.10.
```

複数のバージョンを指定する場合は、各バージョンの間にハイフン (-) を入力します (たとえば、5.8-5.9)。

この関数は、通知メッセージを表示します。これらのメッセージの表示は、JASS_LOG_NOTICE 環境変数を使用して制御できます。

注 – JASS_LOG_NOTICE 環境変数は、Solaris 10 OS を実行しているシステムには使用しません。

この環境変数についての詳細は、第 7 章を参照してください。

logMessage

この関数は、ユーザーに対してメッセージを表示するときに使用します。使用するのは、メッセージに関連付けられているタグが含まれていない場合です。この関数は logFormattedMessage 関数に似ていますが、フォーマットされていないメッセージを表示します。この関数は、そのまま表示される単一文字列値を引数として受け取り、何も変更は行いません。

フォーマットされていないメッセージは、詳細レベルが 3 (完全モード) 以上の場合にのみ表示されます。この関数と詳細レベルについての詳細は、第 7 章を参照してください。

使用例 :

```
logMessage "Verify system controller static ARP configuration."
```

出力例 :

```
Verify system controller static ARP configuration.
```

logNotGlobalZone

この関数は、大域ゾーンで実行する必要があるためにスクリプトが実行されない logNotice を使用してメッセージを記録します。つまり、スクリプトは非大域ゾーンでは実行できません。

引数**:** なし

戻り値**:** なし

使用例 :

```
logNotGlobalZone
```

logNotice

この関数は、通知メッセージを表示するときに使用します。この関数は、通知メッセージとして表示する単一文字列値を引数として受け取ります。通知メッセージとは、文字列 [NOTE] が含まれているメッセージのことです。

使用例 :

```
logNotice "Service ${svc} does not exist in ${INETD}."
```

出力例 :

```
[NOTE] Service telnet does not exist in /etc/inetd.conf.
```

通知メッセージの表示は、JASS_LOG_NOTICE 環境変数を使用して制御できます。この環境変数についての詳細は、第 7 章を参照してください。

logPackageExists と logPackageNotExists

この 2 つの関数は、ソフトウェアパッケージがインストールされているかどうかを判断するチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_packageExists 関数および check_packageNotExists 関数で使用します。

次の引数を指定できます。

- 調べるソフトウェアパッケージの名前を示す文字列値
- 脆弱性値を示す 0 以上の整数
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)

使用例 :

```
logPackageExists SUNWcsr 0
```

出力例 :

```
[PASS] Package SUNWcsr is installed.
```

表示されるメッセージは、成功または失敗のいずれかです。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

logPatchExists と logPatchNotExists

この 2 つの関数は、ソフトウェアのパッチがインストールされているかどうかを判断するチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_patchExists 関数および check_patchNotExists 関数で使用します。

次の引数を指定できます。

- 調べるパッチの識別子 (番号) を示す文字列値
- 脆弱性値を示す 0 以上の整数
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)

使用例 :

```
logPatchExists 123456-01 0
```

出力例 :

```
[PASS] Patch ID 123456-01 or higher is installed.
```

表示されるメッセージは、成功または失敗のいずれかです。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

logProcessArgsMatch と logProcessArgsNoMatch

この 2 つの関数は、実行時プロセス引数のチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_processArgsMatch 関数および check_processArgsNoMatch 関数で使用します。

次の引数を指定できます。

- 調べるプロセスの名前を示す文字列値
- 引数の検索パターンを示す文字列値
- 脆弱性値を示す 0 以上の整数
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)

使用例 :

```
logProcessArgsMatch inetd "-t" 0
```

出力例 :

```
[PASS] Process inetd found with argument -t.
```

表示されるメッセージは、成功または失敗のいずれかです。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

logProcessExists と logProcessNotExists

この 2 つの関数は、プロセスのチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_processExists 関数および check_processNotExists 関数で使用します。

引数: $1 - プロセス名 (文字列)
$2 - 脆弱性値 (数値)
$3 - PASS または FAIL メッセージの次に
ユーザーに表示する関連情報 (オプション)

使用例：

```
logProcessExists nfsd 0
```

出力例：

```
[PASS] Process nfsd was found.
```

表示されるメッセージは、成功または失敗のいずれかです。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

logProcessNotFound

この関数は、検出されないプロセスに対する FAIL メッセージを記録するときに使用します。指定したプロセスがシステムで検出されなかったときに、この関数は標準の「プロセスの未検出」メッセージを表示します。

引数: $1 - プロセス名 (文字列)
$2 - PASS または FAIL メッセージの次に
ユーザーに表示する関連情報 (オプション)

使用例：

```
logProcessNotFound inetd
```

出力例：

```
[FAIL] Process inetd was not found.
```

これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数を使用して制御できます。この環境変数についての詳細は、第 7 章を参照してください。

logScore

この関数は、監査実行時に検出されたエラー数を報告するときに使用します。

引数: $1 - レポートと関連付ける文字列
$2 - エラーの数 (文字列)

戻り値: 監査実行時に検出されたエラー数。

使用例 :

```
logScore "Script Total:" "0"
```

出力例 :

```
[PASS] Script Total: 0 Errors
```

logScriptFailure

この関数は、スクリプトの失敗を、対応するスクリプトの失敗ログに記録するときに使用します。

引数: $1 - 失敗のタイプ:
"エラー"
"警告"
"注意"
"失敗"

$2 - 記録された失敗のタイプのカウント (文字列)。

使用例 :

```
logScriptFailure "failure" 1
```

この例は、${JASS_REPOSITORY}/${JASS_TIMESTAMP}/jass-script-failures.txt ファイルに対して 1 つの失敗を記録します。

logServiceConfigExists と logServiceConfigNotExists

この 2 つの関数は、構成ファイルが存在するかどうかを判断するチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_serviceConfigExists 関数および check_serviceConfigNotExists 関数で使用します。

引数: $1 - サービス名 (文字列)
$2 - 脆弱性値 (数値)
$3 - PASS または FAIL メッセージの次に
ユーザーに表示する関連情報 (オプション)

使用例 :

```
logServiceConfigExists /etc/apache/httpd.conf 0
```

出力例 :

```
[PASS] Service Config File /etc/apache/httpd.conf was found.
```

表示されるメッセージは、成功または失敗のいずれかです。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

logServiceDisabled と logServiceEnabled

この 2 つの関数は、指定したサービスが同じように有効または無効になっていることを記録するときに使用します。

引数: $1 - サービス名 (文字列)
$2 - 脆弱性値 (数値)
$3 - PASS または FAIL メッセージの次に
ユーザーに表示する関連情報 (オプション)

使用例 :

```
logServiceDisabled "svc:/network/telnet:default" 0 ""
```

出力例 :

```
[PASS] Service svc:/network/telnet:default was not enabled.
```

logServiceInstalled と logServiceNotInstalled

この 2 つの関数は、指定したサービスが同じようにインストールされている、またはインストールされていないことを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_serviceEnabled 関数および check_serviceDisabled 関数で使用します。

引数: $1 - サービス名 (文字列)
$2 - 脆弱性値 (数値)
$3 - PASS または FAIL メッセージの次に
ユーザーに表示する関連情報 (オプション)

使用例 :

```
logServiceInstalled “svc:/network/telnet:default” 1 “”
```

出力例 :

```
[FAIL] Service svc:/network/telnet:default was installed.
```

logServiceOptionDisabled と logServiceOptionEnabled

この関数は、あるサービスで、指定したオプションが特定の値に設定されているかどうかを記録するときに使用します。この関数は check_serviceOptionDisabled 関数および check_serviceOptionEnabled 関数と一緒に使用します。

引数: $1 - プロセス名 (文字列)
$2 - サービスプロパティー名 (文字列)
$3 - サービス名 (文字列)
$4 - サービスプロパティー値 (文字列)
$5 - 脆弱性値 (数値)
$6 - PASS または FAIL メッセージの次に
ユーザーに表示する関連情報 (オプション)

使用例 :

```
logServiceOptionEnabled “in.ftpd” “inetd_start/exec”
“svc:/network/ftp” “-1” 0 “”
```

出力例 :

```
[PASS] Service in.ftpd of svc:/network/ftp property
inetd_start/exec has option -1.
```

logServiceProcessList

この関数は、SMF サービスと関連付けられたプロセスのリストを出力するときに使用します。プロセスごとに、プロセス ID、プロセスユーザー ID、プロセスコマンドの 3 つの項目が出力されます。

引数: $1 - SMF サービス
$2 - PASS または FAIL
$3 - プロセス ID (pid)、プロセスユーザー ID (user)、およびプロセスコマンド (command) と関連付けられたプロセスのリスト。

使用例 :

```
logServiceProcessList svc:/network/telnet 0 "245 root in.telnetd"
```

出力例 :

```
[PASS] Service svc:/network/telnet was found running (pid 245,
user root, command in.telnetd).
```

logServicePropDisabled と logServicePropEnabled

この関数は、あるサービスで、指定したオプションが有効/無効のどちらに設定されているかを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_serviceOptionEnabled 関数および check_serviceOptionDisabled 関数と一緒に使用します。

引数: $1 - サービス名 (文字列)
$2 - プロパティー名 (文字列)
$3 - プロパティー値 (文字列)
$4 - 脆弱性値 (数値)
$5 - PASS または FAIL メッセージの次にユーザーに表示する関連情報 (オプション)

使用例 :

```
logServicePropDisabled svc:/network/ftp enable_tcpwrappers
enabled 1 ""
```

出力例 :

```
[FAIL] Service svc:/network/ftp property enable_tcpwrappers was
enabled.
```

logServiceRunning と logServiceNotRunning

この関数は、特定のサービスが実行中であるかどうかを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_serviceRunning 関数および check_serviceNotRunning 関数と一緒に使用します。

引数: $1 - サービス名 (文字列)
$2 - 脆弱性値 (数値)
$3 - プロセスリスト (オプション)
$4 - PASS または FAIL メッセージの次に
ユーザーに表示する関連情報 (オプション)

使用例 :

```
logServiceRunning svc:/network/ftp 1
```

出力例 :

```
[FAIL] Service svc:/network/ftp was not running.
```

logStartScriptExists と logStartScriptNotExists

この 2 つの関数は、実行コントロール開始スクリプトが存在するかどうかを判断するチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_startScriptExists 関数および check_startScriptNotExists 関数で使用します。

引数: $1 - テストする開始スクリプト (文字列)
$2 - 脆弱性値 (数値)
$3 - PASS または FAIL メッセージの次に
ユーザーに表示する関連情報 (オプション)

使用例 :

```
logStartScriptExists /etc/rc3.d/S89sshd 0
```

出力例 :

```
[PASS] Start Script /etc/rc3.d/S89sshd was found.
```

表示されるメッセージは、成功または失敗のいずれかです。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

logStopScriptExists と logStopScriptNotExists

この 2 つの関数は、実行コントロール停止スクリプトが存在するかどうかを判断するチェック結果に関連するメッセージを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_stopScriptExists 関数および check_stopScriptNotExists 関数で使用します。

引数: $1 - テストする停止スクリプト (文字列)
$2 - 脆弱性値 (数値)
$3 - PASS または FAIL メッセージの次に
ユーザーに表示する関連情報 (オプション)

使用例 :

```
logStopScriptExists /etc/rc2.d/K03sshd 0
```

出力例：

```
[PASS] Stop Script /etc/rc2.d/K03sshd was found.
```

表示されるメッセージは、成功または失敗のいずれかです。これらのメッセージの表示は、JASS_LOG_FAILURE 環境変数と JASS_LOG_SUCCESS 環境変数を使用して制御できます。これらの環境変数についての詳細は、第 7 章を参照してください。

logSuccess

この関数は、成功メッセージを表示するときに使用します。この関数は、監査成功メッセージとして表示する単一文字列値を引数として受け取ります。成功メッセージとは、文字列 [PASS] が含まれているメッセージのことです。

使用例：

```
logSuccess "Package SUNWsshdr is installed."
```

出力例：

```
[PASS] Package SUNWsshdr is installed.
```

成功メッセージの表示は、JASS_LOG_SUCCESS 環境変数を使用して制御できます。この環境変数についての詳細は、第 7 章を参照してください。

logSummary

この関数は、Solaris Security Toolkit の実行からの概要情報を表示するときに使用します。この関数の引数は、実行と比較するドライバ、およびスクリプトの実行回数です。

使用例：

```
logSummary undo.driver 61
```

出力例:

```
================================================================================
[SUMMARY] Results Summary for UNDO run of jass-execute
[SUMMARY] The run completed with a total of 91 scripts run.
[SUMMARY] There were  Failures in   0 Scripts
[SUMMARY] There were  Errors   in   0 Scripts
[SUMMARY] There was a Warning  in   1 Script
[SUMMARY] There were  Notes    in  61 Scripts

[SUMMARY] Warning Scripts listed in:
         /var/opt/SUNWjass/run/20050616052247/jass-undo-script-warnings.txt
[SUMMARY] Notes Scripts listed in:
         /var/opt/SUNWjass/run/20050616052247/jass-undo-script-notes.txt
================================================================================
```

logUserLocked と logUserNotLocked

この関数は、特定のユーザーアカウントがロックされていたかどうかを記録するときに使用します。どちらの関数も必要に応じて単独で使用することができますが、主に check_userLocked 関数および check_userNotLocked 関数で使用します。

引数:
- $1 - ユーザー名 (文字列)
- $2 - 脆弱性値 (数値)
- $3 - PASS または FAIL メッセージの次にユーザーに表示する関連情報 (オプション)

使用例 :

```
logUserLocked "uucp" 1
```

出力例 :

```
[FAIL] User uucp was not locked.
```

logUndoBackupWarning

この関数は、元に戻す処理の結果に関する一般的な警告を記録するときに使用します。

使用例：

```
logUndoBackupWarning
```

出力例：

```
[WARN] Creating backup copies of some files may cause unintended
effects.
[WARN] This is particularly true of /etc/hostname.[interface]
files as well as crontab files in /var/spool/cron/crontabs.
```

logWarning

この関数は、警告メッセージを表示するときに使用します。この関数は、警告メッセージとして表示する単一文字列値を引数として受け取ります。警告メッセージとは、文字列 [WARN]が含まれているメッセージのことです。

使用例：

```
logWarning "User ${acct} is not listed in ${JASS_PASSWD}."
```

出力例：

```
[WARN] User abc is not listed in /etc/passwd.
```

警告メッセージの表示は、JASS_LOG_WARNING 環境変数を使用して制御できます。この環境変数についての詳細は、第 7 章を参照してください。

その他の共通関数の使用

ここで説明する関数は、Solaris Security Toolkit ソフトウェアのいくつかの領域内で使用され、ほかのフレームワーク関数 (接尾辞 .func で終わるファイル) で提供される機能に特化していないその他の共通関数です。これらの関数は、Drivers ディレクトリの common_misc.funcs ファイルに格納されています。このファイルには、isNumeric や printPretty などの共通ユーティリティー関数が含まれています。

ここでは、以下のその他の共通関数について説明します。

adjustScore

注 – この関数は、監査の実行に対してのみ適用されます。

この関数は、`audit_public.funcs` ファイルで定義されている関数によって提供されているメソッド以外でスコアを増やすときに使用します。たとえば、監査スクリプトで失敗を判定できるだけでよい場合があります。こういった場合には、この関数を使用してスコアを調整して失敗を示すようにします。ユーザーが値を提供しないときは、エラーメッセージが記録され、スコアは調整されません。

引数**:** `$1` - 監査スクリプトの現在のスコアに追加する値 (正の整数)

戻り値**:** なし

使用例 :

```
adjustScore 1
```

checkLogStatus

注 – この関数は、監査操作に対してのみ適用されます。

この関数は、呼び出し元関数がその結果を記録するよう要求しているかどうかを判断するときに使用します。

引数**:** `$1` - ログパラメータの値

戻り値: 0 - 呼び出し元関数により記録するよう要求された出力はなし
1 - 値は LOG で、呼び出し元関数がその結果を記録するよう要求している

使用例 :

```
checkLogStatus "${_logParameter}"
```

clean_path

この関数は、余分なスラッシュ文字 (/) をファイル名から削除するときに使用します。パス名をユーザーに表示したり、記録したりする前に、この関数を使用してパス名の整理を行います。

引数: $1 - クリーンするパス

戻り値: すべての重複するスラッシュ文字 (/)が削除された後の $1 にある値を返す。

使用例 :

```
newPath=`clean_path "${oldPath}"`
```

extractComments

この関数は、ファイルやスクリプトからコメントを削除するときに使用します。この関数では、コメントを数字 (#) で始まり、行の最後まで続くテキストの部分文字列として定義します。

引数: $1 - スクリプト名やファイル名などのトークンのリスト

戻り値: コメントアウトされているすべてのテキストを削除。

使用例 :

```
FinishScripts=`extractComments "${JASS_FILES}"`
```

get_driver_report

この関数は、ログファイルを読み込み、エラーまたは警告を報告したスクリプトの数を返すときに使用します。

引数: $1 - チェックするログファイル

戻り値: 255 - 指定されていない失敗
0 - 成功
1 - ログファイルは読み取り不可能

使用例 :

```
failures=`get_driver_report "${JASS_SCRIPT_FAIL_LOG}"`
```

get_lists_conjunction

この関数は、リスト A と B を取得し、A と B 両方にある要素から構成されるリスト C を戻すときに使用します。

引数: $1 - *listA* (空白で区切られたトークンから構成)
$2 - *listB* (空白で区切られたトークンから構成)

戻り値: リスト A とリスト B の両方にあるすべての要素を含むリスト C。

使用例 :

```
SvcsToLog=`get_lists_conjunction "${JASS_SVCS_DISABLE}"
"${JASS_SVCS_ENABLE}"`
```

get_lists_disjunction

この関数は、リスト A と B を取得し、リスト A 内には存在するがリスト B には存在しない要素から構成されるリスト C を戻すときに使用します。

引数: $1 - *listA* (空白で区切られたトークンから構成)
$2 - *listB* (空白で区切られたトークンから構成)

戻り値: リスト A にはあるがリスト B には存在しない要素を含むリスト C。

使用例 :

```
SvcsToDisable=`get_lists_disjunction "${JASS_SVCS_DISABLE}"
"${JASS_SVCS_ENABLE}"`
```

invalidVulnVal

注 – この関数は、監査操作に対してのみ適用されます。

この関数は、脆弱性値の引数が正の数であるかどうかを判定するときに使用します。この関数は、失敗ごとにエラーメッセージを記録します。脆弱性値として無効な引数が関数に指定されていないかどうかを判定する際に、この関数が必要となります。その他のすべての点では、isNumeric 関数と同様に動作します。

引数: $1 - チェックする脆弱性

戻り値: 0 - 脆弱性は正の数
1 - 脆弱性は正の数ではない

使用例 :

```
invalidVulnVal "${testVulnerability}"
```

isNumeric

この関数は、文字列の引数が正の数であるかどうかを判定するときに使用します。この関数は、入力値が 1 つの正の数であることを検証する必要がある場合に、ソフトウェア全体にわたりヘルパー関数で使用されます。値が正の数である場合は、0 が表示され、それ以外の数の場合は、1 が表示されます。

引数: $1 - チェックする文字列

戻り値: 0 - 文字列は正の数
1 - 文字列は正の数ではない

使用例 :

```
isNumeric "${testString}"
```

printPretty

この関数は、印刷出力結果をフォーマットして読みやすくするときに使用します。この関数は、フォーマットされていない入力文字列を受け取って、フォーマット処理します。フォーマットされた文字列は 72 文字で改行され、各出力行は 3 文字分だけインデントされます。

引数: $1 - 印刷する文字列

戻り値: なし

使用例 :

```
printPretty "${CommentHeader}"
```

printPrettyPath

この関数は、パス名をフォーマットするときに使用します。この関数は、フォーマットされていないパス名を入力として受け取ります。入力された文字列から余分なスラッシュを取り除いてから、結果を表示します。文字列が空の場合には、表示場所にキーワード「<No Value>」が表示されます。

引数: $1 - 印刷する文字列

戻り値: なし

使用例 :

```
printPrettyPath "${PathToLogFile}"
```

strip_path

この関数は、ファイル名から JASS_ROOT_DIR 接頭辞を削除するときに使用します。この関数は、文字列引数を入力として受け取り、JASS_ROOT_DIR 接頭辞を削除して、単一のスラッシュ文字 (/) に置き換えてからその値を返します。JASS マニフェストファイルにパス名を格納する際に、add_to_manifest 関数とともにこの関数を使用します。

引数: $1 - クリーンするファイルパス

戻り値: なし

使用例：

```
StrippedString=`strip_path "${JASS_ROOT_DIR}/etc/motd"`
```

ドライバ関数の使用

以下の関数は、ドライバとしての機能を持つ関数です。これらの関数は、Drivers ディレクトリの driver_public.funcs ファイルに格納されています。このファイルには、add_pkg や copy_a_file などの関数が含まれています。

スクリプトをカスタマイズまたは作成する場合は、次の関数を使用して標準の操作を実行してください。

add_crontab_entry_if_missing

注 – この関数は、Solaris 10 OS の SMF でのみ使用します。

この関数は、プログラム `$2` がユーザーの `$1` `crontab` ではない場合に、`crontab` 行 `$3` を `crontab` に追加するときに使用します。`$4` が 0 である場合、変更前に `crontab` ファイルをバックアップします (使用例を参照)。この関数は `crontab` コメント行を無視します。

引数: `$1` - 変更する `crontab` のユーザー ID
`$2` - `crontab` に追加するプログラム (フルパス名)
`$3` - $2 が `crontab` ファイルに存在しない場合に追加する `crontab` 行
`$4` - 0 の場合、変更前に `backup_file` を呼び出す
(そうでない場合ファイルは作成されているか、
すでにバックアップされている。)

戻り値: `1` - `crontab` ファイルがバックアップされていた場合。
それ以外の場合は、引数 `$4` を変更せずに戻す。

使用例 :

```
add_crontab_entry_if_missing 'root' '/usr/lib/acct/dodisk' '0 2 * * 4
/usr/lib/acct/dodisk' 0
```

add_option_to_ftpd_property

注 – この関数は Solaris 10 の SMF にのみ使用し、`ftp` デーモンにのみ適用されます (オプション `-1` または `-a`)。

注意 – 関数 `add_option_to_gl_property` または `add_option_to_smf_property` がある場合は、その関数の名前を `add_option_to_ftpd_property` に変更してください。

この関数は、Solaris 10 OS で、SMF が有効な `in.ftpd` サービスプロパティー値にオプションを追加するときに使用します。強化操作用にのみこの関数を使用します。この関数は、元に戻す処理のための Solaris Security Toolkit マニフェストファイルに書き込みを行います。

引数: `$1` - 開始コマンドに追加するオプション `a` または `1` (それぞれ `ftpaccess`(4) および `log ftp` セッションと使用)

戻り値: なし

使用例 :

```
add_option_to_ftpd_property "a"
```

add_patch

この関数は、Solaris OS パッチをシステムに追加するときに使用します。デフォルトでは、インストールされるパッチが JASS_PATCH_DIR ディレクトリに格納されていることを前提としています。この関数のオプションを、表 2-2 に示します。

表 2-2　add_patch 終了スクリプト関数のオプション

オプション	説明
-o *options*	渡されるオプション
-M *patchdir*	ソースディレクトリへの完全指定パス
patchlist	パッチリスト、または適用するパッチリストが含まれているファイル名

使用例 :

```
add_patch 123456-01
add_patch -M ${JASS_PATCH_DIR}/OtherPatches patch_list.txt
```

add_pkg

この関数は、Solaris OS パッケージをシステムに追加するときに使用します。デフォルトでは、パッケージが JASS_PACKAGE_DIR ディレクトリに格納されており、また、そのパッケージが Sun の標準フォーマット、スプールディレクトリ、またはパッケージストリームファイルのいずれかにあることを前提としています。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。元に戻す処理を実行すると、この関数を使用して追加したパッケージは、システムから削除されます。この関数のオプションを、表 2-3 に示します。

表 2-3　add_pkg 関数のオプション

オプション	説明
-a *ask_file*	pkgadd ask ファイル名。デフォルトでは、他のファイルを指定しない場合、pkgadd ask ファイル (noask_pkgadd) が使用されます。
-d *src_loc*	インストールするソースパッケージ (ストリームまたはディレクトリ) への完全指定パス
-o *options*	pkgadd コマンドオプション
package	インストールするパッケージ

使用例：

```
add_pkg ABCtest
add_pkg -d ${JASS_ROOT_DIR}/${JASS_PACKAGE_DIR}/SUNWjass.pkg SUNWjass
```

add_to_manifest

この関数は、ヘルパー関数を呼び出さずに、強化処理中に項目をマニフェストファイルに手動で挿入するときに使用します。コマンドで元に戻す処理を実行しなければならない場合に、この方法が最もよく使用されます。このオプションは、システムの完全性と Solaris Security Toolkit のリポジトリが保護されるように注意して使用してください。

add_to_manifest コマンドには、次の構文を使用します。

```
add_to_manifest operation src dst args
```

このコマンドでは、JASS_RUN_MANIFEST ファイルの項目を JASS_REPOSITORY/jass-manifest.txt ファイルに格納します。これは、終了スクリプトで実行した変更を元に戻すために重要なファイルです。

注 – Solaris Security Toolkit によるすべての操作において、上記の各引数がサポートされるわけではありません。src、dst、および args のオプションは、選択した操作によって意味が異なる場合があります。これについては、表 2-4 で説明します。

---

add_to_manifest 関数でサポートされる操作を、表 2-4 に示します。この表では、各オプションの説明のあとに、追加されたマニフェスト項目の例を記載しています。

注意 – x マニフェストオプションを使用する際は、細心の注意を払って実行してください。root ユーザーとして Solaris Security Toolkit を元に戻す処理を行うときに、この操作で指定したコマンドが実行されます。不注意に実行すると、データが失

われたり、対象システムが不安定になる可能性があります。たとえば、元に戻す処理中に rm -rf/ を実行すると、X マニフェスト項目によってシステムのルートパーティションが削除されることがあります。

表 2-4 add_to_manifest オプションとマニフェスト項目例

オプション	説明
C	ファイルがコピーされたことを示します。この場合、src パラメータと dst パラメータは、それぞれ元のファイルとコピーしたファイルの名前を表します。その他の引数は使用しません。 `install-templates.fin /etc/syslog.conf /etc/ \ syslog.conf.JASS.20020823230626`
D	ディレクトリが作成されたことを示します。この場合、src パラメータは新規作成されたディレクトリの名前を表します。その他の引数は使用しません。 `disable-lp.fin /var/spool/cron/crontabs.JASS`
J	システム上でファイルが新規作成されたことを示します。この操作は、src パラメータで指定したファイルがシステムに存在していない場合にのみ使用されます。元に戻す処理中に、この操作コードが付いたファイルが削除されます。この操作では、src パラメータと dst パラメータは、それぞれ元のファイルと保存したファイルの名前 (ファイル名に JASS_SUFFIX を含める必要がある) を表します。 `disable-power-mgmt.fin /noautoshutdown \ /noautoshutdown.JASS.20020823230629`
M	ファイルが移動されたことを示します。この場合、src パラメータと dst パラメータは、それぞれ元のファイルと移動したファイルの名前を表します。その他の引数は使用しません。 `disable-ldap-client.fin /etc/rcS.d/K41ldap.client \ /etc/rcS.d/_K41ldap.client.JASS.20020823230628`

表 2-4　add_to_manifest オプションとマニフェスト項目例 (続き)

オプション	説明
R	ファイルがシステムから削除されたことを示します。この場合、src パラメータは削除されたディレクトリの名前を表します。この操作コードが付いているファイルは、Solaris Security Toolkit の元に戻すコマンドでは復元できません。
S	シンボリックリンクが作成されたことを示します。この場合、src パラメータと dst パラメータは、それぞれソースファイルとターゲットファイルの名前を表します。元に戻す処理中に、この操作のタグが付いたファイルのシンボリックリンクが、システムから削除されます。 `install-templates.fin ../init.d/nddconfig /etc/rc2.d/ \ S70nddconfig`
X	Solaris Security Toolkit でこの操作コードを持つマニフェスト項目を処理するときに、実行しなければならないコマンドが定義されていることを示します。これは特殊な操作であり、標準の操作の範囲を超えている複雑なコマンドを実行するために、最もよく使用されます。たとえば、install-fix-modes.fin 終了スクリプトで、Fix Modes プログラムで行った変更を元に戻すように指示するときは、次のマニフェスト項目を追加します。 `/opt/FixModes/fix-modes -u` このコマンドでは、-u オプションを指定して fix-modes プログラムを実行するように指示しています。この操作で処理するすべてのコマンドは、プログラムへの絶対パスを使用して指定する必要があります。

backup_file

この関数は、既存のファイルシステムオブジェクトをバックアップするときに使用します。元のファイルのバックアップには、標準の命名規則を使用します。この命名規則では、元のファイル名の最後に JASS_SUFFIX を付加します。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

Solaris Security Toolkit ソフトウェアで実行中に変更されたファイルのバックアップコピーを保存するかしないかを、JASS_SAVE_BACKUP 変数で指定します。この環境変数を 0 に設定すると、バックアップファイルはシステムに保存されません。ファイルを保存しなかった場合には、元に戻すコマンドを使用して復元することはできません。

使用例 :

```
backup_file /etc/motd
```

backup_file_in_safe_directory

この関数は、元のディレクトリに格納できないファイルを無効にするとき (詳細は 64 ページの「disable_file」を参照)、および元ファイルを移動するだけでなくさらに編集するためにファイルのコピーを残すときに使用します。これには、ディレクトリ /etc/skel/、/var/spool/cron/crontabs/、/etc/init.d/、および /etc/rc*x*.d/ のすべてのファイルが含まれます。

引数: $1 - コピー元ファイルへの完全指定パス
$2 - ファイルを元に戻す「-u」に設定されている場合、以前の時刻表示はファイル名から削除されます。

戻り値: なし

使用例 :

```
backup_file_in_safe_directory
${JASS_ROOT_DIR}etc/rcS.d/S42coreadm
```

change_group

この関数は、ファイルグループの所有権を変更するときに使用します。この関数では、元に戻す処理を行ったときに元に戻せるように、必要なマニフェスト項目を自動的に追加します。

引数: $1 - ファイル所有者のグループ ID
$2 - グループ所有権を変更する 1 つまたは複数のファイル
通常または特別なファイルまたはディレクトリでなければならず、ソフトリンクであってはなりません。

戻り値: 0 - ファイルが現在正しいグループ所有権を持っている場合
0 以外 - ファイルまたはファイルアクセス権が指定されていないか、chown が失敗した場合

使用例 :

```
change_group root ${JASS_ROOT_DIR}var/core
```

change_mode

この関数は、ファイルのアクセス権モードを変更するときに使用します。この関数では、元に戻す処理を行ったときに元に戻せるように、必要なマニフェスト項目を自動的に追加します。

引数: $1 - 8 進数 chmod(1) 形式のファイルアクセス権 (0700 など)
$2 - chmod の対象である 1 つまたは複数のファイル
通常または特別なファイルまたはディレクトリでなければならず、ソフトリンクは対象となりません。

戻り値: 0 - ファイルが現在正しい所有権を持っている場合
0 以外 - ファイルまたはファイルアクセス権が指定されていないか、chown が失敗した場合

使用例 :

```
change_mode 0700 ${JASS_ROOT_DIR}var/core
```

change_owner

この関数は、ファイルの所有権 (およびオプションでグループ) を変更するときに使用します。この関数では、元に戻す処理を行ったときに元に戻せるように、必要なマニフェスト項目を自動的に追加します。

引数: $1 - ファイル所有者のユーザー ID
$2 - 所有権を変更する 1 つまたは複数のファイル (通常または特別なファイル/ディレクトリでなければならず、ソフトリンクであってはなりません。)

戻り値: 0 - ファイルが現在正しい所有権を持っている場合
0 以外 - ファイルまたはファイルアクセス権が指定されていないか、chown が失敗した場合

使用例 :

```
change_owner root:root ${JASS_ROOT_DIR}var/core
change_owner root ${JASS_ROOT_DIR}var/core
```

check_and_log_change_needed

この関数は、共通の操作を移動し、ファイル内の現在の値をチェックし、1 つのファイルにまとめてフレームワーク関数に格納することにより、終了スクリプトをクリーンに保つときに使用します。この関数は、終了スクリプトの作成者が単一ファイル内の変数を繰り返しチェックする場合に便利です。

この関数は、ファイル内で等号 (=) により区切られているパラメータをチェックし、記録します。新しい値が設定されると、グローバル変数 new_var が新しい値に設定されます。それ以外の場合、new_var はファイル内の既存の値に設定されます。最新の値が以前の値と異なる場合、ログメッセージが出力され、グローバル変数 change_needed が増分されます。

この関数は write_val_to_file 関数と一緒に使用してください (75 ページの「write_val_to_file」を参照)。

引数: $1 - ファイル名
$2 - ファイル内のキーワード
$3 - 新しい値

戻り値: 空である場合を除き、グローバル環境変数 new_var を新しい値に設定。空の場合はファイル内の値に設定され、設定されていない場合は “” になる。

使用例 :

```
change_needed="0"
check_and_log_change_needed "/etc/default/passwd" "MINALPHA"
"${JASS_PASS_MINALPHA}"
minalpha="${new_var}"
check_and_log_change_needed "/etc/default/passwd" "MINLOWER"
"${JASS_PASS_MINLOWER}"
minlower="${new_var}"

if [ "${change_needed}" != "0" ]; then
   ...
```

check_os_min_version

この関数は、複数の Solaris OS リリースに搭載されている機能を検出するときに使用します。OS の最小リリースバージョンを示す引数を 1 つだけとります。対象プラットフォーム上の OS の実際のリリースバージョンが引数の値以上である場合、0 の値を返し、そうでない場合は 1 を返します。エラーが発生した場合は、255 を返します。

たとえば、この関数はコード例 2-3 に示すように使用することができます。

コード例 2-3　　複数の OS リリースに搭載されている機能の検出

```
if check_os_min_revision 5.10 ; then
    disable_service svc:/network/dns/server:default
elif check_os_min_revision 5.7 ; then
    disable_conf_file ${JASS_ROOT_DIR}etc named.conf
else
    disable_conf_file ${JASS_ROOT_DIR}etc named.boot
fi
```

この例では、まず Solaris 10 OS で使用可能であった SMF FMRI を使用して、ドメインネームシステム (DNS) サービスが無効になっています。それ例外の場合、Solaris 7 OS で DNS を無効にするには `/etc/named.conf` の名前を変更し、Solaris 2.6 OS またはそれ以前では `/etc/named.boot` の名前を変更します。

check_os_revision

この関数は、特定の OS バージョン、または値の範囲をチェックするときに使用します。1 つまたは 2 つの引数をとることができます。引数を 1 つ指定したときは、対象のオペレーティングシステムのバージョンが引数と同じ場合にのみ `0` を返し、そうでない場合には `1` を返します。

同様に、引数を 2 つ指定したときには、対象のオペレーティングシステムのバージョンが 2 つの値の範囲内に含まれている場合に、結果が `0` になります。いずれの場合も、エラーが発生したときは `255` を返します。

たとえば、この関数は コード例 2-4 に示すように使用することができます。

コード例 2-4　　特定の OS バージョンや範囲のチェック

```
if check_os_revision 5.5.1 5.8; then
   if [ "${JASS_DISABLE_MODE}" = "conf" ]; then
      disable_conf_file ${JASS_ROOT_DIR}/etc asppp.cf
   elif [ "${JASS_DISABLE_MODE}" = "script" ]; then
      if [ "${JASS_KILL_SCRIPT_DISABLE}" = "1" ]; then
         disable_rc_file ${JASS_ROOT_DIR}/etc/rcS.d K50asppp
         disable_rc_file ${JASS_ROOT_DIR}/etc/rc0.d K47asppp
         disable_rc_file ${JASS_ROOT_DIR}/etc/rc0.d K50asppp
         disable_rc_file ${JASS_ROOT_DIR}/etc/rc1.d K47asppp
         disable_rc_file ${JASS_ROOT_DIR}/etc/rc1.d K50asppp
      fi
      disable_rc_file ${JASS_ROOT_DIR}/etc/rc2.d S47asppp
   fi
```

コード例 2-4　特定の OS バージョンや範囲のチェック (続き)

```
else
   logInvalidOSRevision "5.5.1-5.8"
fi
```

この例では、対象 OS バージョンが Solaris OS バージョン 2.5.1 (SunOS 5.1) から 8 (SunOS 5.8) までの範囲内に含まれている場合に、JASS_DISABLE_MODE の値に基づいて、スクリプトと構成ファイルのみを無効にしています。

check_readOnlyMounted

この関数は、指定したファイルが読み取り専用ファイルシステムでマウントされているかどうかを判断するときに使用します。

引数:　$1 - チェックするファイル

戻り値:　255 - エラー発生
0 - ファイル $1 が存在するファイルシステムが読み取り専用でマウントされている。
1 - ファイル $1 が存在するファイルシステムが読み取り専用でマウントされていいない。

使用例 :

```
check_readOnlyMounted /usr/bin/ls
```

checksum

この関数は、ファイルのチェックサムの計算に使用します。この関数は、チェックサムが計算されるファイルを示す単一文字列値を引数としてとります。

- Solaris 10 OS では、この関数は Solaris digest プログラムを使用して MD5 チェックサムを計算します。
- Solaris 9 OS またはそれ以前では、この関数ではチェックサムの計算に Solaris cksum プログラムを使用しており、*checksum*:*number of octets* という形式で値を出力します。

コード例 2-5　Solaris 10 OS での MD5 からのチェックサム出力

```
checksum file-name
5b7dff9afe0ed2593f04caa578a303ba
```

convert_inetd_service_to_frmi

この関数は、/etc/inet/inetd.conf ファイルの inetd サービス名を、inetconv(1M) コマンドで使用する SMF FMRI に変換します。この関数は、SMF FMRI に対してではなく、/etc/inet/inetd.conf でレガシー inetd サービス名のみを使用します。変換された FMRI は標準出力に出力されます。

引数: $1 - 変換する inetd サービス名

戻り値: 0 - 成功
1 - 失敗

使用例 :

```
tooltalk_fmri=`convert_inetd_service_to_fmri 100083`
```

copy_a_dir

この関数は、ディレクトリの内容を繰り返しコピーするときに使用します。コピー元のディレクトリ名とコピー先のディレクトリ名の 2 つの引数をとります。この関数は、コピー元ディレクトリの内容を、コピー先パラメータで指定したディレクトリにコピーします。指定したディレクトリが存在していない場合は、新しいディレクトリを作成します。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

使用例 :

```
copy_a_dir /tmp/test1 /tmp/test2
```

copy_a_file

この関数は、1 つの通常ファイルをまるごとコピーするときに使用します。コピー元のファイル名とコピー先のファイル名の 2 つの引数をとります。この関数は、コピー元ファイルの内容を、コピー先パラメータで指定したファイルにコピーします。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

使用例 :

```
copy_a_file /tmp/test-file-a /tmp/test-file-b
```

copy_a_symlink

この関数は、シンボリックリンクを対象のプラットフォームにコピーするときに使用します。コピー元のリンク名とコピー先のファイル名の 2 つの引数をとります。この関数は、コピー先パラメータとして渡された新しいファイル名を使用して、指定されたコピー元リンクから新しいシンボリックリンクを作成します。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

使用例 :

```
copy_a_symlink /tmp/test-link-a /tmp/test-link-b
```

copy_files

この関数は、JASS_HOME_DIR/Files ディレクトリツリーにある 1 組のファイルシステムオブジェクトを、対象のシステムにコピーするときに使用します。この関数では、次のコピー関数を使用し、元に戻す処理を行ったときに確実に変更が元に戻るようにしています。

- copy_a_dir
- copy_a_file
- copy_a_symlink

通常ファイル、ディレクトリ、およびシンボリックリンクのコピーが可能です。

使用例 :

```
copy_files /etc/init.d/nddconfig
```

```
copy_files "/etc/init.d/nddconfig /etc/motd /etc/issue"
```

環境変数により指定される値を含むファイル名に付加されているタグに基づいてファイルを選択的にコピーできることで、この関数は機能を拡張しています。(すべての環境変数の詳細は、第 7 章を参照してください。)

コピーするファイルは、次の条件で選択され、照合に使用される優先度順にリストされています。たとえば、ホスト固有ファイルと汎用ファイルの両方が存在し、対象システムの名前がホスト固有ファイルで定義されているホスト名と一致する場合には、ホスト固有ファイルが使用されます。次の例では、JASS_HOME_DIR 環境変数で指定された /opt/SUNWjass をホームディレクトリとして使用しますが、ユーザーによっては別のホームディレクトリを指定している場合があります。この例では、検索対象のディレクトリツリーは /opt/SUNWjass/Files/ です。

注 – copy_files 関数では、リストに指定されていても *JASS_HOME_DIR*/Files ディレクトリツリーで検出されないオブジェクトは無視します。

1.ホスト固有のバージョン - /opt/SUNWjass/Files/*file*.*JASS_HOSTNAME*

このオプションでは、ホスト対象プラットフォームの名前と JASS_HOSTNAME 環境変数で指定した値が一致する場合にのみ、ファイルがコピーされます。たとえば、ファイル名が etc/issue で JASS_HOSTNAME が eng1 である場合、この条件の下でコピーされるファイルは次のとおりです。

/opt/SUNWjass/Files/etc/issue.eng1

2. キーワード + OS 固有のバージョン - /opt/SUNWjass/Files/*file*-*JASS_FILE_COPY_KEYWORD*+*JASS_OS_VERSION*

このオプションでは、キーワードと OS バージョンの名前が、JASS_FILE_COPY_KEYWORD 環境変数および JASS_OS_VERSION 環境変数で指定した値と一致する場合にのみ、ファイルがコピーされます。

たとえば、検索されるファイルが /etc/hosts.allow であり、JASS_FILE_COPY_KEYWORD が「secure」(secure.driver の場合) であり、かつ JASS_OS_VERSION が 5.10 である場合、この条件の下でコピーされるファイルは次のとおりです。

/opt/SUNWjass/Files/etc/hosts.allow-secure+5.10

3. キーワード固有のバージョン - /opt/SUNWjass/Files/*file*-*JASS_FILE_COPY_KEYWORD*

このオプションでは、JASS_FILE_COPY_KEYWORD 環境変数で指定された値にキーワードが一致する場合にのみ、ファイルがコピーされます。たとえば、JASS_FILE_COPY_KEYWORD が「server」である場合、この条件の下でコピーされるファイルは次のとおりです。

/opt/SUNWjass/Files/etc/hosts.allow-server

4. OS 固有のバージョン - /opt/SUNWjass/Files/*file*+*JASS_OS_REVISION*

このオプションでは、対象プラットフォームの OS リビジョンと、JASS_HOSTNAME 環境変数で指定した値が一致する場合にのみ、ファイルがコピーされます。たとえば、検索されるファイルが /etc/hosts.allow であり、JASS_OS_REVISION が「5.10」である場合、この条件の下でコピーされるファイルは次のとおりです。

/opt/SUNWjass/Files/etc/hosts.allow+5.10

5. 汎用バージョン - /opt/SUNWjass/Files/*file*

このオプションでは、ファイルが対象システムにコピーされます。

たとえば、ファイル名が etc/hosts.allow である場合、この条件の下でコピーされるファイルは次のとおりです。

/opt/SUNWjass/Files/etc/hosts.allow

6. コピー元ファイルのサイズが 0 - ファイルの長さまたはサイズがゼロの場合には、そのファイルはシステムにコピーされません。

create_a_file

この関数は、対象システム上に空ファイルを作成するときに使用します。touch、chown、および chmod コマンドを組み合わせて使用し、特定の所有者、グループ、およびアクセス権を持つ空ファイルを作成します。

注 – 存在するファイルに対するアクセス権や所有権は変更しません。

特定のアクセス権を持つファイルを作成します。

使用例 :

```
create_a_file -o guppy:staff -m 750 /usr/local/testing
```

この例では、/usr/local ディレクトリに、guppy と、グループ staff に所有され、アクセス権 750 を持つ testing という名前のファイルが作成されます。この関数では、表 2-5 に記載されているオプションを使用します。

表 2-5 create_a_file コマンドのオプション

オプション	有効な入力
[-o user[:group]]	chown(1) の構文に従い、user と user:group を使用します。
[-m perms]	chmod(1) の構文に従い、perms を使用します。
/some/fully/qualified/path/file	ファイルへの絶対パス

使用例 :

```
create_a_file /usr/local/testing
```

```
create_a_file -o root /usr/local/testing

create_a_file -o root:sys /usr/local/testing

create_a_file -o root -m 0750 /usr/local/testing
```

create_file_timestamp

この関数は、指定したファイルとすべてのファイルバックアップ操作に対して一意の時刻表示値を作成するときに使用します。一意の接尾辞値が必要となる、バックアップ済みファイルのバックアップを作成する場合に、この関数は便利です。作成される時刻表示値の形式は、JASS_TIMESTAMP と同じ形式です。この関数で作成された時刻表示値は、JASS_SUFFIX 環境変数に格納されます。詳細については、第 7 章の 255 ページの「JASS_TIMESTAMP」を参照してください。

使用例 :

```
create_file_timestamp /usr/local/testing
```

disable_conf_file

この関数は、サービス構成ファイルを無効にするときに使用します。この関数は、このファイルが格納されているディレクトリ名と、サービス構成ファイル名を示す 2 つの文字列値を引数として受け取ります。下線 (_) の接頭辞をファイル名の先頭に付加することで、サービス構成ファイルを無効にして、このファイルが実行されないようにします。

使用例 :

```
disable_conf_file /etc/dfs dfstab
```

この例では、ファイル名を /etc/dfs/dfstab から /etc/dfs/_dfstab.JASS.*timestamp* という名前に変更しています。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

disable_file

この関数は、元のディレクトリに格納できないファイルを無効にするときに使用します。たとえば、/var/spool/cron/crontabs ディレクトリに、個人ユーザーの crontab ファイルが保存されているとします。無効化またはバックアップした crontab ファイルのコピーを crontabs ディレクトリに格納すると、cron サービスではエラーになります。それは、無効化またはバックアップしたファイルの名前と一致するユーザー名がないためです。

この問題に対処するには、この関数で .JASS 接尾辞が付いたミラーディレクトリを作成し、この中に無効になったファイルを格納します。たとえば、無効にするファイルが /var/spool/cron/crontabs ディレクトリ内にある場合には、/var/spool/cron/crontabs.JASS ディレクトリを作成し、ここに無効になったファイルを移動します。

他の無効化関数と同様に、無効にするファイルには .JASS.*timestamp* という接尾辞が付きます。ただし、この関数を使用すると、無効になったファイルは元のファイルと同じディレクトリには格納されません。

使用例：

```
disable_file /var/spool/cron/crontabs/uucp
```

この例では、ファイル /var/spool/cron/crontabs/uucp を /var/spool/cron/crontabs.JASS ディレクトリに移動して、uucp.JASS.*timestamp* という名前に変更しています。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

disable_rc_file

この関数は、実行コントロールファイルの実行を無効にするときに使用します。この関数は、スクリプトが格納されているディレクトリ名と、実行コントロールスクリプト名を示す 2 つの文字列値を引数として受け取ります。ファイルを実行するときは、開始実行コントロールスクリプトの名前を S で始め、終了実行コントロールスクリプトの名前を Kで始めるようにします。下線 (_) の接頭辞をファイル名の先頭に付加することで、スクリプトを無効にして、実行コントロールフレームワークでこのファイルが実行されないようにします。また、無効にしたファイルには接尾辞 .JASS.*timestamp* を付加します。

使用例：

```
disable_rc_file /etc/rc2.d S71rpc
```

この例では、ファイル名を /etc/rc2.d/S71rpc から /etc/rc2.d/_S71rpc.JASS.*timestamp* という名前に変更しています。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

disable_service

注 – この関数は、Solaris 10 の SMF でのみ使用します。

この関数は、特定の FMRI リスト上のすべての SMF サービスを無効にするときに使用します。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

引数: $1 - 無効にする 1 つ以上の SMF サービスの FMRI

戻り値: なし

使用例 :

```
disable_service "svc:/application/x11/xfs:default"
```

enable_service

注 – この関数は、Solaris 10 の SMF でのみ使用します。

この関数は、特定の FMRI リスト上のすべての SMF サービスを有効にするときに使用します。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

引数: $1 - 有効にする 1 つ以上の SMF サービスの FMRI

戻り値: なし

使用例 :

```
enable_service "svc:/network/ipfilter:default"
```

find_sst_run_with

この関数は、指定されたキーワード値ペアで、まだアクティブな最新の Solaris Security Toolkit 処理を検索するときに使用します。キーワード値ペアの格納と取得の詳細については、`set_stored_keyword_val` (74 ページの「set_stored_keyword_val」) および `get_stored_keyword_val` (67 ページの「get_stored_keyword_val」) を参照してください。

この関数は、システム上の取り消されていないすべての Solaris Security Toolkit の実行を検索します。実行のいずれかで `set_stored_keyword_val` コマンドを使用して、検索対象のキーワード値ペアを格納している場合、関数は最新のタイムスタンプを返します。このコマンドを使用している実行が存在しない場合、何も返されません。

引数: `$1` - チェックするキーワード
`$2` - 検索対象の値

戻り値: そのスクリプトとキーワード値ペアを使用している最新のアクティブな実行のタイムスタンプを出力し、そのような実行が見つからなかった場合は "" を出力する。

使用例 :

```
last_date=`find_sst_run_with MY_STORED_VALUE 17`
```

get_expanded_file_name

この関数は、60 ページの「copy_files」で説明されているタグ拡張ファイル名を返すときに使用します。

引数: `$1` - ファイル名

戻り値: 拡張ファイル名、または空 (ファイル名が一致しなかった場合)

使用例 :

```
get_expanded_file_name /etc/motd
```

この例では、システム `jordan` 上で関数が実行されたときにファイル `JASS_HOME/Files/etc/motd.jordan` が存在していた場合、`/etc/motd.jordan` が返されます。

get_stored_keyword_val

この関数は、保存ファイルから、格納されているキーワード値ペアを取得するときに使用します。使用する保存ファイルのデフォルトは JASS_RUN_VALUES ファイルですが、独自のファイル名を指定できます。

引数: $1 - チェックするキーワード
$2 - リポジトリ名、デフォルトは空白

戻り値: 0 - キーワードを検出。RETURN_VALUE はファイル内でその値に設定済み
1 - ファイルが見つからない。
2 - キーワードはファイル内で設定されていない。

使用例 :

```
if get_stored_keyword_val MY_STORED_VALUE; then
...
```

get_users_with_retries_set

この関数は、lock_after_retries が設定されている user_attr エントリを持つパスワード付きのユーザーアカウントを取得するときに使用します。この関数は、監査スクリプトと終了スクリプトの両方で便利です。155 ページの「enable-account-lockout.fin」または 205 ページの「enable-account-lockout.aud」を参照してください。

引数: $1 - 除去するユーザーのリスト

戻り値: パスワードと lock_after_retries が設定されているユーザーのリスト。

使用例 :

```
user_list=`get_users_with_retries_set "root"`
```

is_patch_applied と is_patch_not_applied

この 2 つの関数は、パッチがシステムに適用されているかどうかを判定するときに使用します。これらの関数は、チェックするパッチ番号を示す単一文字列値を引数として受け取ります。

この値は、次のいずれかの方法で指定できます。

- 「123456」のようにパッチ番号を指定します。対象システムにパッチがインストールされているときは、0 が表示されます。パッチがインストールされていないときは、1 が表示されます。

  使用例 :

```
is_patch_applied 123456
```

- 「13456-03」のようにパッチ番号とバージョン番号を指定します。システムにパッチがインストールされ、かつそのパッチが指定したバージョン以上であるときは、値 0 が表示されます。システムにパッチが存在しないときは、1 が表示されます。パッチがインストールされていても、そのバージョンが指定した値より小さいときは、2 が表示されます。

  使用例 :

```
is_patch_applied 123456-02
```

is_service_enabled

注 – この関数は、Solaris 10 の SMF でのみ使用します。

この関数は、SMF サービスが有効であるかどうかを判断するときに使用します。

引数: $1 - チェックする SMF サービスの FMRI

戻り値: 0 - サービスは有効であるか、再起動後有効になる。
1 - サービスは無効でアップグレードマニフェストには有効化スクリプトが存在しない、または FMRI が認識されていない。

使用例 :

```
is_service_enabled "svc:/network/ipfilter:default"
```

is_service_installed

注 – この関数は、Solaris 10 の SMF でのみ使用します。

この関数は、SMF サービスがインストールされているかどうかを判断するときに使用します。スタンドアロンモードでは、SMF コマンドが検証を実行します。JumpStart モードでは、サービスマニフェストの `.xml` ファイルを検索することで検証が実行されます。

引数: `$1` - チェックする SMF サービスの FMRI

戻り値: `0` - サービスはインストールされている (スタンドアロンモード)、またはサービスマニフェストが存在する (JumpStart モード)。
`1` - サービスはインストールされていない (スタンダードモード)、サービスマニフェストが存在しない (JumpStart モード)、または FMRI が認識されていない。

使用例 :

```
is_service_installed "svc:/network/ipfilter:default"
```

is_service_running

注 – この関数は、Solaris 10 の SMF でのみ使用し、JumpStart モードでは使用できません。

この関数は、SMF サービスが実行中であるかどうかを判断するときに使用します。

引数: `$1` - チェックするサービスの FMRI

戻り値: `0` - サービスは実行中
`1` - サービスは実行中ではない

使用例 :

```
is_service_running "svc:/network/ipfilter:default"
```

is_user_account_extant

注 – この関数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この関数は、あるユーザーアカウントが存在するかどうかを判断するときに使用します。

引数: `$1` - チェックするユーザーアカウント名

戻り値: `0` - ユーザーアカウントは存在する
`1` - ユーザーアカウントは存在しない

使用例 :

```
is_user_account_extant "nuucp"
```

is_user_account_locked

注 – この関数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この関数は、ユーザーアカウントがパスワードファイル内でロックされているかどうかをチェックするときに使用します。

引数: `$1` - チェックするユーザーアカウント名

戻り値: `0` - ユーザーアカウントはロックされている
`1` - ユーザーアカウントはロックされていない

使用例 :

```
is_user_account_locked "nuucp"
```

is_user_account_login_not_set

注 – この関数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この関数は、あるユーザーアカウントにパスワードが設定されているかどうかをチェックするときに使用します。

引数: $1 - チェックするユーザーアカウント名

戻り値: 0 - ユーザーパスワードは「NP」ではない
1 - ユーザーパスワードは「NP」である

この関数から「NP」(パスワードなし) が返された場合、そのユーザーにはパスワードは定義されておらず、パスワードなしでログインできます。実際にユーザーがパスワードなしでログインできるかどうかは、ユーザーのログイン方法と、そのログインメカニズムのセキュリティー制限によって決まります。たとえば、Secure Shell のデフォルト構成では、パスワードを持っていないユーザーのログインを許可しません。

使用例 :

```
is_user_account_login_not_set "root"
```

is_user_account_passworded

注 – この関数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この関数は、あるユーザーアカウントが /etc/shadow ファイルにパスワードエントリを持っているかどうかを確認するときに使用します。

引数: $1 - チェックするユーザーアカウント名

戻り値: 0 - ユーザーアカウントはパスワードファイル内にある
1 - ユーザーアカウントはパスワードファイル内にない

使用例 :

```
is_user_account_passworded "root"
```

lock_user_account

注 – この関数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この関数は、あるユーザーアカウントをロックするときに使用します。

引数: $1 - ロックするユーザーアカウント名

戻り値: なし

使用例：

```
lock_user_account “nuucp”
```

make_link

この関数は、シンボリックファイルリンクを作成するときに使用します。この関数では、元に戻す処理を行ったときに元に戻せるように、必要なマニフェスト項目を自動的に追加します。

引数: $1 - ソースリンクファイル名
$2 - 宛先リンクファイル名

戻り値: なし

使用例：

```
make_link ../lib/sendmail ${JASS_ROOT_DIR}usr/bin/newaliases
```

mkdir_dashp

この関数は、対象システム上にディレクトリを新規作成するときに使用します。この関数は、作成するディレクトリ名を示す単一文字列値を引数として受け取ります。mkdir に -p オプションを付けると、対象のディレクトリが存在している場合にエラーが報告されなくなります。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

使用例：

```
mkdir_dashp /usr/local
```

move_a_file

この関数は、ファイル名を別のファイル名に移動するときに使用します。元のファイル名と移動先のファイル名の 2 つのエントリが必要です。元のファイル名を、移動先パラメータで指定したファイル名に移動、つまり変更します。この関数では、元に戻す処理を行ったときにこの操作を元に戻せるように、必要なマニフェスト項目を自動的に追加します。

使用例 :

```
move_a_file /tmp/test-file-a /tmp/test-file-b
```

rm_pkg

この関数は、Solaris OS パッケージをシステムから削除するときに使用します。この関数で実行した操作は確定されるので、元に戻す処理で復元することはできません。表 2-6 に、この関数のオプションを示します。

表 2-6 rm_pkg 関数のオプション

オプション	説明
-a *ask_file*	pkgrm ask ファイル名。デフォルトでは、他のファイルを指定しない場合、pkgrm ask ファイル (noask_pkgrm) が使用されます。
-o *options*	pkgrm コマンドオプション
package	削除するパッケージ

使用例 :

```
rm_pkg SUNWadmr
```

set_service_property_value

注 – この関数は、Solaris 10 の SMF でのみ使用します。

この関数は、SMF サービスのプロパティー値を設定するときに使用します。

引数: $1 - サービスの FMRI
$2 - 設定するプロパティー名
$3 - 設定するプロパティー値

戻り値: なし

使用例 :

```
set_service_property_value "svc:/network/inetd" "defaults/tcp_wrappers" "true"
```

set_stored_keyword_val

この関数は、格納されているキーワード値ペアを保存ファイルに設定するときに使用します。使用されるデフォルトのファイルは JASS_RUN_VALUES ファイルです。

引数: $1 - 設定するキーワード
$2 - 設定する値

戻り値: 0 - キーワードは設定されている。ファイル内にすでに存在しているキーワードが設定されている場合、古い値は上書きされます
1 - ファイル書き込み時に問題発生。

使用例 :

```
get_stored_keyword_val MY_STORED_VALUE 23
```

unlock_user_account

注 – この関数は、Solaris 10 の SMF でのみ使用します。

この関数は、あるユーザーアカウントのロックを解除するときに使用します。この関数では、元に戻す処理を行ったときに元に戻せるように、必要なマニフェスト項目を自動的に追加します。

引数: $1 - ロックを解除するユーザーアカウント名

戻り値: なし

使用例 :

```
unlock_user_account "adm"
```

update_inetconv_in_upgrade

この関数は、再起動後に実行されるアップグレードファイルに inetconv(1M) コマンドを実行する指示を書き込むときに使用します。inetconv コマンドは inetd.conf エントリを SMF リポジトリにインポートします。この関数では、元に戻す処理を行ったときに元に戻せるように、必要なマニフェスト項目を自動的に追加します。

引数: なし

戻り値: 0 - 成功
1 - 失敗

使用例 :

```
update_inetconv_in_upgrade
```

warn_on_default_files

この関数は、ユーザーによって変更されていない Solaris Security Toolkit ソフトウェア内の任意のファイルに関する logWarning コマンドを発行するときに使用します。これらのファイルは Solaris Security Toolkit によりインストール可能ですが、構成が不完全な場合、予期せぬ結果を招くため、これらのファイルをチェックしてファイルが想定どおりであることを確認する必要があります。ファイルを変更したり、ソフトウェアに付属していないカスタムバージョンを使用したりすると、警告は出されません。

引数: ${1} - チェックする 1 つ以上のファイル

接頭辞なしのフロントスラッシュルート (/) を起点とした完全指定のインストール済みターゲットパスを指定します。
たとえば、/etc/motd のように指定します。

戻り値: なし

使用例 :

```
warn_on_default_files /etc/opt/ipc/ipf.conf
```

write_val_to_file

この関数は、等号 (=) で区切られている名前値ペアをファイルに書き込むときに使用します。値が NULL である場合、何も書き込まれません。この関数は check_and_log_change_needed 関数と一緒に使用してください (56 ページの「check_and_log_change_needed」を参照)。

引数: $1 - ファイル名
$2 - ファイル内のキーワード
$3 - 新しい値

戻り値: なし

使用例：

```
write_val_to_file /etc/default/passwd MINALPHA 7
```

監査関数の使用

このソフトウェアの監査関数には、プライベートとパブリックの 2 種類が用意されています。audit_private.funcs ファイルで定義されている関数はプライベートであるため、パブリックには使用できません。このファイルで定義されているプライベートスクリプトは使用しないでください。audit_public.funcs ファイルで定義されているパブリックスクリプトだけを使用してください。

パブリック関数は、監査スクリプトで使用される監査関数を定義する関数で、JASS_AUDIT_DIR ファイルに格納されています。このファイルで定義されている関数はパブリックなので、標準およびカスタムの監査スクリプトのどちらででも自由に使用できます。多くの場合、このファイルで定義されている関数は、audit_private.funcs ファイルで定義されている関数を呼び出すスタブです。今後のリリースでの元のコードの変更や拡張について心配することなく、ユーザーがこれらのスクリプトをパブリックインタフェースにコーディングできるように、これらのスタブが実装されています。

監査関数は、システムに格納されている構成と実行時構成の構成要素を評価する、監査スクリプトの一部として使用します。次の関数は、Solaris Security Toolkit ソフトウェアの監査フレームワークに対するパブリックインタフェースです。

監査スクリプトをカスタマイズまたは作成する場合は、次の関数を使用して標準の操作を実行してください。

check_fileContentsExist と check_fileContentsNotExist

この 2 つの関数は、指定ファイルに、指定した検索文字列と一致する内容が含まれているかどうかを判定するときに使用します。検索文字列は、正規表現の形式で指定します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるファイルの名前を示す文字列値
- 検索パターンを示す文字列値
- 監査チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、log_FileContentsExist 関数または log_FileContentsNotExist 関数のいずれかにより自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、環境変数が LOG に設定されている場合は、この情報がそのままログ関数に渡されます。

使用例 :

```
check_fileContentsExist /etc/default/inetinit "TCP_STRONG_ISS=2" 1 LOG
```

check_fileExists と check_fileNotExists

この 2 つの関数は、対象システム上にファイルが存在しているかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるファイルの名前を示す文字列値
- 監査チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、環境変数が LOG に設定されている場合は、この情報がそのままログ関数に渡されます。

使用例 :

```
check_fileExists /etc/inet/inetd.conf 1 LOG
```

check_fileGroupMatch と check_fileGroupNoMatch

この 2 つの関数は、ファイルが対象システム上のグループに属しているかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるファイルの名前を示す文字列値
- チェックするグループを示す文字列値。グループの値には、名前またはグループ識別子 (GID) を指定できます。グループ名が数値でネームサービステーブルにない場合は、この数値が GID となります。
- 監査チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、環境変数が LOG に設定されている場合は、この情報がそのままログ関数に渡されます。

使用例 :

```
check_fileGroupMatch /etc/passwd sys 1 LOG

check_fileGroupMatch /etc/passwd 3 1 LOG
```

check_fileModeMatch と check_fileModeNoMatch

この 2 つの関数は、対象システム上でファイルにアクセス権が指定されているかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるファイルの名前を示す文字列値
- チェックするモード、つまりアクセス権を示す文字列値。アクセス権の値には、記号値と 8 進数値のいずれかを指定できます。この関数では、この環境変数に対して、find(1) コマンドの perm オプションを実行するときと同じ値を受け取ります。
- 監査チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、環境変数が LOG に設定されている場合は、この情報がそのままログ関数に渡されます。

使用例 :

```
check_fileModeMatch /etc/passwd "0444" 1 LOG

check_fileModeMatch /etc/passwd "ugo=r" 1 LOG
```

check_fileOwnerMatch と check_fileOwnerNoMatch

この 2 つの関数は、ファイルが対象システム上の特定のユーザーに属しているかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるファイルの名前を示す文字列値
- チェックするユーザーを示す文字列値。ユーザーの値には、ユーザー名とユーザー識別子のいずれかを指定できます。
- 監査チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。この引数にその他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、上記の環境変数が LOG に設定されているときに、この情報がそのままログ関数に渡されます。

使用例 :

```
check_fileOwnerMatch /etc/passwd root 1 LOG

check_fileOwnerMatch /etc/passwd 0 1 LOG
```

check_fileTemplate

この関数は、Solaris Security Toolkit ソフトウェアで定義されているファイルテンプレートが、対象システムにインストールされているファイルテンプレートと一致しているかどうかを判定するときに使用します。たとえば、この関数を使用してファイルテンプレート /etc/motd をチェックする場合、JASS_FILES_DIR/etc/motd と /etc/motd の内容を比較し、内容が同一であるかどうかを判定します。同じ内容である場合には、成功は 0、失敗は 1、エラー状態は 255 と表示されます。複数のファイルを指定する場合には、すべてのファイルが表示コード 0 を取得する必要があります。

次の引数を指定できます。

- 調べるファイルの名前、またはファイルをスペースで区切ったリスト (たとえば、a b c) を示す文字列値
- チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数

- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、環境変数が LOG に設定されている場合は、この情報がそのままログ関数に渡されます。

使用例 :

```
check_fileTemplate /etc/motd 1 LOG
```

check_fileTypeMatch と check_fileTypeNoMatch

この 2 つの関数は、ファイルシステムオブジェクトが、対象システム上の特定のオブジェクトであるかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるファイルの名前を示す文字列値
- チェックするファイルタイプを示す文字列値。使用可能なタイプについての詳細は、24 ページの「logFileTypeMatch と logFileTypeNoMatch」を参照してください。

  検出されるファイルタイプを、表 2-7 に示します。

表 2-7 check_fileTypeMatch 関数で検出されるファイルタイプ

ファイルタイプ	説明
b	ブロック型特殊ファイル
c	文字型特殊ファイル
d	ディレクトリ
D	door
f	通常ファイル
l	シンボリックリンク
p	名前付きパイプ (先入れ先出し)
s	ソケット

- チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、環境変数が LOG に設定されている場合は、この情報がそのままログ関数に渡されます。

使用例 :

```
check_fileTypeMatch /etc/passwd f 1 LOG

check_fileTypeMatch /etc d 1 LOG
```

check_if_crontab_entry_present

この関数は、crontab エントリ $2 がユーザー $1 の crontab ファイル内に存在するかどうかをチェックするときに使用します。

引数: $1 - crontab ファイルの所有者のユーザー名
$2 - crontab テーブル内でチェックするプログラム名

戻り値: 0 - $2 は存在する
0 以外 - crontab エントリ $2 が見つからないか
crontab ファイルが見つからない

使用例 :

```
check_if_crontab_entry_present adm /usr/lib/acct/ckpacct
```

check_keyword_value_pair

この関数を使用すれば、共通の監査作業である、ファイル内のキーワード値ペアのチェックがより便利になります。キーワードは、等号 (=) によって値から分離された、行の最初の空白文字以外の文字である必要があります。チェック対象のファイルが存在する必要があります。存在しないと、関数の動作が未定義になります。

引数: $1 - チェックするファイル
$2 - $3 内の値と照合してチェックするキーワード
$3 - $2 内のキーワードと照合してチェックする値

戻り値: なし

使用例 :

```
check_keyword_value_pair {JASS_ROOT_DIR}etc/security/policy.conf
CRYPT_DEFAULT 1
```

check_minimized

この関数は、最小化プラットフォームでのみパッケージチェックを実行する必要があるときに使用します。最小化プラットフォームとは、不要な Solaris OS パッケージを削除したプラットフォームです。この関数は、JASS_CHECK_MINIMIZED 環境変数により動作が制御されることを除けば、check_packagesNotExist 関数と同じです。対象システムが最小化されていない場合には、JASS_CHECK_MINIMIZED 環境変数を 0 に設定する必要があります。この場合、いずれのチェックも行われず、チェックが実行されなかったという通知とともに、値 0 が表示されます。対象システムが最小化されている場合には、check_packageNotExists 関数と同様に動作し、成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるパッケージの名前を示す文字列値
- チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、上記の環境変数が LOG に設定されているときに、この情報がそのままログ関数に渡されます。

使用例 :

```
check_minimized SUNWatfsu 1 LOG
```

check_minimized_service

注 – この関数は、Solaris 10 OS を実行しているシステム上の SMF でのみ使用します。

この関数は、インストールされていないサービスをチェックするときに使用します。この関数を使用するのは、パッケージの存在が必ずしもエラーではない場合、たとえばシステムが最小化されていない場合などの特別なケースです。この関数は、環境変数 JASS_CHECK_MINIMIZED = 1 (詳細は第 7 章を参照) により制御されます。

引数: $1 - services - チェックするサービスのリスト
$2 - vulnValue - 脆弱性値 (整数)
$3 - logStatus - ログ状態 (オプション)

戻り値: 255 - エラーが発生した場合、または指定された引数が無効である場合
0 - パッケージがまったく存在しない場合
1 - 1 つ以上のパッケージが存在する場合

使用例 :

```
check_minimized_service "svc:/network/finger:default" 1 LOG
```

check_packageExists と check_packageNotExists

この 2 つの関数は、対象システムにソフトウェアパッケージがインストールされているかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるパッケージの名前を示す文字列値
- 監査チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、環境変数が LOG に設定されている場合は、この情報がそのままログ関数に渡されます。

使用例 :

```
check_packageExists SUNWsshdu 1 LOG
```

check_patchExists と check_patchNotExists

この 2 つの関数は、対象システムにソフトウェアのパッチがインストールされているかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるパッチの名前を示す文字列値
- チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、環境変数が LOG に設定されている場合は、この情報がそのままログ関数に渡されます。

使用例 :

```
check_patchExists 123456 1 LOG

check_patchExists 123456-01 1 LOG
```

注 – パッチのバージョンを指定することができます。指定する場合は、インストールされているバージョンが、指定するバージョン以上でなければなりません。バージョンを指定しない場合は、いずれかのバージョンのパッチがインストールされていれば、成功と表示されます。

check_processArgsMatch と check_processArgsNoMatch

この 2 つの関数は、特定の実行時引数を使用するプロセスが、システム上で実行されているかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるプロセスの名前を示す文字列値
- チェックする実行時引数を示す文字列値

- チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、上記の環境変数が LOG に設定されているときに、この情報がそのままログ関数に渡されます。

使用例 :

```
check_processArgsMatch /usr/sbin/syslogd "-t" 1 LOG
```

check_processExists と check_processNotExists

この 2 つの関数は、対象システムでプロセスが実行されているかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるプロセスの名前を示す文字列値
- チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、上記の環境変数が LOG に設定されているときに、この情報がそのままログ関数に渡されます。

使用例 :

```
check_processExists sshd 1 LOG
```

check_serviceConfigExists と check_serviceConfigNotExists

この 2 つの関数は、対象システムにサービス構成ファイルが存在しているかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べるサービス構成ファイルの名前を示す文字列値
- チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、上記の環境変数が LOG に設定されているときに、この情報がそのままログ関数に渡されます。

使用例 :

```
check_serviceConfigExists /etc/ssh/sshd_config 1 LOG
```

check_serviceDisabled と check_serviceEnabled

注 – この 2 つの関数は、Solaris 10 の SMF でのみ使用します。

この 2 つの関数は、サービスのリストをチェックして、各サービスが無効と有効のどちらであるかを確認するときに使用します。

引数: $1 - services - チェックするサービスのリスト
$2 - vulnValue - 脆弱性値
$3 - logStatus - ログ状態 (失敗時にログが必要である場合は LOG に設定)
$4 - relatedInfo - 関連情報の文字列

戻り値: 255 - エラーが発生した場合、または指定された引数が無効である場合
0 - すべてのサービスが無効/有効である場合
1 - 1 つ以上のサービスが無効/有効である場合

check_serviceInstalled と check_serviceNotInstalled

注 – この 2 つの関数は、Solaris 10 の SMF でのみ使用します。

この関数は、サービスのリストをチェックして、各サービスがインストールされているかどうかを確認するときに使用します。

引数:
- `$1` - `services` - チェックするサービスのリスト
- `$2` - `vulnValue` - 脆弱性値
- `$3` - `logStatus` - ログ状態 (失敗時にログが必要である場合は `LOG` に設定)
- `$4` - `relatedInfo` - 関連情報の文字列

戻り値:
- 255 - エラーが発生した場合、または指定された引数が無効である場合
- `0` - サービスがまったく存在しないまたはすべて存在する場合
- `1` - 1 つ以上のパッケージが存在する、または存在しない場合

使用例 :

```
check_serviceInstalled svc:/network/ssh:default 1 LOG
```

check_serviceOptionEnabled と check_serviceOptionDisabled

注 – この 2 つの関数は、Solaris 10 の SMF でのみ使用します。

この 2 つの関数は、SMF コマンドを使用して、あるサービスのプロパティーのオプションが有効と無効のどちらであるかをチェックするときに使用します。

引数:
- `$1` - チェックするサービスのリスト
- `$2` - チェックするサービスのプロパティー
- `$3` - オプションの前のパターン
- `$4` - オプション
- `$5` - 整数の脆弱性値
- `$6` - 失敗時にログが必要である場合は `LOG` に設定 (オプション)
- `$7` - 関連情報の文字列 (オプション)

戻り値:
- 255 - エラーが発生した場合、または指定された引数が無効である場合
- `0` - サービスオプションは有効 (無効)
- `1` - サービスオプションは無効 (有効)

check_servicePropDisabled

注 – この関数は、Solaris 10 の SMF でのみ使用します。

この関数は、SMF コマンドを使用して、あるサービスのプロパティーのオプションが無効であるかどうかをチェックするときに使用します。

引数: `$1` - FMRI のリスト
`$2` - `property`
`$3` - `vulnvalue`
`$4` - `logStatus`

戻り値: 255 - エラーが発生した場合、または指定された引数が無効である場合
`0` - プロパティーは有効 (無効)
`1` - プロパティーは無効 (有効)

check_serviceRunning と check_serviceNotRunning

注 – この 2 つの関数は、Solaris 10 の SMF でのみ使用します。

この 2 つの関数は、サービスのリストをチェックして、各サービスが実行中であるかどうかを確認するときに使用します。

引数: `$1` - サービスのリスト
`$2` - `vulnvalue`
`$3` - `logStatus`
`$4` - `related Info`

戻り値: 255 - エラーが発生した場合、または指定された引数が無効である場合
`0` - すべてのサービスが実行中である/実行中でない
`1` - 1 つ以上のサービスは実行中ではない

check_startScriptExists と check_startScriptNotExists

この 2 つの関数は、対象システムに実行コントロール開始スクリプトが存在しているかどうかを判定するときに使用します。成功は `0`、失敗は `1`、エラー状態は `255` と表示されます。

次の引数を指定できます。

- 調べる実行コントロール開始スクリプトの名前を示す文字列値
- チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、上記の環境変数が LOG に設定されているときに、この情報がそのままログ関数に渡されます。

使用例 :

```
check_startScriptExists /etc/rc3.d/S89sshd 1 LOG
```

check_stopScriptExists と check_stopScriptNotExists

この 2 つの関数は、対象システムに実行コントロール停止スクリプトが存在しているかどうかを判定するときに使用します。成功は 0、失敗は 1、エラー状態は 255 と表示されます。

次の引数を指定できます。

- 調べる実行コントロール停止スクリプトの名前を示す文字列値
- チェックが失敗したときに使用する脆弱性値を示す 0 以上の整数
- 関数のログ状態を示す文字列値。この値を文字列値 LOG にすると、自動的に結果が記録されます。その他の文字列キーワードを指定した場合は、自動的に記録は行われないため、呼び出し元のプログラムコードで状態メッセージを記録する必要があります。
- PASS または FAIL メッセージの次にユーザーに表示する関連情報を示す文字列値 (オプション)。この引数を使用すると、上記の環境変数が LOG に設定されているときに、この情報がそのままログ関数に渡されます。

使用例 :

```
check_stopScriptExists /etc/rc2.d/K03sshd 1 LOG
```

check_userLocked と check_userNotLocked

注 – この 2 つの関数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この 2 つの関数は、ユーザーアカウントロックされているかどうかをチェックするときに使用します。

引数: `$1` - ユーザー ID

戻り値: 255 - エラーが発生した場合、または指定された引数が無効である場合
`0` - ユーザーがロックされている場合
`1` - ユーザーがロックされていない場合

finish_audit

この関数は、チェックスクリプトのすべてのプロセスが完了したことと、そのスクリプトのスコアを計算する必要があることを通知するときに使用します。通常、この関数はチェックスクリプトの最後のエントリになります。スクリプトの終了を示すメッセージを表示する場合は、この関数に単一文字列を引数として渡します。

使用例 :

```
finish_audit
```

```
finish_audit "End of script"
```

get_cmdFromService

注 – この関数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この関数は、実行中のサービスのコマンドまたはプロセスのリストを取得するときに使用します。

引数: `$1` - サービス名

戻り値: `""` - サービスプロセステストと関連付けられている
プロセスがない場合 - プロセスは
`{ pid user comm [pid user common] }` の形式で特定の
サービスと関連付けられている

使用例 :

```
get_cmdFromService svc:/network/ssh:default
```

start_audit

この関数は、監査スクリプトを呼び出すときに使用します。通常、この関数は監査スクリプトの最初の命令であり、コメントや変数宣言は含まれません。この関数は、スクリプト名を定義し、バナーを表示して、スクリプトスコアを 0 にリセットします。

引数: $1 - 監査スクリプト名
$2 - 監査スクリプトの説明 (複数行にすることも可能で、logFormattedMessage 関数を使用してフォーマットされる。)
$3 - PASS または FAIL メッセージの次にユーザーに表示する関連情報 (オプション)で、logFormattedMessage 関数を使用してフォーマットされる。

戻り値: 255 - エラーが発生した場合、または指定された引数が無効である場合
0 - ユーザーがロックされている場合
1 - ユーザーがロックされていない場合

使用例 :

```
start_audit disable-apache.aud "Apache" "Description of Check"
```

出力例 :

```
#-----------------------------------------------------------------
# Apache
#
# Description of Check
#-----------------------------------------------------------------
```

第3章

ファイルテンプレート

この章では、Solaris Security Toolkit ソフトウェアに含まれているファイルテンプレートを使用、変更、およびカスタマイズする方法について説明します。また、ドライバが関数を処理する方法と、ファイルテンプレートに格納されているその他の情報を処理する方法についても説明します。

この章では、以下の項目を説明します。

ファイルテンプレートのカスタマイズ

ファイルテンプレートは、Solaris Security Toolkit ソフトウェアの重要な構成要素です。テンプレートファイルは、環境変数、OS バージョン番号、キーワード、およびクライアントホスト名を使用して、ユーザーがスクリプトを簡単にカスタマイズおよび配布するためのメカニズムを提供します。`Files` ディレクトリ内のファイルを終了および監査スクリプトと組み合わせて使用すると、利用するセキュリティープロファイル (ドライバ) の設計に応じて、必要な変更を特定することができます。

この節では、`Files` ディレクトリでのファイルの新規作成方法を含む、ファイルテンプレートのカスタマイズ方法と推奨事項について説明します。

ドライバ、終了スクリプト、および監査スクリプトのカスタマイズについては、以下の章を参照してください。

- ドライバのカスタマイズについては、第 4 章を参照。
- 終了スクリプトのカスタマイズについては、第 5 章を参照。
- 監査スクリプトのカスタマイズについては、第 6 章を参照。

注 – カスタマイズしたファイルをより多くのユーザーのために役立てたいときは、拡張機能要求を提出することをご検討ください。Solaris Security Toolkit 開発チームは、ユーザーに役立つようソフトウェアを改善する方法を常に求めております。

▼ ファイルテンプレートをカスタマイズするには

ファイルテンプレート (ファイル) をカスタマイズするには、次の手順を実行します。このカスタマイズでは、Solaris Security Toolkit ソフトウェアのカスタムバージョンを使用可能にするとともに、リリースされたソフトウェアの新バージョンがシステムにインストールされたときに、そのカスタムバージョンが上書きされないようにします。

1. **カスタマイズするファイルとその関連ファイルをコピーします。**
2. **コピーしたファイルを、カスタムファイルとして識別される名前に変更します。**

   推奨事項については、『Solaris Security Toolkit 4.2 管理マニュアル』の第 1 章「Solaris Security Toolkit ソフトウェアの構成およびカスタマイズ」を参照してください。
3. **必要な場合、この一意の名前が付いたファイルが呼び出されるように、カスタムドライバを変更します。**

   次のサンプルコードでは、JASS_FILES 環境変数を変更して、特定のホストにコピーするファイルをカスタマイズしています。

```
JASS_FILES="
[...]
        /etc/init.d/nddconfig
        /etc/rc2.d/S70nddconfig
[...]
"
```

このサンプルコードでは、カスタマイズされたセキュリティー強化用のドライバ abccorp-server-hardening.driver が、カスタムの nddconfig ファイルを使用します。Solaris Security Toolkit ソフトウェアが更新されたら新バージョンで上書きされる可能性のある、元の nddconfig ファイルを変更する代わりに、コピー先システムのホスト名を Files ディレクトリにあるファイル名の最後に付加して、カス

タムの nddconfig スクリプトを作成します。次の例では、スクリプトファイル名にコピー先システムのホスト名が付いた、カスタムの nddconfig スクリプトを示しています。

```
/opt/SUNWjass/Files/etc/init.d/nddconfig.hostname099
```

ここで、hostname099 はシステムのホスト名です。

注 – 場合によっては、ソフトウェアで特定の名前が必要となるために、スクリプト名を変更できないことがあります。その場合には、この章の説明にあるように、接尾辞を使用します。あるいは、コピーを作成し、必要に応じてそのコピーファイルの名前を変更する終了スクリプトを作成します。後者の方法を選択した場合は、コピー操作とファイル名変更操作が、元に戻す処理を行ったときに確実に復元されることを確認してください。変更が復元されるように、ファイル、ドライバ、およびスクリプトをカスタマイズする方法についての詳細は、『Solaris Security Toolkit 4.2 管理マニュアル』の第 4 章を参照してください。

ファイルのコピー方法について

ファイルは、JASS_FILES 環境変数や JASS_FILE_OS_VERSION 環境変数などの、ある種の環境変数を定義した方法に基づいて、自動的に JASS_HOME_DIR/Files ディレクトリからコピーされます。すべての環境変数についての詳細は、第 7 章を参照してください。

Solaris Security Toolkit ソフトウェアは、JASS_HOME_DIR/Files ディレクトリ内の複数のファイルを見分けるとともに、JASS_FILES や JASS_FILE_OS_VERSION などの環境変数の定義についても識別します。

コピーするファイルは、次の条件で選択され、照合に使用される優先度順にリストされています。たとえば、ホスト固有ファイルと汎用ファイルの両方が存在し、対象システムの名前がホスト固有ファイルで定義されているホスト名と一致する場合には、ホスト固有ファイルが使用されます。次の例では、JASS_HOME_DIR 環境変数で指定された /opt/SUNWjass をホームディレクトリとして使用しますが、ユーザーによっては別のホームディレクトリを指定している場合があります。この例では、検索対象のディレクトリツリーは /opt/SUNWjass/Files/ です。

注 – copy_files 関数では、リストに指定されていても *JASS_HOME_DIR*/Files ディレクトリツリーで検出されないオブジェクトは無視します。

1.ホスト固有のバージョン - /opt/SUNWjass/Files/*file*.*JASS_HOSTNAME*

このオプションでは、ホスト対象プラットフォームの名前と JASS_HOSTNAME 環境変数で指定した値が一致する場合にのみ、ファイルがコピーされます。たとえば、ファイル名が etc/issue で JASS_HOSTNAME が eng1 である場合、この条件の下でコピーされるファイルは次のとおりです。

```
/opt/SUNWjass/Files/etc/issue.eng1
```

2. キーワード + OS 固有のバージョン - /opt/SUNWjass/Files/*file*+*JASS_FILE_COPY_KEYWORD*+*JASS_OS_VERSION*

   このオプションでは、キーワードと OS バージョンの名前が、JASS_FILE_COPY_KEYWORD 環境変数および JASS_OS_VERSION 環境変数で指定した値と一致する場合にのみ、ファイルがコピーされます。

   たとえば、検索されるファイルが /etc/hosts.allow であり、JASS_FILE_COPY_KEYWORD が「secure」(secure.driver の場合) であり、かつ JASS_OS_VERSION が 5.10 である場合、この条件の下でコピーされるファイルは次のとおりです。

   ```
   /opt/SUNWjass/Files/etc/hosts.allow-secure+5.10
   ```

3. キーワード固有のバージョン - /opt/SUNWjass/Files/*file*+*JASS_FILE_COPY_KEYWORD*

   このオプションでは、JASS_FILE_COPY_KEYWORD 環境変数で指定された値にキーワードが一致する場合にのみ、ファイルがコピーされます。たとえば、JASS_FILE_COPY_KEYWORD が「server」である場合、この条件の下でコピーされるファイルは次のとおりです。

   ```
   /opt/SUNWjass/Files/etc/hosts.allow-server
   ```

4. OS 固有のバージョン - /opt/SUNWjass/Files/*file*+*JASS_OS_REVISION*

   このオプションでは、対象プラットフォームの OS リビジョンと、JASS_HOSTNAME 環境変数で指定した値が一致する場合にのみ、ファイルがコピーされます。たとえば、検索されるファイルが /etc/hosts.allow であり、JASS_OS_REVISION が「5.10」である場合、この条件の下でコピーされるファイルは次のとおりです。

   ```
   /opt/SUNWjass/Files/etc/hosts.allow+5.10
   ```

5. 汎用バージョン - /opt/SUNWjass/Files/*file*

   このオプションでは、ファイルが対象システムにコピーされます。

   たとえば、ファイル名が etc/hosts.allow である場合、この条件の下でコピーされるファイルは次のとおりです。

   ```
   /opt/SUNWjass/Files/etc/hosts.allow
   ```

6. コピー元ファイルのサイズが 0 - ファイルの長さまたはサイズがゼロの場合には、そのファイルはシステムにコピーされません。

構成ファイルの使用

環境変数を参照する構成ファイルを編集することによって、Solaris Security Toolkit ソフトウェアを構成することができます。この機能を使用すると、終了スクリプトや監査スクリプトを直接変更することなく、さまざまな環境で Solaris Security Toolkit ソフトウェアドライバを使用できるようになります。

Solaris Security Toolkit 環境変数はすべて、構成ファイルで維持管理されています。これらの構成ファイルはドライバによってインポートされ、ドライバから呼び出されたときに、終了スクリプトと監査スクリプトで変数を使用できるようになります。

Solaris Security Toolkit ソフトウェアには、以下の 3 つの主要構成ファイルがあり、すべて Drivers ディレクトリに格納されています。

- `driver.init`
- `finish.init`
- `user.init.SAMPLE`

driver.init

このファイルには、Solaris Security Toolkit ソフトウェアのフレームワークとすべての操作を定義する環境変数が含まれています。

注 – `driver.init` ファイルは、Solaris Security Toolkit ソフトウェアを新しいバージョンにアップグレードするときに上書きされるため、変更しないでください。

`driver.init` スクリプトには、`JASS_VERSION` や `JASS_ROOT_DIR` などのコアの環境変数が含まれます。

このスクリプトは `user.init` スクリプトを読み込むことにより、ユーザー変数や環境変数の優先指定を組み込みます。また、このスクリプトは `finish.init` ファイルの内容を読み込んで、定義されていない可能性のある終了スクリプト変数を設定します。このスクリプトは、ドライバで使用されるパブリックインタフェースとしての役割を果たし、Solaris Security Toolkit ソフトウェアで使用されるすべての変数を読み込みます。これ以外の初期設定関数は、ドライバ、終了スクリプト、または監査スクリプトのいずれかにより直接アクセスするようには設計されていません。

この `.init` スクリプトに含まれる各環境変数については、第 7 章で説明します。

finish.init

このファイルには、個々の終了スクリプトの動作を定義する環境変数が含まれています。システムのセキュリティー強化方法に影響を与えるのは、以下の 2 つの要素です。

- 選択したドライバに含まれている、実行する終了スクリプトとインストールするファイルのリスト
- finish.init ファイルで定義されている、実行する終了スクリプトの動作

注 – finish.init ファイルは、Solaris Security Toolkit ソフトウェアを新しいバージョンにアップグレードするときに上書きされるため、変更しないでください。

この .init スクリプトに含まれる各環境変数については、第 7 章で説明します。

user.init.SAMPLE

user.init ファイルで変数を定義すると、この変数を driver.init ファイルと finish.init ファイルで定義した変数より優先させることができます。このファイルには、ユーザー定義変数を追加することもできます。管理者はこの機能を使用すると、Solaris Security Toolkit ソフトウェアそれ自体を変更することなく、実際の使用環境のニーズと要件に合わせて、Solaris Security Toolkit ソフトウェアをカスタマイズできます。

user.init.SAMPLE は、Solaris Security Toolkit ソフトウェアが正常に機能するために定義しなければならない項目を示すサンプルファイルです。
user.init.SAMPLE ファイルを user.init にコピーしてから、環境に合うように変更します。user.init ファイルは Solaris Security Toolkit ソフトウェアには含まれていないため、このファイルを作成してカスタマイズしても、このソフトウェアを新しいバージョンにアップグレードする際に上書きされることはありません。

user.init ファイルでは、以下の環境変数のデフォルト値を提供しています。

- JASS_PACKAGE_MOUNT
- JASS_PATCH_MOUNT

この 2 つの変数のデフォルト値は、それぞれ *JumpStart-server-IP address*/jumpstart/Packages と *JumpStart-server-IP address*/jumpstart/Patches です。これらのデフォルト値は、『Solaris Security Toolkit 4.2 管理マニュアル』の第 5 章と、Sun BluePrints™ マニュアル『JumpStart Technology: Effective Use in the Solaris Operating Environment』で推奨されている値です。これらの資料に記載されている推奨値を使用する場合は、user.init.SAMPLE ファイルで変更する必要はありません。このファイルをそのまま user.init にコピーしてください。

ただし、JumpStart 環境を別の使用環境に移行する場合は、使用している JumpStart サーバーとディレクトリパスを参照するように変更する必要があるので、これらの変数を確認してください。上記の各環境変数については、第 7 章で説明します。

user.init ファイルを使用すると、JASS_SVCS_ENABLE、JASS_SVCS_DISABLE などのその他の環境変数を変更することもできます。しかし、変数が特定のドライバですでに使用されている場合もあるので、Solaris Security Toolkit ソフトウェアの動作を変更するときは慎重に行う必要があります。

たとえば、suncluster3x-secure.driver は JASS_SVCS_ENABLE を使用して、/etc/inetd.conf ファイルの特定のサービスを有効にしています。他のサービスを有効にしたい場合は、suncluster3x ドライバファイルをコピーして、そのコピーファイルをカスタマイズし、JASS_SVCS_ENABLE の定義をコメントアウトして、新しい JASS_SVCS_ENABLE の定義を user.init ファイルに追加します。

変数定義の順序に基づいて、user.init ファイルに含まれているすべての定義は、その変数のそのほかのすべての定義を上書きします。それでも、必須ではありませんが、suncluster3x-secure.driver で JASS_SVCS_ENABLE をコメントアウトすることをお勧めします。

注 – pkgrm コマンドを使って SUNWjass を削除すると、user.init および user.run ファイルが作成されている場合、これらのファイルは削除されません。ただし、Files ディレクトリと sysidcfg ファイルは現在の Solaris Security Toolkit ソフトウェアに含まれているので、これらは削除されます。

▼ user.init スクリプトに新しい変数を追加する

次の操作で、user.init スクリプトに環境変数を追加できます。

1. **デフォルト値を使用して変数宣言を追加します。**
2. **新しい変数を user.init ファイルにエクスポートします。**

   この処理ではグローバルなデフォルト値がエクスポートされますが、セキュリティープロファイル (ドライバ) 内でこの値を無効にすれば、後で必要に応じて変更が可能です。

   新しい変数 JASS_ACCT_DISABLE を user.init ファイルに追加して、ユーザーアカウントリストを無効にするコードを、コード例 3-1 に示します。終了スクリプトの実行時に、アカウントが無効になります。

コード例 3-1 ユーザー定義変数の追加

```
JASS_ACCT_DISABLE="user1 user2 user3"; export JASS_ACCT_DISABLE
```

注 – user.run スクリプトには、環境変数を追加したり、そのほかの変更を行ったりしないでください。ユーザーは user.run スクリプトを変更できません。すべての環境変数の上書きは user.init に含まれている必要があります。

▼ user.init ファイルを使用して変数にエントリを追加する

コード例 3-2 に、user.init ファイルを使用して変数にエントリを追加する方法を示します。

コード例 3-2 user.init ファイルを使用して変数にエントリを追加する

```
if [ -f ${JASS_HOME_DIR}/Drivers/finish.init ]; then
  . ${JASS_HOME_DIR}/Drivers/finish.init
fi

JASS_AT_ALLOW="${JASS_AT_ALLOW} newuser1"
export JASS_AT_ALLOW

JASS_CRON_ALLOW="${JASS_CRON_ALLOW} newuser1"
export JASS_CRON_ALLOW

JASS_CRON_DENY="${JASS_CRON_DENY} newuser2"
export JASS_CRON_DENY
```

ファイルテンプレートの使用

Solaris Security Toolkit ソフトウェアは、JASS_FILES 環境変数と copy_files 関数が含まれた Files ディレクトリを使用します。このディレクトリにはファイルテンプレートが格納されており、このテンプレートは、セキュリティー強化実行時に JumpStart クライアントにコピーされます。

以下のファイルテンプレートが Files ディレクトリに格納されています。また以下の項でこれらの各ファイルを説明します。

- 101 ページの「.cshrc」
- 102 ページの「.profile」
- 102 ページの「etc/default/sendmail」
- 103 ページの「etc/dt/config/Xaccess」
- 103 ページの「etc/ftpd/banner.msg」

.cshrc

注 – Solaris 10 OS を実行するシステムでは、このファイルは必須です。`ROOT_HOME_DIR` がスラッシュ (`/`) である場合、`set-root-home-dir.fin` スクリプトとともに使用されます。バージョン 10 以外のバージョンの Solaris オペレーティングシステムを実行しているシステムでは、このファイルは、ソフトウェアが正常に動作しているときは必要ありません。また、必要に応じて変更したり交換したりすることができます。

この構成ファイルは、サンプルとして用意されています。ファイルの完了や履歴など、いくつかの共通 `csh` 変数を設定して、`csh` ユーザーに基本的な構成を提供します。また、現在の作業ディレクトリへのパスを含むコマンド行プロンプトだけでなく、`kill` および `erase` 端末オプションも設定します。

ROOT_HOME_DIR がスラッシュ (/) である場合、set-root-home-dir.fin スクリプトによりインストールされます。それ以外の場合、ROOT_HOME_DIR がデフォルト値 /root であれば、Solaris Security Toolkit は root/.cshrc を使用します。

.profile

注 – Solaris 10 OS を実行するシステムでは、このファイルは必須です。ROOT_HOME_DIR がスラッシュ (/) である場合、set-root-home-dir.fin スクリプトとともに使用されます。バージョン 10 以外のバージョンの Solaris オペレーティングシステムを実行しているシステムでは、このファイルは、ソフトウェアが正常に動作しているときは必要ありません。また、必要に応じて変更したり交換したりすることができます。

この構成ファイルは、サンプルとして用意されています。この構成ファイルは、Solaris Security Toolkit ソフトウェアとともに配布される際に、開始された root sh シェルに対して、UMASK、PATH、および MANPATH の定義のみを行います。

ROOT_HOME_DIR がスラッシュ (/) である場合、set-root-home-dir.fin スクリプトによりインストールされます。それ以外の場合、ROOT_HOME_DIR がデフォルト値 /root であれば、Solaris Security Toolkit は root/.profile を使用します。

etc/default/sendmail

注 – このファイルは、Solaris 8 OS を実行しているシステムでのみ使用します。

Solaris 8 OS では、キュー処理モードのみで sendmail を実行するために、sendmail 構成ファイルを使用することができました。このファイルがコピーされるのは、disable-sendmail.fin スクリプトでセキュリティー強化された Solaris 8 OS システムだけです。

disable-sendmail.fin スクリプトは OS のバージョンに対応しており、セキュリティー強化される OS に応じて sendmail の動作を変更します。詳細については、Sun BluePrints OnLine 掲載記事『Solaris Operating Environment Security: Updated for Solaris 9 OE』を参照してください。

デフォルトでは、このファイルは disable-sendmail.fin によって、セキュリティー強化する Solaris 8 OS にコピーされます。

etc/dt/config/Xaccess

このファイルは、システム上で稼動している X サーバーへの遠隔アクセスを、直接またはブロードキャストにかかわらず、すべて無効にします。X のサポート要件と、Solaris Security Toolkit ソフトウェアの使用環境によっては、このファイルが適していない場合もあります。

デフォルトでは、このファイルは hardening.driver によって、セキュリティー強化するシステムにコピーされます。

etc/ftpd/banner.msg

注 – このファイルは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このファイルは、ファイル転送プロトコル (FTP) サービスの接続バナーを定義します。

デフォルトでは、このファイルは server-secure.driver によって、set-banner-ftpd.fin スクリプトにより強化されているシステムにコピーされます。

etc/hosts.allow と etc/hosts.deny

注 – この 2 つのファイルは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

この 2 つのファイルは、enable-tcpwrappers.fin 終了スクリプトによって Solaris 9 および 10 OS システムにインストールされます。hosts.allow ファイルと hosts.deny ファイルをインストールしたあと、次の操作により、終了スクリプトでトランスミッションコントロールプロトコル (TCP) ラッパーが有効になります。

- Solaris 9 OS を実行しているシステムの /etc/default/inetd 構成ファイルを変更する
- Solaris 10 OS を実行しているシステムで、関連する SMF 操作を呼び出し、inetd、sendmail、および rpc ベースのサービス用の TCP ラッパーの使用を有効にする

hosts.allow ファイルと hosts.deny ファイルは、ローカルポリシー、手順、および要件に基づいて、セキュリティープロファイルをカスタマイズするためのサンプルファイルです。hosts.allow ファイルのセキュリティー保護されたバージョンのドライバでは、許可される Solaris Secure Shell (SSH) アクセスを LOCAL と定義して

います。これは、システムが接続されているサブネットからのみ SSH 接続が許可されることを意味します。hosts.deny ファイルのセキュリティー保護されたバージョンのドライバでは、hosts.allow で許可されていない接続を試みてもすべて拒否されます。

デフォルトでは、このファイルは enable-tcpwrappers.fin によって、セキュリティー強化するシステムにコピーされます。

注 – Solaris Security Toolkit 4.2 ソフトウェアはキーワードをサポートしています。キーワードは、配布パッケージに含まれているさまざまな hosts.allow ファイルを区別するために使用されます。キーワードは、JASS_FILE_COPY_KEYWORD 環境変数内にあり、この注のあとに記載した 3 つのファイル用の「15k_sc」、「server」、および「suncluster」です。

---

etc/hosts.allow-15k_sc

注 – このファイルは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

Sun Fire ハイエンドシステム用のこの hosts.allow ファイルは、tcpwrappers(4) コマンドを使用したアクセスの制御に使用します。このファイルは enable-tcpwrappers.fin スクリプトによりインストールされ、サイトの要件を満たすように構成する必要があります。

etc/hosts.allow-server

注 – このファイルは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

Sun Fire ハイエンドシステム以外の Sun サーバー用のこの hosts.allow ファイルは、tcpwrappers(4) コマンドを使用したアクセスの制御に使用します。このファイルは enable-tcpwrappers.fin スクリプトによりインストールされ、サイトの要件を満たすように構成する必要があります。

etc/hosts.allow-suncluster

注 – このファイルは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

Sun Cluster システム用のこの hosts.allow ファイルは、tcpwrappers(4) コマンドを使用したアクセスの制御に使用します。このファイルは enable-tcpwrappers.fin スクリプトによりインストールされ、サイトの要件を満たすように構成する必要があります。

注意 – suncluster3x-secure.driver を適用したあと、hosts.allow-suncluster ファイルには、クラスタノードの完全指定のドメイン名を追加する必要があります。

etc/init.d/nddconfig

このファイルは、ネットワーク設定の実装に必要な nddconfig 起動スクリプトをコピーし、セキュリティーを向上させます。セキュリティーに関するネットワーク設定については、Sun BluePrints OnLine 掲載記事『Solaris Operating Environment Network Settings for Security: Updated for the Solaris 9 Operating Environment』を参照してください。

デフォルトでは、このファイルは hardening.driver によって、セキュリティー強化するシステムにコピーされます。

etc/init.d/set-tmp-permissions

このファイルは、システムの再起動時に、/tmp ディレクトリと /var/tmp ディレクトリに対する正しいアクセス権を設定します。矛盾が検出された場合は、その内容が標準出力先に出力されて、SYSLOG によって記録されます。S01MOUNTFSYS からの mountall コマンドの実行前と実行後にチェックが実行されるように、このファイルは /etc/rc2.d に 2 回インストールされます。このチェックを行うことにより、マウントポイントとマウントされるファイルシステムの双方で、正しいアクセス権と所有権を持つことができます。

デフォルトでは、このファイルは hardening.driver によって、セキュリティー強化するシステムにコピーされます。

etc/init.d/sms_arpconfig

このファイルは、/etc/rc2.d/S73sms_arpconfig、/etc/sms_domain_arp、および /etc/sms_sc_arp ファイルと組み合わせて Sun Fire ハイエンドシステムで使用します。内部の IP ベース管理ネットワーク上に静的なアドレス解決プロトコル (ARP) を実装して、セキュリティーを向上させます。これらの機能の使用方法の詳細については、Sun BluePrints OnLine 掲載記事『Securing the Sun Fire 12K and 15K System Controllers』と『Securing the Sun Fire 12K and 15K Domains』を参照してください。

デフォルトでは、このファイルは s15k-static-arp.fin によって、セキュリティー強化するシステムにコピーされます。

etc/init.d/swapadd

このファイルは disable-nfs-client.[fin|aud] スクリプトにより使用され、NFS が無効であっても swapadd コマンドを使用してスワップ空間が追加されるようにします。

etc/issue と etc/motd

この 2 つのファイルは、アメリカ合衆国政府の勧告に基づいたもので、ユーザーの活動がモニターされる可能性があるという法律上の通知を表示します。組織独自の法律上のバナーを表示する場合は、そのバナーをこの 2 つのファイルに組み込みます。

この 2 つのファイルはデフォルトのテンプレートとして提供されています。組織に適用する通知については、弁護士に作成や検討を依頼するようにしてください。

デフォルトでは、このファイルは hardening.driver によって、セキュリティー強化するシステムにコピーされます。

etc/notrouter

注 – このファイルは、Solaris OS 9 またはそれ以前のバージョンを実行しているシステムでのみ使用します。

このファイルは、/etc/notrouter ファイルを作成することにより、Solaris OS 9 またはそれ以前のリリースを実行しているシステム上のインタフェース間で IP 転送を無効にするために使用します。ネットワークインタフェースの数にかかわらず、クライアントはルーターとして機能しなくなります。

デフォルトでは、このファイルは hardening.driver によって、セキュリティー強化するシステムにコピーされます。

etc/opt/ipf/ipf.conf

このファイルは一般的な ipfilter 構成ファイルで、ipfilter サービス (svc:/network/ipfilter:default) により使用されます。このサービスは enable-ipfilter.fin スクリプトにより有効にされ、ファイルがインストールされます。このファイルは、サイトの要件を満たすように構成する必要があります。

etc/opt/ipf/ipf.conf-15k_sc

このファイルは Sun Fire ハイエンドシステムシステムコントローラ用の ipfilter 構成ファイルで、ipfilter サービス (svc:/network/ipfilter:default) により使用されます。このサービスは enable-ipfilter.fin スクリプトにより有効にされ、ファイルがインストールされます。このファイルは、サイトの要件を満たすように構成する必要があります。

etc/opt/ipf/ipf.conf-server

このファイルは Sun サーバー用の ipfilter 構成ファイルで、ipfilter サービス (svc:/network/ipfilter:default) により使用されます。このサービスは enable-ipfilter.fin スクリプトにより有効にされ、ファイルがインストールされます。このファイルは、サイトの要件を満たすよう構成する必要があります。

etc/rc2.d/S00set-tmp-permissions と etc/rc2.d/S07set-tmp-permissions

---

注 – この 2 つのファイルは、/etc/init.d/set-tmp-permissions へのシンボリックリンクです。

これらのファイルは、システムの再起動時に、/tmp ディレクトリと /var/tmp ディレクトリに対する正しいアクセス権を設定します。矛盾が検出された場合は、その内容が標準出力先に出力されて、SYSLOG によって記録されます。S01MOUNTFSYS からの mountall コマンドの実行前と実行後にチェックが実行されるように、このス

クリプトは /etc/rc2.d に 2 回インストールされます。このチェックを行うことにより、マウントポイントとマウントされるファイルシステムの双方で、正しいアクセス権と所有権を持つことができます。

デフォルトでは、これらのファイルは hardening.driver によって、セキュリティー強化するシステムにコピーされます。

etc/rc2.d/S70nddconfig

注 – このファイルは、/etc/init.d/nddconfig へのシンボリックリンクです。

このファイルは、ネットワーク設定の実装に必要な S70nddconfig 起動スクリプトをコピーして、セキュリティーを向上させます。Sun BluePrints OnLine 掲載記事『Solaris Operating Environment Network Settings for Security: Updated for Solaris 9 Operating Environment』を参照してください。

デフォルトでは、このファイルは hardening.driver によって、セキュリティー強化するシステムにコピーされます。

etc/rc2.d/S73sms_arpconfig

注 – このファイルは、/etc/init.d/sms_arpconfig へのシンボリックリンクです。

このファイルは、/etc/init.d/sms_arpconfig、/etc/sms_domain_arp、および /etc/sms_sc_arp とともに Sun Fire ハイエンドシステムで使用します。内部の IP ベース管理ネットワーク上に静的なアドレス解決プロトコル (ARP) を実装して、セキュリティーを向上させます。これらの機能の使用方法の詳細については、Sun BluePrints OnLine 掲載記事『Securing the Sun Fire 12K and 15K System Controllers』と『Securing the Sun Fire 12K and 15K Domains』を参照してください。

デフォルトでは、このファイルは s15k-static-arp.fin によって、セキュリティー強化するシステムにコピーされます。

etc/rc2.d/S77swapadd

このファイルは、disable-nfs-client.fin の実行時にインストールされます。通常、disable-nfs-client.fin でスワップ空間が開始されるときに、このタスクを実行するために、Solaris Security Toolkit ソフトウェアによってこの実行コントロールスクリプトが追加されます。

etc/security/audit_control

これはSolaris OS 監査サブシステム用の構成ファイルで、Solaris 基本セキュリティーモジュールとも呼ばれます。このファイルを Solaris 8、9、または 10 OS システムに追加すると、監査サブシステムが構成されます。

このファイルは Solaris Security Toolkit ソフトウェアによって、Solaris 8、9、および 10 OS システムにインストールされます。詳細については、Sun BluePrints OnLine 掲載記事『Auditing in the Solaris 8 Operating Environment』を参照してください。

デフォルトでは、これらのファイルは enable-bsm.fin によって、セキュリティー強化する Solaris 8、9、または 10 OS にコピーされます。

etc/security/audit_class+5.8 と etc/security/audit_event+5.8

これらはSolaris OS 監査サブシステム用の構成ファイルで、Solaris 基本セキュリティーモジュールとも呼ばれます。これらのファイルを Solaris 8 OS システムに追加すると、監査サブシステムが構成されます。

これらのファイルは Solaris Security Toolkit ソフトウェアによって、Solaris 8 OS システムにインストールされます。詳細については、Sun BluePrints OnLine 掲載記事『Auditing in the Solaris 8 Operating Environment』を参照してください。

デフォルトでは、これらのファイルは enable-bsm.fin によって、セキュリティー強化する Solaris 8 OS にコピーされます。

etc/security/audit_class+5.9 と etc/security/audit_event+5.9

これらはSolaris OS 監査サブシステム用の構成ファイルで、Solaris 基本セキュリティーモジュールとも呼ばれます。これらのファイルを Solaris 9 OS システムに追加すると、監査サブシステムが構成されます。

これらのファイルは Solaris Security Toolkit ソフトウェアによって、Solaris 9 OS システムにインストールされます。詳細については、Sun BluePrints OnLine 掲載記事『Auditing in the Solaris 8 Operating Environment』を参照してください。

デフォルトでは、これらのファイルは `enable-bsm.fin` によって、セキュリティー強化する Solaris 9 OS にコピーされます。

etc/sms_domain_arp と /etc/sms_sc_arp

この 2 つのファイルは、`/etc/init.d/sms_arpconfig` ファイルと `/etc/S70sms_arpconfig` ファイルとともに Sun Fire ハイエンドシステムで使用します。内部の IP ベース管理ネットワーク上に静的なアドレス解決プロトコル (ARP) を実装して、セキュリティーを向上させます。これらの機能の使用方法の詳細については、Sun BluePrints OnLine 掲載記事『Securing the Sun Fire 12K and 15K System Controllers』と『Securing the Sun Fire 12K and 15K Domains』を参照してください。

デフォルトでは、これらのファイルは `s15k-static-arp.fin` によって、セキュリティー強化するシステムにコピーされます。

etc/syslog.conf

このファイルは、詳細な記録を行います。独自の中央ログサーバーを追加する組織にとっては、プレースホルダとしての役割を果たすため、事前のログ分析を実行できます。

デフォルトでは、このファイルは `hardening.driver` によって、セキュリティー強化するシステムにコピーされます。

root/.cshrc

注 – Solaris 10 OS を実行するシステムでは、このファイルは必須です。`ROOT_HOME_DIR` がスラッシュ (/) である場合、`set-root-home-dir.fin` スクリプトとともに使用されます。バージョン 10 以外のバージョンの Solaris オペレーティングシステムを実行しているシステムでは、このファイルは、ソフトウェアが正常に動作しているときは必要ありません。また、必要に応じて変更したり交換したりすることができます。

この構成ファイルは、サンプルとして用意されています。ファイルの完了や履歴など、いくつかの共通 csh 変数を設定して、csh ユーザーに基本的な構成を提供します。また、現在の作業ディレクトリへのパスを含むコマンド行プロンプトだけでなく、kill および erase 端末オプションも設定します。

root/.profile

注 – Solaris 10 OS を実行するシステムでは、このファイルは必須です。ROOT_HOME_DIR がスラッシュ (/) である場合、set-root-home-dir.fin スクリプトとともに使用されます。バージョン 10 以外のバージョンの Solaris オペレーティングシステムを実行しているシステムでは、このファイルは、ソフトウェアが正常に動作しているときは必要ありません。また、必要に応じて変更したり交換したりすることができます。

この構成ファイルは、サンプルとして用意されています。この構成ファイルは、Solaris Security Toolkit ソフトウェアとともに配布される際に、開始された root sh シェルに対して、UMASK、PATH、および MANPATH の定義のみを行います。

var/opt/SUNWjass/BART/rules

この規則ファイルは、Solaris 10 OS システムの enable-bart{.fin|aud} スクリプトで、Basic Auditing and Reporting Tool (BART) により使用されます。規則ファイルの詳細は、156 ページの「enable-bart.fin」を参照してください。

var/opt/SUNWjass/BART/rules-secure

この規則ファイルは、Solaris 10 OS システムの enable-bart{.fin|aud} スクリプトで、Basic Auditing and Reporting Tool (BART) 用の secure.driver により使用されます。規則ファイルの詳細は、156 ページの「enable-bart.fin」を参照してください。

第4章

ドライバ

この章では、ドライバの使用、追加、変更、および削除について説明します。この章で取り上げるドライバは、Solaris OS システムを強化、最小化、および監査するために、Solaris Security Toolkit ソフトウェアで使用されるドライバです。一連のドライバと関連ファイルによって、セキュリティープロファイルが構成されます。

`secure.driver` は、Solaris Security Toolkit ソフトウェアを使用してセキュリティー保護されたシステム構成を開発するための出発点として最もよく使われるドライバです。`secure.driver` は、Solaris Secure Shell (SSH) ソフトウェアを除き、(OS の動作に必要のない) ネットワークサービスを含むすべてのサービスを無効にします。使用環境によっては、この処理が適していない場合もあります。使用しているシステムに必要なセキュリティー変更を確認してから、この章や関連する章に記載されている情報を使用して変更を行なってください。

この章では、以下の項目を説明します。

ドライバの関数と処理について

セキュリティー強化と監査を行う際のコアの処理は、`driver.run` スクリプト内の関数で定義されます。セキュリティー強化と監査を行う場合、使用するドライバは、セキュリティープロファイルが構成された後で、`driver.run` スクリプトを呼び出します。つまり、ドライバにより `driver.init` ファイルが呼び出されて、`JASS_FILES` 環境変数と `JASS_SCRIPTS` 環境変数が定義された後で、`driver.run` スクリプト関数が呼び出されます。強化と監査の両方の操作で `JASS_FILES` 環境変数と `JASS_SCRIPTS` 環境変数に含まれている各エントリを処理するのが、このスクリプトです。

注意 – secure.driver ドライバを使用してセキュリティー保護されたシステムは、disable-rpc.fin スクリプトが含まれているため、JumpStart や NIS を使用できません。その代わりに、disable-rpc.fin スクリプトを含まない新しいドライバを作成する必要があります。JumpStart や NIS を使用していたマシンで disable-rpc.fin スクリプトを使用していて、ログインできない場合は、システムをシングルユーザーモードで再起動し (boot -s)、SMF を使用して bind を有効にするか (svcadm enable bind)、(/etc/nsswitch.conf および /var/svc/profile/ns_* SMF ファイルを使用して) NIS を使用しないようにネームサービスを変更します。

このスクリプトの高度な処理フローを、次に示します。

1. 機能 (.funcs) ファイルを読み込む

   これらの機能ファイルはすべて *JASS_HOME_DIR*/Drivers ディレクトリに格納されています。

2. 基本チェックを行う
3. ユーザー機能の優先指定を読み込む
4. ファイルシステムを JumpStart クライアントにマウントする (JumpStart モードのみ)
5. JASS_FILES 環境変数で指定されているファイルをコピーまたは監査する (オプション)
6. JASS_SCRIPTS 環境変数で指定されているスクリプトを実行する (オプション)
7. 実行に対する合計スコアを計算する (監査操作のみ)
8. ファイルシステムを JumpStart クライアントからアンマウントする (JumpStart モードのみ)

上記の各機能については、以降の節で詳細に説明します。

機能ファイルを読み込む

driver.run スクリプトで行う最初のタスクは、機能ファイルの読み込みです。この段階で機能ファイルを読み込むことで、各ファイルの機能を driver.run スクリプトが利用できるようになります。実行されるすべてのスクリプトが共通関数を利用できるようになります。このタスク実行時に読み込まれる機能ファイルは、以下のとおりです。

- audit_private.funcs
- audit_public.funcs
- clean_private.funcs

- driver_private.funcs
- driver_public.funcs
- common_misc.funcs
- common_log.funcs

基本チェックを行う

Solaris Security Toolkit ソフトウェアでは、コアの環境変数が設定されているかどうかのチェックを行います。このチェックで、ソフトウェアが正常に実行されることを確認します。いずれかのチェックに失敗すると、エラーが報告され、ソフトウェアが終了します。チェックでは、以下の確認が行われます。

- JASS_OS_REVISION 環境変数が定義されていること。この環境変数が定義されていない場合、driver.init スクリプトが呼び出されていないか、または環境変数が不適切に変更されている可能性があります。
- JumpStart モードの場合、JASS_PACKAGE_MOUNT 環境変数が定義されていること。この環境変数が正しく定義されていない場合は、JumpStart のインストール時に Packages ディレクトリを検索できないことがあります。
- JumpStart モードの場合、JASS_PATCH_MOUNT 環境変数が定義されていること。この環境変数が正しく定義されていない場合は、JumpStart のインストール時に Patches ディレクトリを検索できないことがあります。

ユーザー機能の優先指定を読み込む

現在のプロファイルを続いて処理する前に、user.run ファイルが存在する場合は、このファイルを読み込みます。このファイルには、Solaris Security Toolkit ソフトウェアのデフォルトの関数に優先する関数を含む、サイト固有または組織固有の関数がすべて格納されています。このファイルはデフォルトでは存在していないため、この機能が必要な場合は、ユーザーが手動で作成する必要があります。

この機能を利用して、新しい関数を実装したり、使用環境に合うように既存の関数をカスタマイズすると、Solaris Security Toolkit ソフトウェアの機能を拡張または強化することができます。この user.run ファイルは、関数用のファイルであることを除けば、環境変数用のファイルである user.init ファイルとほぼ同じです。

ファイルシステムを JumpStart クライアントにマウントする

注 – ローカルのブート可能 CD-ROM を使用して JumpStart をインストールする場合は、ローカルメディアからディレクトリにアクセスするようにこの機能を変更してください。ネットワークファイルシステム (NFS) を使用して、リモートサーバーから `Patches` ディレクトリと `Packages` ディレクトリにアクセスする場合は、変更する必要はありません。

JumpStart モードでは、`driver.run` スクリプトは `mount_filesystems` という内部サブルーチンを呼び出します。このルーチンは、JumpStart クライアント上に次のディレクトリをマウントします。

- `JASS_PACKAGE_MOUNT` (`JASS_PACKAGE_DIR` 上にマウントされる)
- `JASS_PATCH_MOUNT` (`JASS_PATCH_DIR` 上にマウントされる)

その他のファイルシステムのマウントポイントが必要な場合は、`user.run` スクリプトを使用して必要なマウントポイントを実装します。このルーチンは JumpStart モード固有のものなので、スタンドアロンモードでは実行されません。

ファイルをコピーまたは監査する

共通関数の読み込み、環境変数の初期化、ファイルシステムのマウント (必要な場合) を行なって Solaris Security Toolkit ソフトウェアの基盤が確立したら、すぐに実行することができます。セキュリティー強化と監査のどちらを実行するかにかかわらず、Solaris Security Toolkit ソフトウェアは、対象システムでコピーや検証が行われるファイルテンプレートの詳細リストを作成します。このリストは、`JASS_FILES` グローバル環境変数にあるエントリを、`JASS_FILES_`*x_xx* OS バージョン環境変数 (たとえば、Solaris 10 OS では `JASS_FILES_5_10`) にあるエントリと連結して作成します。グローバル環境変数と OS 環境変数はいずれもオプションであり、一方だけを定義したり、両方とも定義しなくてもかまいません。連結されたリストは、`JASS_FILES` 環境変数に格納されます。この変数についての詳細は、第 7 章の 238 ページの「JASS_FILES」を参照してください。

作成されたリストに少なくとも 1 つのエントリが含まれている場合、`JASS_SCRIPTS` リストの先頭に `install-templates.fin` という特殊な終了スクリプトが追加されます。強化処理では、このスクリプトが、作成されたリストの内容を受け取って対象システムにコピーしたあとで、その他の終了スクリプトが実行されます。監査処理では、`install-templates.aud` スクリプトが、ファイルが対象システム上のファイルと一致することを確認します。

スクリプトを実行する

Solaris Security Toolkit ソフトウェアは、JASS_SCRIPTS 環境変数で定義されたスクリプトを実行します。セキュリティー強化と監査のどちらを実行するかにかかわらず、Solaris Security Toolkit ソフトウェアは、対象システムでコピーや検証が行われるファイルテンプレートの詳細リストを作成します。このリストは、JASS_SCRIPTS グローバル環境変数にあるエントリを、JASS_SCRIPTS_*x_xx* OS バージョン環境変数 (たとえば、Solaris 10 OS では JASS_SCRIPTS_5_10) にあるエントリと連結して作成します。グローバル環境変数と OS 環境変数はいずれもオプションであり、一方だけを定義したり、両方とも定義しなくてもかまいません。連結されたリストは、JASS_SCRIPTS 環境変数に格納されます。この変数についての詳細は、第 7 章の 242 ページの「JASS_FINISH_DIR」を参照してください。

強化処理では、すべての終了スクリプトが順に実行されます。終了スクリプトは、JASS_FINISH_DIR ディレクトリに格納されています。

監査処理では、最初にいくつかの追加処理を実行する必要があります。JASS_SCRIPTS で定義されているスクリプトで監査を実行する前に、名前を終了スクリプト名から監査スクリプト名に変更してください。ファイル名の拡張子が自動的に .fin から .aud に変更されます。また Solaris Security Toolkit ソフトウェアでは、監査スクリプトが JASS_AUDIT_DIR に格納されていることを前提としています。ファイル名とディレクトリの変更が行われた後で、すべての監査スクリプトが順に実行されます。

スクリプトの出力は、以下のいずれかの方法で処理されます。

- jass-execute -o オプションで指定されているファイルに記録される。ファイルを指定しない場合は、標準出力に出力されます。このオプションは、スタンドアロンモードでのみ使用できます。
- JumpStart インストール時に JumpStart クライアント上の /var/sadm/system/logs/finish.log ファイルに記録される。/var/sadm/system/logs/finish.log ファイルは、クライアント上で実行されるすべての JumpStart コマンドが使用する、標準のログファイルです。このオプションは、JumpStart モードでのみ使用できます。
- JASS_REPOSITORY/*timestamp*/jass-install-log.txt ファイルまたは jass-audit-log.txt ファイルに記録される。*timestamp* は、YYYYMMDDHHMMSS 形式の完全指定時刻パラメータです。この値は、Solaris Security Toolkit ソフトウェアの各実行に対する定数であり、実行の開始時刻を表します。たとえば、2005 年 7 月 1 日午前 1 時 30 分に開始された処理は、20050701013000 という値で表されます。ログファイルは、実行されるたびに生成されます。強化処理では、jass-install-log.txt ファイルが作成されます。監査処理では、jass-audit-log.txt ファイルが作成されます。ファイルの内容は変更しないでください。

実行に対する合計スコアを計算する

強化処理では、ドライバに関する操作がすべて完了すると、ドライバの合計スコアが計算されます。このスコアはドライバの状態を示し、複数のドライバが呼び出されているときは総合計の中に含まれます。使用しているドライバが 1 つだけの場合は、この合計と総合計は同じ値です。すべてのチェックに合格した場合、スコアはゼロです。チェックに失敗した場合、スコアは失敗したチェックの個数を示す数値になります。

ファイルシステムを JumpStart クライアントからアンマウントする

JumpStart モードで操作している場合、ドライバ関連の操作がすべて終了すると、116 ページの「ファイルシステムを JumpStart クライアントにマウントする」の処理時にマウントされたファイルシステムがアンマウントされます。通常、この機能は JumpStart クライアントのインストールの終了を示します。この時点で、呼び出し元ドライバに制御が戻ります。ドライバは、終了して処理を終わりにするか、または別のドライバを呼び出して新しい処理を開始することができます。

ドライバのカスタマイズ

組織のポリシー、基準、アプリケーション要件は、たとえわずかでも各組織で異なるため、Solaris Security Toolkit ドライバの変更は最も頻繁に行われるタスクの 1 つです。このため、Solaris Security Toolkit ソフトウェアでは、ドライバが実行するタスクをカスタマイズする機能をサポートしています。

システムやアプリケーションで、選択したドライバにより無効にされる一部のサービスとデーモンを必要とする場合や、アクティブでないスクリプトを有効にする場合は、その処理を行なってから Solaris Security Toolkit ソフトウェアを実行してください。

同様に、有効にしておかなければならないサービスが存在し、選択したドライバによって無効にされる場合は、選択したドライバを Solaris Security Toolkit ソフトウェアで実行する前に、この選択したドライバの構成を無効にします。ソフトウェアの構成を確認して、必要なカスタマイズをすべて行なってから、システムの構成を変更するようにしてください。この方法の方が、元に戻さなければならない変更を検出して、別の構成で再適用するよりは効率的です。

Solaris Security Toolkit ソフトウェアを使用してサービスを無効にする主な方法には、次の 2 つの方法があります。1 つめの方法は、ドライバを変更して、JASS_SCRIPTS パラメータで定義されている実行してはならない終了スクリプトをコメントアウトまたは削除する方法です。この方法は、最も一般的に使用されるドライバのカスタマイズ方法の 1 つです。

たとえば、使用環境で NFS ベースのサービスが必要な場合は、次のようにすると、これらのサービスを有効にしておくことができます。hardening.driver のローカルコピーにある disable-nfs-server.fin スクリプトと disable-rpc.fin スクリプトの先頭に # 記号を付加して、これらのスクリプトをコメントアウトします。あるいは、これらのスクリプトをファイルから完全に削除することもできます。一般的な規則として、コメントアウトまたは削除したエントリは、以下のような情報とともにファイルヘッダーに記載しておくことをお勧めします。

- 無効にしたスクリプト名
- そのスクリプトを無効にした担当者名
- 変更を行った日時を示す時刻表示
- この変更が必要であった理由の簡単な説明

上記の情報を記載しておくと、特にソフトウェアの新バージョン用にドライバを更新しなければならない場合など、長期にわたるドライバの維持管理に役立ちます。

注 – Solaris Security Toolkit ソフトウェアとともに配布されたドライバを直接変更しないでください。Solaris Security Toolkit ソフトウェアを削除またはアップグレードしたときに、ドライバの変更が影響を受けないように、Solaris Security Toolkit 配布パッケージに含まれているドライバのコピーに対して必ず変更を行なってください。

サービスを無効にするもう 1 つの方法は、環境変数をカスタマイズする方法です。通常、このカスタマイズは、ドライバまたは user.init ファイルのいずれかで行います。user.init ファイルでの変更は、その変更がグローバルなもので、すべてのドライバで使用される場合にのみ行なってください。そうでない場合は、変更を必要とするドライバに限定して変更を行なってください。

たとえば、inetd デーモンによって開始されるサービスを有効または無効にするには、JASS_SVCS_ENABLE 環境変数と JASS_SVCS_DISABLE 環境変数を使用します。変数の使用についての詳細は、第 7 章を参照してください。また、第 7 章の 227 ページの「変数のカスタマイズと割り当て」も参照してください。

▼ ドライバをカスタマイズするには

元のファイルが更新されたときに、カスタマイズしたファイルが更新された新しいファイルで上書きされないようにドライバをカスタマイズするには、次の手順を実行します。なお、ソフトウェアのアップグレードや削除を行うときに、カスタマイズしたファイルが誤って削除されないようにするときにも、この手順を使用します。

1. **カスタマイズするドライバとその関連ファイルをコピーします。**

   たとえば、組織固有の `secure.driver` を作成する場合、`Drivers` ディレクトリに格納されている次のドライバをコピーします。

   - `secure.driver`
   - `config.driver`
   - `hardening.driver`

   `config.driver` と `hardening.driver` は `secure.driver` から呼び出されるドライバであるため、この 2 つのドライバもコピーする必要があります。カスタマイズしているドライバがほかのドライバを呼び出したり使用したりしない場合は、カスタマイズしているドライバのみコピーします。

2. **コピーしたファイルを、カスタムドライバとして識別される名前に変更します。**

   たとえば、自社名を使用する場合は、次のようなファイル名になります。

   - `abccorp-secure.driver`
   - `abccorp-config.driver`
   - `abccorp-hardening.driver`

   詳細については、『Solaris Security Toolkit 4.2 管理マニュアル』の第 1 章「Solaris Security Toolkit ソフトウェアの構成およびカスタマイズ」を参照してください。

3. **カスタムの *prefix*`-secure.driver` を、新しい関連ファイル *prefix*`-config.driver` と *prefix*`-hardening.driver` を呼び出すように変更します。**

   この手順は、新しい *prefix*`-secure.driver` から元の `config.driver` と `hardening.driver` が呼び出されないようにするときに必要となります。カスタマイズしているドライバが他のドライバを呼び出したり使用したりしないときは、この手順は必要ありません。

4. **ドライバからファイルをコピー、追加、または削除するには、`JASS_FILES` 環境変数を変更します。**

   この変数についての詳細は、第 7 章を参照してください。

   次のコーディング例は、`Drivers/config.driver` ファイルからの抜粋です。このセキュリティープロファイルは、プラットフォーム上で基本の構成を行います。このセキュリティープロファイルには、ファイルテンプレートと終了スクリプトの明確な使用方法例が示されています。

   次の例では、`driver.run` 関数が呼び出されたときに、`JASS_HOME_DIR/Files/` ディレクトリから `/.cshrc` ファイルと `/.profile` ファイルを対象プラットフォームにコピーするようドライバを構成しています。

```
JASS_FILES="
/.cshrc
/.profile
"
```

a. **いずれかのファイルの内容を変更するには、**`JASS_HOME_DIR/Files/`** ディレクトリに格納されているファイルを変更します。**

b. **ファイルテンプレートを追加または削除する必要があるだけならば、そのように** `JASS_FILES` **変数を変更します。**

c. **Solaris OS バージョンを定義する場合は、**`JASS_FILES`** 変数の末尾にオペレーティングシステムのメジャーおよびマイナーバージョン番号を下線 (_) で区切って付加します。**

注 – 手順 c では、Solaris OS のバージョンに加えて、ほかの条件を定義および追加することもできます。使用できる各種の条件については、60 ページの「copy_files」の説明を参照してください。

Solaris Security Toolkit ソフトウェアでは、オペレーティングシステムのバージョン固有のファイルリストをサポートしています。定義されている Solaris OS バージョンで Solaris Security Toolkit ソフトウェアが実行されている場合にのみ、これらのファイルリストが一般ファイルリストの内容に追加されます。たとえば、Solaris 10 OS は次のように指定します。

```
JASS_FILES_5_10
```

5. **ドライバからスクリプトを追加または削除するには、**`JASS_SCRIPTS`** 変数を変更します。**

   この変数についての詳細は、第 7 章を参照してください。

6. **他のドライバを呼び出すには、ネストまたは階層セキュリティープロファイルを作成します。**

   大部分のプラットフォームに共通する基準を設定し、かつプラットフォームまたはアプリケーション間の相違を維持する場合に、この手法は便利です。

   コード例 4-1 は `secure.driver` ファイルからの抜粋です。このファイルは、構成ドライバおよび強化ドライバの両方を呼び出すラッパーとして使用されます。この場合は、セキュリティープロファイルの実際の機能を実装しています。これは頻繁に使用

されるモデルですが、必ずしもこのとおりである必要はありません。たとえ実際に (コード例 4-1 の場合のように) 使用されなくても、各ドライバは JASS_FILES と JASS_SCRIPTS の規則をサポートします。

コード例 4-1　　ネストまたは階層セキュリティープロファイルの作成

```
DIR="`/bin/dirname $0`"
export DIR

. ${DIR}/driver.init
. ${DIR}/config.driver
. ${DIR}/hardening.driver
```

もう少し複雑な構成を、コード例 4-2 に示します。この例では、ドライバは他の基本的なドライバを呼び出すだけでなく、独自の機能も実装しています。この例では、この新しいセキュリティープロファイルで /etc/named.conf ファイルをインストールし、config.driver ドライバと hardening.driver ドライバを実行したあとで、configure-dns.fin スクリプトを実行します。

コード例 4-2　　ドライバによる独自機能の実装

```
DIR="`/bin/dirname $0`"
export DIR

. ${DIR}/driver.init
. ${DIR}/config.driver
. ${DIR}//hardening.driver

JASS_FILES="
/etc/named.conf
"

JASS_SCRIPTS="
configure-dns.fin
"

. ${DIR}/driver.run
```

注 – コード例 4-2 では、さまざまなレベルの機能と適用範囲を提供するように、ドライバをネストさせる方法の一例を示しています。/etc/named.conf と configure-dns.fin は、コーディング例としてのみ表示されています。デフォルトでは、これらのファイルは Solaris Security Toolkit ソフトウェアで提供されていません。

---

7. **ドライバのカスタマイズが終了したら、そのドライバを Drivers ディレクトリに保存します。**

8. ドライバが正しく機能するかどうかテストします。

標準のドライバの使用

この節では、デフォルトで Drivers ディレクトリに用意されている、以下のドライバについて説明します。

- 123 ページの「config.driver」
- 124 ページの「hardening.driver」
- 127 ページの「secure.driver」

Solaris Security Toolkit ソフトウェアには、これら標準のドライバのほかにもドライバが含まれています。製品固有のドライバのリストについては、128 ページの「製品固有のドライバの使用」を参照してください。

config.driver

このドライバは、secure.driver から呼び出され、そのドライバに関連するタスクを実行します。関連する関数を 1 つのドライバにまとめて共通関数を作成し、その共通関数を構成単位として使用すると、より複雑な構成を構築することができます。次の例では、類似したタスクを 1 つの独自のドライバに分離することで、異なるセキュリティー要件を持つ複数のマシンで 1 つの基本 Solaris OS 構成ドライバを共有できるようにしています。

config.driver からの抜粋をコード例 4-3 に示します。

コード例 4-3　　config.driver からの抜粋

```
DIR="`/bin/dirname $0`"
export DIR

. ${DIR}/driver.init

JASS_FILES="
/.cshrc
"

JASS_SCRIPTS="
set-root-password.fin
set-term-type.fin
"

. ${DIR}/driver.run
```

config.driver は次のいくつかの作業を実行します。

1. driver.init ファイルを呼び出して、Solaris Security Toolkit フレームワークを初期化し、その実行環境を構成します。
2. JASS_FILES 環境変数と JASS_SCRIPTS 環境変数を設定します。

   これらの変数では、このドライバが実行する実際の設定変更を定義します。
3. driver.run スクリプトを呼び出します。driver.run スクリプトはファイルのインストールを完了させて、構成専用のスクリプトをすべて実行します。

コード例 4-3 では、JASS_HOME_DIR/Files ディレクトリに格納されている .cshrc ファイルが、/.cshrc にコピーされて、終了スクリプト (set-root-password.fin と set-term-type.fin) が対象システム上で実行されます。

hardening.driver

Solaris Security Toolkit ソフトウェアに含まれているセキュリティー専用スクリプトのほとんどは、hardening.driver にリストされています。このドライバは、hardening.driver に含まれていないセキュリティー拡張機能を追加で実装することでスクリプトの変更を行います。config.driver と同様、このドライバでも driver.run スクリプトで実行されるスクリプトを定義します。

このドライバには、以下のスクリプトがリストされています。

- disable-ab2.fin

- disable-apache.fin
- disable-apache2.fin
- disable-appserv.fin
- disable-asppp.fin
- disable-autoinst.fin
- disable-automount.fin
- disable-dhcpd.fin
- disable-directory.fin
- disable-dmi.fin
- disable-dtlogin.fin
- disable-face-log.fin
- disable-IIim.fin
- disable-ipv6.fin
- disable-kdc.fin
- disable-keyserv-uid-nobody.fin
- disable-ldap-client.fin
- disable-lp.fin
- disable-mipagent.fin
- disable-named.fin
- disable-nfs-client.fin
- disable-nfs-server.fin
- disable-nscd-caching.fin
- disable-ppp.fin
- disable-preserve.fin
- disable-power-mgmt.fin
- disable-remote-root-login.fin
- disable-rhosts.fin
- disable-routing.fin
- disable-rpc.fin
- disable-samba.fin
- disable-sendmail.fin
- disable-ssh-root-login.fin
- disable-slp.fin
- disable-sma.fin
- disable-snmp.fin
- disable-spc.fin
- disable-syslogd-listen.fin
- disable-system-accounts.fin
- disable-uucp.fin
- disable-vold.fin
- disable-xserver-listen.fin
- disable-wbem.fin
- disable-xfs.fin
- enable-bart.fin
- enable-account-lockout.fin
- enable-coreadm.fin
- enable-ftpaccess.fin

- enable-ftp-syslog.fin
- enable-inetd-syslog.fin
- enable-ipfilter.fin
- enable-password-history.fin
- enable-priv-nfs-ports.fin
- enable-process-accounting.fin
- enable-rfc1948.fin
- enable-stack-protection.fin
- enable-tcpwrappers.fin
- install-at-allow.fin
- install-ftpusers.fin
- install-loginlog.fin
- install-md5.fin
- install-nddconfig.fin
- install-newaliases.fin
- install-sadmind-options.fin
- install-security-mode.fin
- install-shells.fin
- install-sulog.fin
- remove-unneeded-accounts.fin
- set-banner-dtlogin.fin
- set-banner-ftpd.fin
- set-banner-sendmail.fin
- set-banner-sshd.fin
- set-banner-telnetd.fin
- set-flexible-crypt.fin
- set-ftpd-umask.fin
- set-login-retries.fin
- set-power-restrictions.fin
- set-root-group.fin
- set-root-home-dir.fin
- set-rmmount-nosuid.fin
- set-strict-password-checks.fin
- set-sys-suspend-restrictions.fin
- set-system-umask.fin
- set-tmpfs-limit.fin
- set-user-password-reqs.fin
- set-user-umask.fin
- update-at-deny.fin
- update-cron-allow.fin
- update-cron-deny.fin
- update-cron-log-size.fin
- update-inetd-conf.fin
- install-md5.fin
- install-fix-modes.fin

注 – install-strong-permissions.fin スクリプトによる変更を除き、提供されている終了スクリプトによる変更はすべて元に戻すことができます。このスクリプトで行った変更が必要なくなった場合には、手動で元に戻す必要があります。install-strong-permissions.fin スクリプトは、Solaris 10 OS 上では動作しません。

また、以下のスクリプトは hardening.driver にリストされていますが、コメントアウトされています。

- disable-keyboard-abort.fin
- disable-picld.fin
- print-rhosts.fin
- enable-bsm.fin
- install-strong-permissions.fin

上記スクリプトについては、第 5 章を参照してください。

secure.driver

secure.driver は、クライアントのインストールに使用される rules.SAMPLE ファイルにあるサンプル規則に最もよく含まれているドライバです。このドライバは、Solaris Security Toolkit ソフトウェアで**すべての**強化機能を実装するときにすぐに使用できるドライバです。要求されたタスクの初期化を行い、config.driver ドライバを呼び出してシステムを構成し、hardening.driver を呼び出してセキュリティー強化タスクをすべて実行します。

secure.driver の内容を、コード例 4-4 に示します。

コード例 4-4 secure.driver の内容

```
DIR="`/bin/dirname $0`"
export DIR

. ${DIR}/driver.init

. ${DIR}/config.driver

. ${DIR}/hardening.driver
```

製品固有のドライバの使用

この節では、特定のサン製品やその構成のセキュリティー強化に使用される、製品固有のドライバについて説明します。これらのドライバは、Solaris Security Toolkit とともに Drivers ディレクトリに格納されています。製品固有のドライバを、表 4-1 に一覧表示します。

サンの新製品や更新されたサン製品のセキュリティーを強化するために、定期的に新しいドライバがリリースされます。Solaris Security Toolkit ソフトウェアの今後のバージョンでは、新しいドライバと変更されたドライバが提供される予定です。

表 4-1 製品固有のドライバ

製品	ドライバ名

サーバーシステム[1]	`server-secure.driver`
	`server-config.driver`
	`server-hardening.driver`

Sun Cluster 3.x ソフトウェア	`suncluster3x-secure.driver`
	`suncluster3x-config.driver`
	`suncluster3x-hardening.driver`

Sun Fire ハイエンドシステムシステムコントローラ	`sunfire_15k_sc-secure.driver`
	`sunfire_15k_sc-config.driver`
	`sunfire_15k_sc-hardening.driver`

1 Solaris Security Toolkit バージョン 4.2 ソフトウェアより前のバージョンでは、これらのドライバ名は `server` ではなく `desktop` になっていました。

注 – `server-secure.driver`、`suncluster3x-secure.driver`、および `sunfire_15k_sc-secure.driver` のすべての説明において、`*-secure.driver` は `jass-execute -d` コマンドともに使用されますが、このコマンドは上記の 3 つすべてのドライバを使用し、正しい結果を生成することに注意してください。

---

server-secure.driver

注 – Solaris Security Toolkit 4.2 ソフトウェア以前には、このドライバは desktop-secure.driver と呼ばれていました。Solaris Security Toolkit 4.2 ソフトウェアが動作し Solaris 10 OS を使用するシステムに関しては、このドライバは現在、旧バージョンの Solaris Security Toolkit の sunfire_15k_domain-secure.driver および jumpstart-secure.driver にある機能を組み込んでいます。

このドライバは、secure.driver に基づき、Sun Fire ハイエンドシステムシステムコントローラ以外のシステムのセキュリティー確保に必要となる変更を示すスクリプト例として提供されています。このスクリプトは参考例なので、使用環境に応じてカスタマイズしてください。このドライバが secure.driver と異なる点は次のとおりです。

- 次の inetd サービスは無効ではありません。
  - telnet (Telnet)
  - ftp (ファイル転送プロトコル)
  - dtspc (CDE サブプロセスコントロールサービス)
  - rstatd (カーネル統計情報サーバー)
  - rpc.smserverd (リムーバブルメディアデバイス・サーバー)
- 次のファイルテンプレートは使用されません。
  - /etc/dt/config/Xaccess
  - /etc/syslog.conf
- 次の終了スクリプトは server-secure.driver ではコメントアウトされています。
  - disable-autoinst.fin
  - disable-automount.fin
  - disable-keyboard-abort.fin
  - disable-dtlogin.fin
  - disable-lp.fin
  - disable-nfs-client.fin
  - disable-rpc.fin
  - disable-vold.fin
  - disable-xserver-listen.fin
  - print-rhosts.fin

suncluster3x-secure.driver

このドライバは、Sun™ Cluster 3.*x* ソフトウェアリリースを強化するベースライン構成を提供します。このドライバを変更すると、無効にされている Solaris OS 機能を削除できます。ただし、Sun Cluster ソフトウェアが正常に動作するために必要で、有効になっているサービスは変更しないでください。詳細については、Sun BluePrints OnLine 掲載記事『Securing the Sun Cluster 3.x Software』を参照してください。

sunfire_15k_sc-secure.driver

このドライバは、Sun Fire ハイエンドシステムのシステムコントローラ (SC) をセキュリティー保護するために唯一サポートされるメカニズムです。SC で必要でないサービスはすべて、このドライバによって無効にされます。無効にされているサービスが必要な場合は、そのサービスを無効にしないようにこのドライバを変更します。詳細については、Sun BluePrints OnLine 掲載記事『Securing the Sun Fire 12K and 15K System Controllers』を参照してください。

注意 – `suncluster3x-secure.driver` を適用したあと、`hosts.allow-suncluster` ファイルには、クラスタノードの完全指定のドメイン名を追加する必要があります。

第5章

終了スクリプト

この章では、終了スクリプトの使用、追加、変更、および削除について説明します。この章で取り上げるスクリプトは、Solaris OS システムを強化および最小化するために Solaris Security Toolkit ソフトウェアで使用されるスクリプトです。

Solaris Security Toolkit ソフトウェアのデフォルトのスクリプトは、OS の動作を必要とせずに、ネットワークサービスを含むすべてのサービスを無効にします。使用環境によっては、この処理が適していない場合もあります。使用しているシステムに必要なセキュリティー変更を確認してから、この章に記載されている情報を使用して変更を行なってください。

この章では、以下の項目を説明します。

終了スクリプトのカスタマイズ

終了スクリプトは、Solaris Security Toolkit ソフトウェアの中核としての機能を果たします。このスクリプトは、大部分のセキュリティー変更を一括して実行します。また、セキュリティープロファイル (ドライバ) の設計に応じて、それに伴う変更を、さまざまな方法で組み合わせたりグループ化することができる単独ファイルに取り出します。

この節では、既存の終了スクリプトのカスタマイズと、新しい終了スクリプトの作成についての手順と推奨事項を説明します。また、終了スクリプト機能を使用する際のガイドラインについても説明します。

注 – 変更をより多くのユーザーのために役立てたいときは、拡張機能に関するバグレポートや要望を提出することをご検討ください。Solaris Security Toolkit 開発チームは、ユーザーに役立つソフトウェアの改善方法を常に求めております。

既存の終了スクリプトをカスタマイズする

終了スクリプトは、Solaris Security Toolkit ドライバと同じように、カスタマイズすることができます。Solaris Security Toolkit ソフトウェアで提供されているスクリプトは、変更しないでください。直接元のスクリプトを変更せずに、必ず終了スクリプトのコピーを作成して、それを変更するようにしてください。元のスクリプトを変更すると、Solaris Security Toolkit ソフトウェアをアップグレードまたは削除するときに、変更が失われる可能性があります。できる限りスクリプトの変更は必要最低限にとどめ、その変更について記録に残しておいてください。

終了スクリプトは、環境変数を使用してカスタマイズします。Solaris Security Toolkit に含まれる大部分の終了スクリプトの動作はこの方法でカスタマイズできるため、実際のスクリプトを変更する必要はありません。この方法が使用できない場合には、コードを変更する必要があります。

すべての環境変数のリストと、環境変数を定義する際のガイドラインについては、第 7 章を参照してください。

注 – JumpStart サーバーに Solaris Security Toolkit ソフトウェアをインストールすると、終了スクリプトは、JumpStart クライアント上で実行されているメモリー常駐 miniroot から実行されます。miniroot には、Solaris OS のほぼすべての機能が格納されています。終了スクリプトを作成する場合、クライアントディスクは `/a` にマウントされているため、`chroot` コマンドを使用してコマンドを実行する必要があることもあります。Solaris Security Toolkit ソフトウェアのスタンドアロンモードでの実行中には、この制約はありません。

▼ 終了スクリプトをカスタマイズするには

元のファイルが更新されたときに、カスタマイズしたファイルが更新された新しいファイルで上書きされないように終了スクリプトをカスタマイズするには、次の手順を実行します。なお、`pkgrm` コマンドを使用してソフトウェアを削除しても、カスタマイズしたファイルは削除されません。

1. **カスタマイズするスクリプトとその関連ファイルをコピーします。**

2. コピーしたファイルを、カスタムスクリプトおよびカスタムファイルとして識別される名前に変更します。

   命名規則については、『Solaris Security Toolkit 4.2 管理マニュアル』の第 1 章「Solaris Security Toolkit ソフトウェアの構成およびカスタマイズ」を参照してください。

3. カスタムスクリプトとファイルを変更します。

   コード例 5-1 では、install-openssh.fin を使用してソフトウェアを自動的にインストールする方法を示しています。このコーディング例では、OpenSSH のバージョンを "2.5.2p2" としていますが、OpenSSH の現在のバージョンは "3.5p1" です。当然、インストールするバージョンは、そのソフトウェアがインストールされた時期によって異なります。このスクリプトは、市販の Secure Shell 製品をサポートするように変更することもできます。

コード例 5-1 install-openssh.fin スクリプト例

```
#!/bin/sh
# NOTE: This script is not intended to be used for Solaris 9+.
  logMessage "Installing OpenSSH software.\n"
if check_os_revision 5.5.1 5.8; then
   OPENSSH_VERSION="2.5.2p2"
   OPENSSH_NAME="OBSDssh"
   OPENSSH_PKG_SRC="${OPENSSH_NAME}-${OPENSSH_VERSION}-`uname -p`
`uname -m`-`uname -r`.pkg"
   OPENSSH_PKG_DIR="${JASS_ROOT_DIR}/${JASS_PACKAGE_DIR}"
# Install the OpenSSH package onto the client
   if [ "${JASS_STANDALONE}" = "1" ]; then
      logNotice "This script cannot be used in standalone mode due
to the potential for overwriting the local OBSHssh installation."
   else
logMessage "Installing ${OPENSSH_NAME} from
${OPENSSH_PKG_DIR}/${OPENSSH_PKG_SRC}"
      if [ -f ${OPENSSH_PKG_DIR}/${OPENSSH_PKG_SRC} ]; then
         add_pkg -d ${OPENSSH_PKG_DIR}/${OPENSSH_PKG_SRC}
${OPENSSH_NAME} add_to_manifest X "pkgrm ${OPENSSH_NAME}"
      else
      logFileNotFound "${OPENSSH_NAME}"
[...]
```

この場合、異なるバージョンの OpenSSH をサポートするようにこのスクリプトを変更する唯一の方法は、直接このスクリプトを変更する方法です。変更を終了したら、スクリプトの新しい名前がわかるように、このスクリプトを使用しているセキュリティープロファイルを変更してください。

注 – すでに述べたように、Solaris Security Toolkit ソフトウェアのほとんどの機能は変数でカスタマイズできるので、このスクリプトを直接変更する方法が必要となることはめったにないはずです。

kill スクリプトが無効にされないようにする

注 – Solaris 10 OS を実行するシステム、および Solaris 10 OS で smf(5) に完全に変換されたサービスに関しては、次の節は適用されません。これらの init.d スクリプトはもはや使われなくなり、代わりに svc.startd(1M) がこれらの機能を制御します。これらのサービスに対しては、Solaris 10 OS では Solaris Security Toolkit は JASS_KILL_SCRIPT_DISABLE 変数をまったく使用しません。SMF がすべての起動と停止を処理するため、起動スクリプトと停止スクリプトの分離は必要なくなりました。

通常、キーワード disable から始まる終了スクリプトは、サービスを無効にします。このスクリプトの多くは、実行コントロールディレクトリ (/etc/rc*.d) に格納されているシェルスクリプトを変更します。ほとんどの場合、実行コントロールスクリプトは、start スクリプトと kill スクリプトの 2 つのタイプに分類されます。その名前からわかるように、start スクリプトはサービスを開始し、kill スクリプトはサービスを停止します。また、start スクリプト名は大文字の S から始まり、kill スクリプト名は大文字の K から始まります。

kill スクリプトは、システムのシャットダウンまたは再起動の準備段階で頻繁に使用されます。kill スクリプトは、変更が失われることなくシステム状態が維持されるように、論理的な順序でサービスをシャットダウンします。一般に、start スクリプトと killスクリプトはどちらも、/etc/init.d ディレクトリ内のファイルへのハードリンクです (ただし、必ずそうであるとは限りません)。

Solaris Security Toolkit ソフトウェアのデフォルトのアクションでは、start スクリプトと killスクリプトがいずれも無効になります。この無効化は、JASS_KILL_SCRIPT_DISABLE 環境変数を使用すると変更できます。デフォルトでは、この変数は 1 に設定されており、Solaris Security Toolkit ソフトウェアに start スクリプトと kill スクリプトの両方を無効にするよう指示しています。

このアクションが実行されない方がよい場合があります。たとえば、kill スクリプトは、管理者が手動で開始したサービスを停止するときによく使用されます。Solaris Security Toolkit ソフトウェアでこのスクリプトを無効にした場合、こういったサービスが正常に停止されないか、または正しい順序で停止されないことがあります。kill スクリプトが無効にされないようにするには、user.init ファイルまたはその関連ドライバで、JASS_KILL_SCRIPT_DISABLE 環境変数を 0 に設定してください。

新しい終了スクリプトを作成する

新しい終了スクリプトを作成して、配備する Solaris Security Toolkit ソフトウェアに統合することができます。多くの終了スクリプトは Bourne シェルで作成されるため、新機能を追加するのは比較的簡単です。Solaris 10 OS では、スタンドアロンでの監査と強化で Perl が使用できるため、Solaris 10 OS を実行しているシステム用の Solaris Security Toolkit スクリプトは Perl で書くことができます。UNIX シェルスクリプトの作成経験があまりない開発者は、同様の機能を実行する既存の終了スクリプトを調べて、目的のタスクを実行する方法を習得し、アクションの正しい順序を理解してください。

新しい終了スクリプトを作成するときは、以下の規則を考慮してください。これらの規則を理解していれば、スタンドアロンモードおよび JumpStart モードでスクリプトは確実に機能するようになります。

新しい終了スクリプトを追加するときは、対応する監査スクリプトも必ず追加してください。監査スクリプトは、既存システム上での変更の状態を判定するときに使用されるスクリプトです。詳細については、第 6 章を参照してください。

- 終了スクリプトが相対 `root` ディレクトリに適合していることを確認すること。

  / ディレクトリがシステムの実際の `root` ディレクトリであることを前提にして、スクリプトを構成しないでください。スクリプトの構成が適切でないと、ターゲットの実際の `root` ディレクトリが `/a` であった場合、スクリプトは JumpStart モードでは機能しなくなります。この規則は、`JASS_ROOT_DIR` 環境変数を使用すれば簡単に実現できます。この環境変数とその他の環境変数についての詳細は、第 7 章を参照してください。

  ときには、終了スクリプトで使用するプログラムが、再配置された `root` ディレクトリをサポートしていない場合があります。その場合には、すでに説明したように、`chroot`(1M) コマンドを相対 `root` ディレクトリ内で実行する必要があります。たとえば、`usermod`(1M) コマンドは、ユーザーに代替 `root` ディレクトリの指定を許可しません。この場合、次のように `chroot`(1M) コマンドを使用する必要があります。

```
chroot ${JASS_ROOT_DIR} /usr/sbin/usermod ...arguments...
```

  Solaris Security Toolkit ソフトウェアでは、プラットフォームの実際の `root` ディレクトリの位置を自動的に検出し、その値を `JASS_ROOT_DIR` 変数に割り当てます。`root` ファイルシステムの特定のパスをハードコードするのではなく、この変数を使用してください。たとえば、終了スクリプト内で `/etc/default/login` を使用する代わりに、`JASS_ROOT_DIR/etc/default/login` を使用してください。

- 可能であれば、新規ディレクトリの作成、ファイルのコピー、既存ファイルのバックアップを行うときに、Solaris Security Toolkit ソフトウェアのフレームワークを使用すること。

フレームワーク関数を使用すると、新しいスクリプトによる変更が、他の場所での変更と整合性が保たれていることと、変更を安全に元に戻せることが保証されます。フレームワーク関数のリストについては、第 2 章を参照してください。

Solaris Security Toolkit のすべての機能を正しく、矛盾のないように動作させるフレームワーク関数の例は次のとおりです。

- `backup_file`
- `create_a_file`
- `disable_conf_file`
- `disable_rc_file`
- `disable_service`
- `enable_service`

- 可能な限り、サポートされている標準の方法を使用して、システムの構成や調整を行うこと。

  たとえば、`usermod`(1M) のようなプログラムは、`/etc/passwd` ファイルを直接変更するより優先されます。ソフトウェアにできる限り柔軟性を持たせ、作成される終了スクリプトをできる限り OS のバージョンに依存しないものにするためには、このような優先は必要です。システムの構成方法が複雑であったりあいまいであったりすると、スクリプトの使用期間でのデバッグや維持管理が困難になる可能性があります。サポート可能な変更方法に関する例については、Sun BluePrints OnLine 掲載記事『Solaris Operating Environment Security: Updated for Solaris Operating Environment 9』を参照してください。

- 新規終了スクリプトは OS バージョンを認識可能であること。

  ある関数が 1 つのバージョンの OS で必要ない場合は、その関数を使用しないでください。これにより、ソフトウェアが既存リリースと下位互換性を持つようになり、さらに今後のリリースをサポートする可能性が高くなります。その上、終了スクリプトで OS バージョンを認識可能であれば、警告とエラーメッセージの数を飛躍的に減少させることができます。Solaris Security Toolkit ソフトウェアの終了ディレクトリには、そのスクリプトが使用されている OS を認識可能で、必要な場合に変更するだけでよいサンプルスクリプトが格納されています。この機能を使用しているサンプルスクリプトは、次のとおりです。

  - `enable-rfc1948.fin`
  - `install-ftpusers.fin`

  ソフトウェア開発者に対してこのプロセスをさらに簡単にするために、次の 2 つの関数がフレームワークに含まれています。

  - `check_os_min_revision`
  - `check_os_revision`

  これらの関数についての詳細は、第 2 章を参照してください。

- 終了スクリプトを作成またはカスタマイズする際の最後の考慮事項は、1 つのプラットフォームで Solaris Security Toolkit ソフトウェアを 2 回以上実行できるようにすること。

  終了スクリプトは、実際に変更を行う必要があるかどうかを検出できなければなりません。

たとえば、/etc/default/inetinit スクリプトにすでに TCP_STRONG_ISS=2 という設定があるかどうかを、enable-rfc1948.fin スクリプトでチェックするとします。この設定があれば、ファイルをバックアップしたり、他の変更を行う必要はありません。

```
if [ `grep -c "TCP_STRONG_ISS=2" ${INETINIT}` = 0 ]; then
# The following command will remove any exiting TCP_STRONG_ISS
# value and then insert a new one where TCP_STRONG_ISS is set
# to 2.  This value corresponds to enabling RFC 1948
# unique-per-connection ID sequence number generation.
logMessage "\nSetting 'TCP_STRONG_ISS' to '2' in ${INETINIT}.\n"
backup_file ${INETINIT}
cat ${INETINIT}.${JASS_SUFFIX} |\
sed '/TCP_STRONG_ISS=/d' > ${INETINIT}
echo "TCP_STRONG_ISS=2" >> ${INETINIT}
fi
```

この方法により、不必要なバックアップファイル数が減るだけでなく、同じファイルで何回も重複した変更を行うことから発生するエラーや混乱を防止することもできます。また、この機能を実装すると、終了スクリプトに対応している監査スクリプトの実装に必要なコードの作成も適切に実行することができます。

標準の終了スクリプトの使用

終了スクリプトは、セキュリティー強化処理時にシステムの変更と更新を行います。終了スクリプトがそのほかの処理や操作で使用されることはありません。

finish.init ファイルは、終了スクリプトの構成変数をすべて処理するファイルです。user.init ファイルを変更すると、デフォルトの変数を無効にすることができます。このファイルには、各変数と、その変数の終了スクリプトでの影響と使用についての詳しい説明が記載されています。また、各変数の説明については、第 7 章を参照してください。

finish.init スクリプトに含まれている変数を使用すると、組織のセキュリティーポリシーと要件に合うように、大部分の終了スクリプトをカスタマイズすることができます。変数を使用すれば Solaris Security Toolkit ソフトウェアのほとんどすべての部分をカスタマイズできるため、ソースコードを変更する必要はありません。新しい Solaris Security Toolkit ソフトウェアリリースに移行する際の問題をできるだけ少なくするために、このスクリプトを使用することを強く推奨します。

この節では、Finish ディレクトリに格納されている標準の終了スクリプトについて説明します。Finish ディレクトリ内のスクリプトは、以下のカテゴリに分かれています。

- 無効化 (disable)
- 有効化 (enable)
- インストール (install)
- 最小化 (minimize)
- 印刷 (print)
- 削除 (remove)
- 設定 (set)
- 更新 (update)

これらの標準の終了スクリプトに加え、Solaris Security Toolkit ソフトウェアでは、製品固有の終了スクリプトも提供しています。製品固有の終了スクリプトのリストについては、182 ページの「製品固有の終了スクリプトの使用」を参照してください。

無効化 (disable) 終了スクリプト

この節では、以下の無効化 (disable) 終了スクリプトについて説明します。

disable-ab2.fin

注 – Solaris OS 9 以降では ab2 ソフトウェアは使用されなくなったため、このスクリプトは Solaris OS バージョン 2.5.1 ～ 8 を実行しているシステムでのみ使用します。

このスクリプトは、AnswerBook2™ (ab2) サーバーが起動しないようにします。ab2 サーバーソフトウェアは、Solaris OS Server パックの Documentation CD-ROM に収録されています。

disable-apache.fin

注 – このスクリプトは、Solaris OS バージョン 8 および 9 を実行しているシステムでのみ使用します。

このスクリプトは、Solaris OS バージョン 8 と 9 の配布パッケージに付属している Apache Web サーバーのみが起動しないようにします。システムにインストールされているそのほかの Apache ソフトウェアには影響を与えません。このサービスについての詳細は、apache(1M) のマニュアルページを参照してください。

disable-apache2.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、Solaris 10 OS の配布パッケージにのみ付属している Apache 2 サービスのみが起動しないようにします。システムにインストールされているその他の Apache ソフトウェアには影響を与えません。このサービスについての詳細は、apache(1M) のマニュアルページを参照してください。

disable-appserv.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、Solaris 10 オペレーティングシステムの配布パッケージに付属している Sun Java™ Application Server が起動しないようにします。

disable-asppp.fin

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 8 にのみ使用します。Solaris OS バージョン 9 および 10 では、このサービスは PPP サービスに置き換わっており、disable-ppp.fin 終了スクリプトを使用して無効にします。

このスクリプトは、非同期ポイントツーポイントプロトコル (ASPPP) サービスが開始されないようにします。このサービスは、RFC 1331「The Point-to-Point Protocol (PPP) for the transmission of multi-protocol datagrams over Point-to-Point links」に記載されている機能を実装します。この機能についての詳細は、aspppd(1M) のマニュアルページを参照してください。

disable-autoinst.fin

注意 – sys-unconfig(1M) プログラムで提供されている機能を使用して、システムの構成を製造時の状態に戻す必要がある場合には、disable-autoinst.fin スクリプトを使用しないでください。

注意 – JumpStart 環境を使用している場合は、以下の段落で説明する実行コントロール (起動) スクリプトを無効にして、侵入者によるシステムの再構成を防止してください。これらの実行コントロールスクリプトは、JumpStart 環境では使用しません。

このスクリプトは、自動構成に関連する実行コントロールスクリプトを無効にして、システムが再インストールされないようにします。実行コントロールスクリプトが使用されるのは、`/etc/.UNCONFIGURED` ファイルまたは `/AUTOINSTALL` ファイルが作成されている場合だけです。最初のインストールと構成が終われば、通常はこういったスクリプトを使用可能にしておく理由はありません。

disable-automount.fin

注 – この NFS 自動マウントサービスは、遠隔手続き呼び出し (RPC) ポートマッパーに依存するため、`disable-automount.fin` が使用されていないときは、`disable-rpc.fin` スクリプトも使用しないでください。

このスクリプトは、NFS 自動マウントサービスを無効にします。自動マウントサービスは、`autofs` ファイルシステムからのファイルシステムのマウント要求とアンマウント要求に応答します。このスクリプトが使用されると、NFS 自動マウントサービスが無効になり、すべての自動マウントマップ形式が影響を受けます。この機能についての詳細は、`automountd`(1M) のマニュアルページを参照してください。

disable-dhcp.fin

注 – このスクリプトは、Solaris OS バージョン 8、9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、Solaris OS バージョン 8、9、および 10 に搭載されている Dynamic Host Configuration Protocol (DHCP) サーバーを無効にします。このサービスについての詳細は、`dhcpd`(1M) のマニュアルページを参照してください。

disable-directory.fin

注 – このスクリプトは、Solaris OS バージョン 9 および 10 に付属している Sun Java System Directory Server にのみ使用します。

このスクリプトは、Sun Java System Directory Server (旧称 Sun ONE Directory Server) が起動しないようにします。別パッケージの製品や、9 および 10 以外のバージョンの Solaris OS に付属している Sun Java System Directory Server には影響を与えません。デフォルトでは、Solaris Security Toolkit ソフトウェアが無効にするのは、Solaris OS で提供されているサービスだけです。このサービスについての詳細は、`directoryserver`(1M) のマニュアルページを参照してください。

disable-dmi.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、デスクトップ管理インタフェース (DMI) が起動しないようにします。このサービスについての詳細は、`dmispd`(1M) と `snmpXdmid`(1M) のマニュアルページを参照してください。

disable-dtlogin.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

注 – このサービスは、RPC ポートマッパーに依存するため、`disable-rpc.fin` が**使用されていない**ときは、`disable-dtlogin.fin` スクリプトも**使用しないでください**。

このスクリプトは、起動時に、共通デスクトップ環境 (CDE) サービスなどのウィンドウ環境が開始されないようにします。ただし、後からは (システムの起動後など)、ウィンドウ環境が開始されないようにはしません。このサービスについての詳細は、`dtlogin`(1X) と `dtconfig`(1) のマニュアルページを参照してください。

disable-face-log.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

SUNWfac のパッケージである Framed Access Command Environment (FACE) には誰でも書き込み可能なログファイル /usr/oasys/tmp/TERRLOG が含まれています。このスクリプトはグループおよびその他のユーザーの書き込み権を削除するため、root アカウントのみがファイルに書き込むことができます。つまり、スクリプトはファイル上のアクセス権を次のように変更します。

変更前:

```
-rw--w--w-
```

変更後:

```
-rw-------
```

ログファイル /usr/oasys/tmp/TERRLOG は、/var ではなく、多くの場合ルートファイルシステム上にある /usr の下にあるため、これはサービス拒否攻撃に使用される可能性があります。FACE ログは便利な機能ですが、システム操作には重要ではない場合があります。この機能が必要でない場合は、無効にします。

disable-IIim.fin

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、Internet-Intranet Input Method (IIim) デーモンと HyperText Transfer (htt) サーバーが起動しないようにします。IIim デーモンはあるポートにバインドされている htt エージェントで、htt ソフトウェアからの要求を待ち受けます。要求を受け取ると、IIim はその要求を処理し、必要な情報を収集し、必要な操作を実行して、最終的には要求元に情報を返します。IIim は、韓国語、簡体字中国語、繁体字中国語などの国際言語で情報を伝送する際に特に便利です。

disable-ipv6.fin

注 – このスクリプトは、Solaris OS バージョン 8、9 および 10 を実行しているシステムでのみ使用します。IPv6 機能がシステムで必要な場合は、このスクリプトを使用しないでください。

このスクリプトは、/etc/hostname6.* 内の関連ホスト名ファイルを削除することにより、特定のネットワークインタフェースでの IPv6 の使用を無効にします。また、これにより、in.ndpd サービスが実行されなくなります。

disable-kdc.fin

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、Kerberos 鍵配布センター (KDC) サービスが開始されないようにします。

- **Solaris 9 OS** の場合、`JASS_DISABLE_MODE` が `conf` に設定されていると、`kdc.conf` ファイルが無効になるので、Kerberos クライアントとしての機能に影響することに注意してください。Kerberos クライアントとしてシステムを機能させる必要がある場合は、このスクリプトをこの方法では使用しないでください。
- **Solaris 10 OS** の場合、`disable_service()` 関数を使用して `krb5kdc` FMRI を無効にします。

このサービスについての詳細は、`krb5kdc`(1M) と `kdc.conf`(4) のマニュアルページを参照してください。

disable-keyboard-abort.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

注 – 一部のシステムには、キースイッチを安全位置に設定できるものがあります。これらのシステムでは、キースイッチを安全位置に設定すると、このコマンドによるソフトウェアのデフォルト設定が無効になります。

このスクリプトは、システムがキーボードのアボートシーケンスを無視するように構成します。通常、キーボードのアボートシーケンスが開始されると、オペレーティングシステムが中断されて、コンソールは OpenBoot™ PROM モニターまたはデバッガに入ります。このスクリプトを使用すると、システムは中断されなくなります。この機能についての詳細は、`kbd`(1) のマニュアルページを参照してください。

disable-keyserv-uid-nobody.fin

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、nobody UID によるセキュリティー保護された RPC へのアクセスを無効にします。

- Solaris OS バージョン 9 および 10 では、/etc/init.d/rpc の ENABLE_NOBODY_KEYS 変数を NO に設定すると、アクセスが無効になります。
- Solaris 9 OS より前のバージョンでは、/etc/init.d/rpc 実行コントロールファイルの keyserv コマンドに -d オプションを追加すると、アクセスが無効になります。

このサービスについての詳細は、keyserv(1M) のマニュアルページを参照してください。

disable-ldap-client.fin

注 – このスクリプトは、Solaris OS バージョン 8、9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、Lightweight Directory Access Protocol (LDAP) クライアントデーモンがシステム上で起動しないようにします。この LDAP クライアントデーモンは、システムにディレクトリ検索機能を提供するサービスです。システムが LDAP クライアントとして機能していたり、ディレクトリ検索機能を必要とする場合は、このスクリプトを使用しないでください。このサービスについての詳細は、ldap_cachemgr(1M) と ldapclient(1M) のマニュアルページを参照してください。

disable-lp.fin

このスクリプトは、ラインプリンタ (lp) サービスが開始されないようにします。このスクリプトは、サービスを無効にするだけでなく、lp を /etc/cron.d/cron.deny ファイルに追加して、/var/spool/cron/crontabs ディレクトリ内の lp コマンドをすべて削除することで、cron サブシステムへの lp ユーザーのアクセスも削除します。

lp パッケージがシステムにインストールされている場合もいない場合もあるため、このスクリプトは update-cron-deny.fin スクリプトとは機能が異なります。また、cron-deny-update.fin スクリプトで削除される機能は必要ありませんが、lp サブシステムは必要な場合があります。

disable-mipagent.fin

注 – このスクリプトは、Solaris OS バージョン 8、9、および 10 を実行しているシステムでのみ使用します。

このスクリプトは、モバイルインターネットプロトコル (MIP) エージェントが開始されないようにします。このエージェントは、RFC 2002 「IP Mobility Support」 に記載されている MIP ホームエージェントと外部エージェント機能を実装するサービスです。このサービスについての詳細は、`mipagent`(1M) のマニュアルページを参照してください。

disable-named.fin

注 – このスクリプトは、Solaris OS に付属しているドメインネームシステム (DNS) サービス専用です。このサービスを無効にしても、システムが DNS クライアントとして動作する機能には影響しません。

このスクリプトは、`named`(1M) コマンドで DNS サーバーが起動されないようにします。

disable-nfs-client.fin

注 – NFS クライアントサービスが必要な場合には、このスクリプトを使用しないでください。また、このサービスは RPC サービスに依存しているため、`disable-rpc.fin` スクリプトも**使用しないでください**。

このスクリプトは、NFS クライアントサービスが開始されないようにします。また、ネットワークステータスモニター (`statd`) デーモンとロックマネージャー (`lockd`) デーモンも無効にします。このスクリプトを使用した場合でも、管理者はシステム上に遠隔ファイルシステムをマウントすることができます。しかし、遠隔ファイルシステムでは、ステータスモニターデーモンやロックマネージャーデーモンは利用できません。このサービスについての詳細は、`statd`(1M) と `lockd`(1M) のマニュアルページを参照してください。

disable-nfs-server.fin

注 – システムが遠隔クライアントとファイルシステムを共有する必要がある場合は、このスクリプトを使用しないでください。NFS サーバーサービスが必要な場合には、このスクリプトを使用しないでください。また、このサービスは RPC サービスに依存しているため、`disable-rpc.fin` スクリプトも**使用しないでください**。

このスクリプトは、NFS サービスが開始されないようにします。また、NFS のログ、マウント、アクセスチェック、およびクライアントサービスのサポートを提供するデーモンも無効にします。このサービスについての詳細は、`nfsd`(1M)、`mountd`(1M)、および `dfstab`(4) のマニュアルページを参照してください。

disable-nscd-caching.fin

注意 – ネームサービスを集中的に使用しているシステムでは、パフォーマンスに影響する可能性があります。

このスクリプトは、ネームサービスキャッシュデーモン (NSCD) による `passwd`、`group`、`hosts`、および `ipnodes` エントリのキャッシュを無効にします。Solaris 8 OS では、役割によるアクセス制御 (RBAC) 機能のバグを修正する、パッチ 110386 バージョン 02 以降が適用されている必要があります。適用されていない場合は、エラーメッセージが表示されます。

NSCD は、ネームサービス要求のキャッシュを行います。このデーモンは、保留中の要求に対するパフォーマンスを向上させ、ネームサービスのネットワークトラフィックを削減するために用意されています。`nscd` では、`passwd`、`group`、`hosts` などのデータベースのキャッシュエントリを保持します。セキュリティー上の理由から、このデーモンではシャドウパスワードファイルはキャッシュしません。システムライブラリコールによるネームサービス要求はすべて、`nscd` に送られます。Solaris 8 OS では IPv6 と RBAC が追加されたため、`nscd` キャッシング機能は、さらに多くのネームサービスデータベースに対応するよう拡張されました。

ネームサービスデータをキャッシュすることで スプーフィング攻撃を受けやすくなるため、必要最低限のデータのみをキャッシュするように `nscd` の構成を変更することをお勧めします。ネームサービス要求がスプーフィング攻撃を受けやすいと考えられるときに、`/etc/nscd.conf` ファイルで正の数の生存時間 (`ttl`) をゼロに設定すれば、構成が変更されます。具体的には、`passwd`、`group`、および Solaris 8、9 および 10 OS RBAC 情報の正負の数の `ttl` がゼロになるように、構成を変更してください。

`nscd -g` オプションは、サーバー上の現在の `nscd` 構成を表示するときに使用することができ、`nscd` を調整する際に役立つオプションです。

アプリケーションでは直接ネームサービス呼び出しを行なって、これによりアプリケーションとネームサービスバックエンドのさまざまなバグを見つけ出しているため、完全に nscd を無効にすることはお勧めしません。

disable-picld.fin

注 – このスクリプトは、Solaris OS バージョン 8 および 9 を実行しているシステムでのみ使用します。

このスクリプトは Platform Information and Control Library (PICL) サービスが開始されないようにします。このサービスを無効にすると、環境条件を監視するシステムの機能に影響することがあるため、注意して使用してください。このサービスについての詳細は、picld(1M) のマニュアルページを参照してください。

disable-power-mgmt.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムにのみ適用されます。

このスクリプトは、電源管理サービスが開始されないようにします。(電源管理サービスを使用すると、モニターの電源が切断し、ディスクが停止して、システム自体の電源も切断します。) このスクリプトを使用すると、電源管理機能が無効になります。また、システム管理者が JumpStart モードの自動インストール時に電源管理状態に関する問い合わせを受けないように、noautoshutdown ファイルも作成されます。このサービスについての詳細は、powerd(1M)、pmconfig(1M)、および power.conf(4) のマニュアルページを参照してください。

disable-ppp.fin

注 – このスクリプトは、Solaris OS バージョン 8、9、および 10 を実行しているシステムでのみ使用します。

このスクリプトは、ポイントツーポイントプロトコル (PPP) サービスが開始されないようにします。このサービスは Solaris 8 OS (7/01) から導入され、以前の非同期 PPP サービスを補完するものです。このサービスでは、シリアルポイントツーポイントリンクを介して、データグラムを伝送する方法を提供します。このサービスについての詳細は、pppd(1M) と pppoed(1M) のマニュアルページを参照してください。

disable-preserve.fin

このスクリプトは、システムの再起動時に、保存されている (すでに編集が終わった) ファイルが /usr/preserve に移動されないようにします。通常、こういったファイルは、システムのクラッシュやセッションの損失が原因で突然終了したエディタによって作成されます。これらのファイルは、ファイル名の先頭に Ex が追加されて、/var/tmp に格納されます。

disable-remote-root-login.fin

このスクリプトは、直接遠隔 root ログインを行わないように、/etc/default/login ファイルの CONSOLE 変数を変更します。これは Solaris OS 2.5.1 最終更新バージョン以降でのデフォルトの動作でしたが、この設定が変更されないようにこのスクリプトが用意されています。この設定は、Secure Shell など、システムへのアクセス許可に /bin/login プログラムを使用しないように構成できるプログラムには影響を与えません。この機能についての詳細は、login(1) のマニュアルページを参照してください。

disable-rhosts.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、/etc/pam.conf の Pluggable Authentication Module (PAM) 構成を変更して、rlogin と rsh に対する rhosts 認証を無効にします。

disable-rlogin-rhosts.fin 終了スクリプトは、そのアクションがわかりやすいように disable-rhosts.fin に名前が変更されました。また、rhosts 認証がいずれのサービスでも無効となるように、/etc/pam.conf ファイルの rsh エントリと rlogin エントリが両方ともコメントアウトされます。

この機能についての詳細は、in.rshd(1M)、in.rlogind(1M)、および pam.conf(4) のマニュアルページを参照してください。

disable-routing.fin

このスクリプトは、あるネットワークから別のネットワークへのネットワークパケットのルーティング、つまり「パケット転送」を無効にします。

- Solaris 9 OS またはそれ以前では、ルーティングを無効にするには、/etc/notrouter ファイルを作成します。
- Solaris 10 OS では、/usr/bin/routeadm を使用してルーティングを無効にします。

disable-rpc.fin

注意 – システムで、自動マウント、NFS、ネットワーク情報サービス (NIS)、NIS+、CDE、およびボリューム管理 (Solaris OS バージョン 9 および 10 のみ) のいずれかのサービスを使用している場合は、RPC ポートマッパー機能を無効にしないでください。

このスクリプトは、遠隔手続き呼び出し (RPC) サービスが開始されないようにします。このサービスを無効にすると、NFS や CDE などの付属サービスと、Sun Cluster ソフトウェアなどの別パッケージのサービスに影響を与えることに注意してください。また、サン以外のソフトウェアパッケージには、このサービスが利用可能であることを前提としているものもあります。このサービスを無効にするときは、RPC サービスを必要とするサービスやツールがないことを確認してから行なってください。このサービスについての詳細は、`rpcbind`(1M) のマニュアルページを参照してください。

注意 – `secure.driver` ドライバを使用してセキュリティー保護されたシステムは、`disable-rpc.fin` スクリプトが含まれているため、JumpStart や NIS を使用できません。代わりに、`disable-rpc.fin` スクリプトを含まない新しいドライバを作成する必要があります。

disable-samba.fin

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、Samba ファイルと印刷の共有サービスが開始されないようにします。このスクリプトで無効にするのは、Solaris OS Distribution パッケージに含まれている Samba サービスだけです。システムにインストールされているそのほかの Samba ソフトウェアには影響を与えません。このサービスについての詳細は、`smbd`(1M)、`nmbd`(1M)、および `smb.conf`(4) のマニュアルページを参照してください。

disable-sendmail.fin

注 – Solaris Security Toolkit ソフトウェアの変更では、Solaris OS で電子メールが「受信」されないようにするだけです。送信する電子メールは正常に処理されます。

このスクリプトは、システムが実行している Solaris OS のバージョンに基づいて、さまざまな sendmail のオプションを無効にします。

- Solaris 10 OS では、このスクリプトにより sendmail サービスはほかのホストからのメールを受信しません。このスクリプトは変更された sendmail 構成を作成およびインストールします。この構成により、sendmail デーモンは IPv4 ループバックインタフェース上でのみ待機します。
- Solaris 9 OS では、また別の sendmail オプションが実装されており、デーモンはループバックインタフェース上で待機のみを行います。詳細については、Sun BluePrints OnLine 掲載記事『Solaris Operating Environment Security: Updated for Solaris Operating Environment 9』を参照してください。
- Solaris 8 OS では、同様の機能を実装する /etc/default/sendmail ファイルがインストールされています。この方法では送信電子メールをパージするため、デーモンを継続的に実行させるよりもセキュリティーが確保されます。
- Solaris OS バージョン 2.5.1、2.6、および 7 では、このスクリプトは、sendmail デーモンの起動スクリプトと停止スクリプトを無効にし、cron サブシステムに 1 時間に 1 回 sendmail を実行するエントリを追加します。

disable-slp.fin

注 – このスクリプトは、Solaris OS バージョン 8、9、および 10 を実行しているシステムでのみ使用します。

このスクリプトは、サービスロケーションプロトコル (SLP) サービスが開始されないようにします。このサービスは、Internet Engineering Task Force (IETF) により発行される RFC 2165 および RFC 2608 の定義に従って、SLP バージョン 1 および 2 に共通サーバー機能を提供します。SLP は、ネットワークサービスの検出と選択を行うスケーラブルなフレームワークを提供します。このサービスについての詳細は、slpd(1M) のマニュアルページを参照してください。

disable-sma.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、NET-SNMP サービスに基づくシステム管理エージェント (SMA) サービスが開始されないようにします。

disable-snmp.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、Simple Network Management Protocol (SNMP) サービスが開始されないようにします。システムで機能しているサン以外の SNMP エージェントを、このスクリプトで無効にすることはできません。このスクリプトで無効にできるのは、Solaris OS 配布パッケージに搭載されている SNMP エージェントだけです。このサービスについての詳細は、`snmpdx`(1M) と `mibiisa`(1M) のマニュアルページを参照してください。

disable-spc.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、SunSoft™ Print Client 起動スクリプトをすべて無効にします。

disable-ssh-root-login.fin

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、`root` アカウントへの遠隔アクセスを制限するように、Solaris OS バージョン 9 および 10 に搭載されている Secure Shell サービスを構成します。デフォルトでは、Solaris 9 および 10 OS に付属の Secure Shell によって、遠隔 `root` アクセスが拒否されます。このスクリプトは Secure Shell サービスの機能を確認して、それによって、`disable-remote-root-login.fin` スクリプトの機能と同様のメカニズムを実行します。`/etc/ssh/sshd_config` の `PermitRootLogin` パラメータが `no` に設定されます。この機能についての詳細は、`sshd_config`(4) のマニュアルページを参照してください。

disable-syslogd-listen.fin

注 – SYSLOG サーバーはネットワーク上のほかのマシンに関する SYSLOG メッセージを待機および受信できなければなりませんが、その機能はこの終了スクリプトにより無効になっているため、このスクリプトは SYSLOG サーバーでは使用しないでください。このスクリプトは、Solaris OS バージョン 8、9、および 10 を実行しているシステムでのみ使用します。

このスクリプトは、ログシステムメッセージ (syslogd) サービスが遠隔ログメッセージを受け取らないようにします。

- Solaris 8 OS では、-t オプションが syslogd(1M) コマンド行に追加されます。
- Solaris OS バージョン 9 および 10 では、/etc/default/syslogd ファイルの LOG_FROM_REMOTE 変数が NO に設定されます。

このスクリプトによって、デーモンが User Diagram Protocol (UDP) ポート 514 で待機しなくなります。SYSLOG メッセージをローカルに格納する、または SYSLOG メッセージを別のネットワークアクセス可能なシステムに転送するシステムでは、このスクリプトは便利です。

disable-system-accounts.fin

このスクリプトは、root 以外の特定の未使用のシステムアカウントを無効にします。JASS_ACCT_DISABLE 変数に、システムで無効にするアカウントのリストを明示的に列挙します。

disable-uucp.fin

このスクリプトは、UNIX-to-UNIX コピープログラム (UUCP) 起動スクリプトを無効にします。また、nuucp システムアカウントが、/var/spool/cron/crontabs ディレクトリ内の uucp crontab エントリとともに削除されます。このサービスについての詳細は、uucp(1C) と uucico(1M) のマニュアルページを参照してください。

disable-vold.fin

注 – リムーバブルメディア (フロッピーディスク、CD-ROM など) の自動取り付けおよび取り外しを行う必要がある場合は、このスクリプトを**使用しないでください**。

注 – Solaris 9 OS で VOLD サービスが必要な場合は、このスクリプトを使用しないでください。また、このサービスは RPC サービスと `rpc.smserverd` サービスに依存しているため、これらのサービスを**無効にしないでください**。同様に、`rpc.smserverd` サービスを無効にしないときには、サービスを誤って無効にしないように、`JASS_SVCS_ENABLE` 環境変数に、RPC サービス番号 `100155` (または Solaris 10 OS の場合は `svc:/network/rpc/smserver:default`) を追加する必要があります。

このスクリプトは、ボリューム管理デーモン (VOLD) が起動しないようにします。`vold` は、`/vol` に格納されるファイルシステムイメージを作成して維持管理するデーモンです。デフォルトでは、このイメージには、フロッピーディスク、CD-ROM などのリムーバブルメディアデバイスの記号名が含まれます。このサービスについての詳細は、`vold`(1M) のマニュアルページを参照してください。

disable-wbem.fin

注 – このスクリプトは、Solaris OS バージョン 8、9、および 10 を実行しているシステムでのみ使用します。

注 – WBEM サービスが必要な場合、または Solaris Management Console を使用する必要がある場合は、このスクリプトを使用しないでください。また、このサービスは RPC サービスに依存しているため、`disable-rpc.fin` スクリプトも使用しないでください。

このスクリプトは、Web-Based Enterprise Management (WBEM) サービスが開始されないようにします。WBEM は、エンタープライズコンピューティング環境の管理を統合する、管理およびインターネット関連テクノロジです。Distributed Management Task Force (DMTF) によって開発された WBEM を使用すると、組織で WWW テクノロジをサポートおよび促進する標準ベースの統合管理ツールセットを配布できるようになります。このサービスについての詳細は、`wbem`(5) のマニュアルページを参照してください。

disable-xfs-fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、フォントファイルをクライアントにサービス提供する TCP/IP ベースのサービスである、X Font Server (XFS) を無効にします。XFS は、X ベースのグラフィカルユーザーインタフェース (GUI) の実行には必要ありません。

disable-xserver.listen.fin

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、ポート 6000 で TCP を介して要求を待機および受信する X11 サーバー機能を無効にします。このスクリプトは、`/etc/dt/config/Xservers` ファイルの X サーバー構成行に、オプション `-nolisten` TCP を追加します。このファイルが存在していない場合は、`/usr/dt/config/Xservers` の元の場所からコピーされます。この機能についての詳細は、`Xserver`(1) のマニュアルページを参照してください。

有効化 (enable) 終了スクリプト

この節では、以下の有効化 (enable) 終了スクリプトについて説明します。

enable-account-lockout.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、`/etc/security/policy.conf` ファイルの `LOCK_AFTER_RETRIES` 変数の値を正しく定義します。定義されると、`LOCK_AFTER_RETRIES` で指定されている値をアカウントが超えれば、そのアカウントはロックされ、ロックを解除するには管理者の支援が必要になります。

注意 – システム管理者によりアカウントのロックが解除されると、そのパスワードは削除されます。未承認のログインを防止するため、そのアカウントにはただちに新しいパスワードを設定する必要があります。

enable-bart.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

基本監査報告機能 (BART) は、ファイルシステムレベル全体で動作するファイル追跡ツールです。BART を使用すると、配備されたシステムにインストールされているソフトウェアスタックのコンポーネントに関する情報を、すばやく、簡単かつ確実に収集できます。BART を使用すると、時間がかかる管理作業を単純化することによって、システムのネットワーク管理コストを大幅に削減できます。

BART によって、既知のベースラインを基準として、システム上でどのファイルレベルの変更が起こったかを判断できます。`bart create` コマンドは、完全にインストールおよび構成されたシステムから、ベースラインまたは「制御」マニフェストを作成します。`bart compare` コマンドは、このベースラインと、あとの時点のシステムのスナップショットを比較し、インストールされてからシステムで発生したファイルレベルの変更をリストするレポートを生成します。

注 – `svc` は、Solaris Security Toolkit の制御下にない `/etc` の下にあるファイルを編集するため、`bart compare` コマンドが失敗する場合があります。これらの失敗は実際には失敗ではない場合がありますが、ログを確認する必要があります。

Solaris Security Toolkit 4.2 ソフトウェアは、次の 2 つの BART 規則ファイルをインストールします。

- `secure.driver` 用の `rules-secure` (コード例 5-2)。デフォルトの場所は `/var/opt/SUNWjass/BART/rules-secure` です

コード例 5-2 デフォルトの BART `rules-secure` ファイル

```
/                           !core !tmp/ !var/ !S82mkdtab
        CHECK all
        IGNORE contents mtime

        /etc/rc*.d          S* !S82mkdtab
        sbin                !core
        /usr/bin            !core
        /usr/sbin           !core
        CHECK contents
```

■ ほかのすべてのドライバ用の rules (コード例 5-3)。デフォルトの場所は /var/opt/SUNWjass/BART/rules です

コード例 5-3 デフォルトの BART rules ファイル

```
/etc/rc*.d          S* !S82mkdtab
sbin                !core
/usr/bin            !core
/usr/sbin           !core
CHECK contents
```

システムの BART ファイルレベルチェックからの出力は、/var/opt/SUNWjass/BART/manifests ディレクトリの JASS_TIMESTAMP.txt ファイルに格納されます。

この enable-bart.fin スクリプトが BART を有効にします。このスクリプトが、BART 規則ファイルが存在するかどうか、また存在する場合はその構成が実行中のドライバおよびその BART 規則ファイルと矛盾がないかどうかを判断します。

BART 規則ファイルの構成が実行中のドライバおよびその BART 規則ファイルと矛盾する場合、スクリプトは $JASS_FILES/var/opt/SUNWjass/bart/ から規則ファイルをコピーします。正しい BART 構成ファイルが配置されれば、スクリプトは BART を実行して、/var/opt/SUNWjass/BART/manifests に *JASS_TIMESTAMP*.txt という形式の新しいマニフェストファイル (20050711152248.txt など) を生成します。

注 – Solaris Security Toolkit 4.2 ソフトウェアには、BART マニフェストファイルをチェックするためのインタフェースは用意されていません。

---

enable-bsm.fin

注 – このスクリプトは、Solaris OS バージョン 8 ～ 10 を実行しているシステムでのみ使用します。Solaris 10 OS では、子ゾーンで BSM を有効にする前に、必ずまず大域ゾーンで BSM を有効にしてください。

このスクリプトは、SunSHIELD™ Solaris 基本セキュリティーモジュール (BSM) 監査サービスを有効にします。また、Sun BluePrints OnLine 掲載記事『Auditing in the Solaris 8 Operating Environment』に記載されているデフォルトの監査構成もインストールします。必要に応じて、audit_warn という別名が追加され、これには root アカウントが割り当てられます。また、アボートシーケンスを許可するために、アボート無効コードが無効にされます。この設定は、プラットフォームへの物理的なアクセスが常に可能とは限らない、完全自動データセンター環境で最も使用され

ています。システムの再起動後に、Solaris BSM サブシステムが有効になり、監査が開始されます。このサービスについての詳細は、bsmconv(1M) のマニュアルページを参照してください。

enable-coreadm.fin

注 – このスクリプトは、Solaris OS バージョン 7 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、Solaris OS バージョン 7 ～ 10 に搭載されている coreadm 機能を構成します。生成されたコアファイルを、JASS_CORE_DIR で指定したディレクトリに格納するようにシステムを構成します。さらに、コアファイルに関する情報が収集できるように、各コアファイルには JASS_CORE_PATTERN で指定されたタグが付けられます。通常、収集される情報には、実行可能なプロセス名とコアファイルの作成日時だけでなく、プロセス ID、実効ユーザー ID、およびプロセスの実効グループ ID も含まれます。この機能についての詳細は、coreadm(1M) のマニュアルページを参照してください。

enable-ftpaccess.fin

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、Solaris 9 および Solaris 10 OS での FTP サービスの ftpaccess 機能を有効にします。set-banner-ftp.fin スクリプトと set-ftpd-umask.fin スクリプトで行ったセキュリティー変更を使用するためには、この機能が必要です。たとえば、デフォルトのあいさつ文、ファイル生成マスク、およびそのほかのパラメータを設定する変更は、ftpaccess(4) マニュアルページに記載されています。

- **Solaris 9 OS** では、このスクリプトは、/etc/inet/inetd.conf ファイルの in.ftpd エントリに -a 引数を追加します。
- **Solaris 10 OS** では、svc:/network/ftp inetdstart/exec プロパティーに a オプションが追加されます。

詳細は、in.ftpd(1M) のマニュアルページを参照してください。

enable-ftp-syslog.fin

このスクリプトは、in.ftpd デーモンに、SYSLOG サブシステム経由で試みられたファイル転送プロトコル (FTP) アクセスをすべて記録させます。

- **Solaris 9 OS** およびそれ以前のバージョンでは、/etc/inetd/inetd.conf ファイルの in.ftpd コマンドに -l オプションを追加すると、このオプションが有効になります。
- **Solaris 10 OS** では、svc:/network/ftp inetdstart/exec プロパティーに l オプションが追加されます。

  詳細は、in.ftpd(1M) のマニュアルページを参照してください。

enable-inetd-syslog.fin

このスクリプトは、着信 TCP 接続要求をすべて記録するように、インターネットサービスデーモン (INETD) を構成します。つまり、inetd デーモンが待機している TCP サービスに接続されると、SYSLOG によるログエントリが行われます。

- **Solaris 9 OS** より前のバージョンでは、-t オプションを inetd コマンド行に追加してログを有効にします。
- **Solaris 9 OS** では、/etc/default/inetd ファイルの ENABLE_CONNECTION_LOGGING 変数を YES に設定します。
- **Solaris 10 OS** では、svc:/network/inetd サービスに関して、defaults/tcp_trace プロパティーが true に設定されます。

  詳細は、inetd.conf(4) のマニュアルページを参照してください。

enable-ipfilter.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

Solaris 10 OS は、内容により IP パケットにフィルタを適用するフリーウェアの IP フィルタ (ipfilter) を統合することで、統合ファイアウォール機能を実現しています。このスクリプトは、すべての使用可能なネットワークインタフェースに関して ipfilter を有効にし、実行中のドライバに固有の規則のデフォルトセットを作成します。これらの構成済み規則ファイルは file_copy キーワード接尾辞を使用して、どのファイルがどのドライバと関連付けられているかを特定します。

次の構成済み IPF 規則が Solaris Security Toolkit の $JASS/FILES/etc/opt/ipf ディレクトリに含まれています。

- secure.driver 用の ipf.conf 構成ファイル – ipfilter は次の ipf.conf ファイルを使用して、デフォルトで有効

コード例 5-4 secure.driver デフォルトの IP フィルタ規則ファイル

```
# to load/reload rules use /sbin/ipf -Fa -f /etc/opt/ipf/ipf.conf

block in log proto tcp from any to any
```

コード例 5-4　secure.driver デフォルトの IP フィルタ規則ファイル (続き)

```
block in log proto udp from any to any

# allow connections originating from local machine out
pass out quick proto tcp from any to any flags S/SA keep state
pass out quick proto udp from any to any keep state
```

- server-secure.driver 用の ipf.conf-server 構成ファイル – ipfilter は次の ipf.conf ファイルを使用して、デフォルトで有効

コード例 5-5　server-secure.driver デフォルトの IP フィルタ規則ファイル

```
# to load/reload rules use /sbin/ipf -Fa -f /etc/opt/ipf/ipf.conf

block in log proto tcp from any to any
block in log proto udp from any to any

# allow connections originating from local machine out
pass out quick proto tcp from any to any flags S/SA keep state
pass out quick proto udp from any to any keep state

# allow ssh (port 22)
# (these ip-addresses are also protected by tcp-wrappers)
# (if you change it here, you also need to change /etc/hosts.allow)
pass in quick proto tcp from any to any port = 22
```

- sunfire_15k_sc-secure.driver 用の ipf.conf-15k-sc 構成ファイル – ipfilter は次の ipf.conf ファイルを使用して、デフォルトで有効

コード例 5-6　sunfire_15k_sc-secure.driver デフォルトの IP フィルタ規則ファイル

```
# to load/reload rules use /sbin/ipf -Fa -f /etc/opt/ipf/ipf.conf

block in log proto tcp from any to any
block in log proto udp from any to any

# allow connections originating from local machine out
pass out quick proto tcp from any to any flags S/SA keep state
pass out quick proto udp from any to any keep state

# allow ssh (port 22)
# (these ip-addresses are also protected by tcp-wrappers)
# (if you change it here, you also need to change /etc/hosts.allow)
pass in quick proto tcp from any to any port = 22

# allow all necessary communication in from other SC
pass out quick proto tcp from any to any flags S/SA keep state
pass out quick proto udp from any to any keep state
```

注 – Sun Cluster 3x ソフトウェアは IP フィルタをサポートしないため、suncluster3x-secure.driver ではこのスクリプトを使用しないでください。

enable-ipfilter.fin スクリプトは次の動作を行います。

- plumb されていて、/etc/ipf/pfil.ap ファイルに存在しないインタフェースをチェックし、必要に応じてそれらを監査または追加する。バックアップファイルには存在しないインタフェースがある場合、スクリプトはそれらを追加します。『Solaris Security Toolkit 4.2 マニュアルページガイド』またはマニュアルページで ipfilter(5) コマンドを参照してください。
- システム上の既存の /etc/ip/ipf.conf ファイルを確認し、それがキーワード固有のファイルと同じであるかどうかを確認する。キーワード固有のファイルが同じでない場合、キーワード固有のオプションを使用して、スクリプトは既存の /etc/opt/ipf/ipf.conf ファイルをバックアップし、$JASS_FILES/etc/opt/ipf/ipf.conf ファイルをコピーします。
- サービス管理機能 (SMF) を介して svcadm enable ipfilter コマンドを使用し、network/ipfilter サービスを有効にする。

enable-password-history.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、ドライバの JASS_PASS_HISTORY 環境変数に基づいてさまざまな HISTORY 値の定義を許可することにより、システム上のパスワードの履歴チェックを有効にします。/etc/default/passwd ファイルをチェックし、HISTORY 値が指定されているかどうかを判断します。

- HISTORY 値が /etc/default/passwd ファイルで指定されている場合、スクリプトはその値を JASS_PASS_HISTORY 環境変数内の値と照合し、その値が正しいかどうかを確認します。
- HISTORY 値が、JASS_PASS_HISTORY 環境変数で指定されている値とは異なる、または適切に設定されていない場合、スクリプトはその値を訂正します。

enable-priv-nfs-ports.fin

このスクリプトは、制限付き NFS ポートアクセスを有効にするように、/etc/system ファイルを変更します。変数を設定すると、1024 より小さいポートから発信されている NFS 要求だけが受信されます。

/etc/system ファイルでキーワード値ペアが正しく定義されていない場合は、ファイルで値が書き直されます。まだ定義されていない場合は、キーワード値ペアがファイルに付加されます。

enable-process-accounting.fin

必要な Solaris OS パッケージ (現在は、SUNWaccr と SUNWaccu) がシステムにインストールされている場合、このスクリプトは、Solaris OS プロセスアカウンティングを有効にします。このサービスについての詳細は、acct(1M) のマニュアルページを参照してください。

enable-rfc1948.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、RFC 1948 のサポートを有効にする /etc/default/inetinit ファイルを作成または変更します。(この RFC では、コネクションごとに一意の ID シーケンス番号を生成する方法が定義されています。) /etc/default/inetinit ファイルの変数 TCP_STRONG_ISS を 2 に設定します。詳細は、http://ietf.org/rfc1948.html を参照してください。

enable-stack-protection.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行している SPARC システムでのみ使用します。

注 – この機能を有効にすると、システムが SPARC バージョン 8 のアプリケーションバイナリインタフェース (ABI) に適合しなくなるため、一部のアプリケーションで問題が発生する可能性があります。

SPARC システムでのみ、このスクリプトは、/etc/system ファイルを変更して、スタックの保護と例外のログ記録を有効にします。noexec_user_stack と noexec_user_stack_log を /etc/systemファイルに追加すると、これらのオプションが有効になります。

すでに /etc/system ファイルでキーワード値ペアが定義されている場合は、このファイルの値が正しく設定されていることを確認するために書き直されます。まだ定義されていない場合は、キーワード値ペアがファイルに付加されます。このスクリプトの変数を設定してシステムを再起動すると、スタックを直接実行しようとしても拒否され、SYSLOG を介して試みられたスタック実行が記録されます。一般的なバッファーオーバーフロー攻撃からシステムを保護するときに、この機能を有効にします。

Solaris OS バージョン 9 および 10 では、コアの Solaris 実行可能ファイルの多くがマップファイル (/usr/lib/ld/map.noexstk) に対してリンクされています。このマップファイルでプログラムのスタックを実行不可にすると、このスクリプトと同様の機能が提供されます。しかし、この変更はシステムに対してグローバルなものとなるため、このスクリプトをそのまま使用することをお勧めします。

enable-tcpwrappers.fin

注 – このスクリプトは、付属の TCP ラッパーパッケージを使用している Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

注 – サンプルの hosts.allow ファイルと hosts.deny ファイルは、組織に適した構成となるようにカスタマイズしてから使用してください。ファイルテンプレートは、JASS_ROOT_DIR/Files/etc に格納されています。

このスクリプトは、TCP ラッパーを使用するようにシステムを構成します。Solaris 9 OS の最新アップデートと Solaris 10 OS のすべてのリリースに付属する TCP ラッパーを使用すると、管理者は TCP サービスへのアクセスを制限することができます。デフォルトでは、/etc/inet/inetd.conf ファイルで stream, nowait として定義されているすべてのサービスが保護されます。このスクリプトは、ENABLE_TCPWRAPPERS パラメータを YES に設定するように /etc/default/inetd ファイルを構成します。さらに、TCP ラッパーで保護されるサービスへのアクセスを制御する、サンプルの /etc/hosts.allow ファイルと /etc/hosts.deny ファイルもインストールします。

Solaris 10 OS 専用**:**

- inetd の tcp_wrappers の使用を有効にする
- rpcbind の tcp_wrappers の使用を有効にする
- hosts.allow|deny ファイルのキーワード固有バージョンをコピーする

インストール (install) 終了スクリプト

この節では、以下のインストール (install) 終了スクリプトについて説明します。

- 166 ページの「install-newaliases.fin」
- 166 ページの「install-openssh.fin」
- 167 ページの「install-recommended-patches.fin」
- 167 ページの「install-sadmind-options.fin」
- 167 ページの「install-security-mode.fin」
- 167 ページの「install-shells.fin」
- 168 ページの「install-strong-permissions.fin」
- 168 ページの「install-sulog.fin」
- 168 ページの「install-templates.fin」

install-at-allow.fin

このスクリプトは、/etc/cron.d に at.allow ファイルを作成して、at コマンドの実行を制限します。そして、このファイルに、JASS_AT_ALLOW 変数で定義されたユーザーのリストを格納します。at アクセスを要求するユーザーはすべて、この at.allow ファイルに追加する必要があります。at 機能と batch 機能へのアクセスを判定するときには、このスクリプトを update-at-deny.fin スクリプトとともに使用してください。この機能についての詳細は、at(1) のマニュアルページを参照してください。

install-fix-modes.fin

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 9 を実行しているシステムでのみ使用します。FixModes ソフトウェアで実行される変更は Solaris 9 OS に統合されていますが、別パッケージのアプリケーションやサン以外のアプリケーションの多くにとって FixModes を使用するメリットがあるため、今までとおり FixModes を使用することをお勧めします。

このスクリプトは、JASS_PACKAGE_DIR ディレクトリからクライアントに fix-modes ソフトウェアをコピーし、そのプログラムを実行します。FixModes ソフトウェアは、Solaris システムのアクセス権を強化するために使用します。

install-ftpusers.fin

このスクリプトは、FTP サービスへのアクセスの制限に使用される ftpusers ファイルを作成または変更します。また、JASS_FTPUSERS 変数にリストされているユーザーを ftpusers ファイルに追加します。ただし、ユーザーを追加するのは、このファイルにユーザー名が含まれていない場合だけです。

デフォルトの ftpusers ファイルは、Solaris OS バージョン 8、9、および 10 に含まれています。このファイルへのパスはバージョンによって異なります。

- Solaris 9 および 10 OS のパスは、/etc/ftpd です。

■ Solaris OS 8 以前のファイルパスは /etc です。

着信 FTP サービスの使用を**許可しない**アカウントをすべて、このファイルで指定する必要があります。少なくともこのファイルには、root アカウントだけでなく、すべてのシステムアカウント (たとえば、bin、uucp、smtp、sys など) を含めるようにしてください。これらのアカウントは、侵入者や未承認アクセスを試みるユーザーのターゲットとなることが多いためです。Telnet 経由でのサーバーへの root アクセスが無効になることはよくありますが、FTP 経由での root アクセスが無効になることはありません。この構成では、変更した構成ファイルをアップロードしてシステム構成を変更しようとする侵入者は、バックドアを設置できます。

install-jass.fin

このスクリプトは、Solaris Security Toolkit ソフトウェアが実行されるときに、JumpStart クライアント上に自動的に Solaris Security Toolkit ソフトウェアをインストールします。クライアントの初期インストール後に Solaris Security Toolkit ソフトウェアが実行可能な状態になるように、この方法を使用します。Solaris Security Toolkit ソフトウェアパッケージのインストールは、Solaris OS コマンド pkgadd を使用して行われます。このスクリプトでは、Solaris Security Toolkit ソフトウェアが JASS_PACKAGE_DIR ディレクトリにインストールされていることが前提となっています。Solaris Security Toolkit ソフトウェアパッケージ SUNWjass は、デフォルトでは /opt ディレクトリにインストールされます。

install-loginlog.fin

このスクリプトは、システムで使用される /var/adm/loginlog ファイルを作成し、失敗したログインを記録します。失敗したログインは、ログインの最大失敗回数を超えてから記録されます。この回数は、/etc/default/login 構成ファイルに設定されている RETRIES 変数で指定します。set-login-retries.fin スクリプトも参照してください。詳細は、loginlog(4) のマニュアルページを参照してください。

install-md5.fin

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 9 を実行しているシステムでのみ使用します。

このスクリプトは、メッセージダイジェスト 5 (MD5) アルゴリズムソフトウェアを自動的にインストールします。このソフトウェアは、ファイルシステムオブジェクトのデジタルフィンガープリントを作成するときに使用されます。このソフトウェアについては、Sun BluePrints OnLine 掲載記事『The Solaris Fingerprint Database - A

Security Tool for Solaris Software and Files』に記載されています。デフォルトでは、MD5 ソフトウェアは JASS_MD5_DIR パラメータで指定されているディレクトリにインストールされます。

install-nddconfig.fin

このスクリプトは、nddconfig ファイルをインストールします。このファイルは、Sun BluePrints OnLine 掲載記事『Solaris Operating Environment Network Settings for Security』に基づき、よりセキュリティー保護された値を各種ネットワークパラメータに設定するときに使用されます。

install-newaliases.fin

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 8 を実行しているシステムでのみ使用します。

このスクリプトは、newaliases シンボリックリンクを /usr/lib/sendmail プログラムに追加します。SUNWnisu パッケージがインストールされていないか、または削除されている場合、最小化インストールを行う際に、このリンクが必要となることがあります。このリンクは、newaliases が SUNWnisu パッケージに含まれていた Solaris OS 2.5.1 ～ 8 を実行しているシステムで必要です。

install-openssh.fin

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 8 を実行しているシステムでのみ使用します。Solaris 9 および 10 OS には Secure Shell ソフトウェアが含まれているので、Solaris 9 または 10 OS をインストールするときはこのスクリプトは使用しません。

このスクリプトは、OpenSSH の OpenBSD バージョンを /opt/OBSDssh にインストールします。このスクリプトが記述されているパッケージは、Sun BluePrints OnLine 掲載記事『Configuring OpenSSH for the Solaris Operating Environment』の記載に基づいています。ホストキーが存在する場合は、このスクリプトはホストキーを上書きしません。

OBSDssh-3.5p1-sparc-sun4u-5.8.pkg というストリーム形式パッケージの Solaris OS が、JASS_PACKAGE_DIR ディレクトリ内にあることを前提として、インストールが行われます。

install-recommended-patches.fin

注 – このスクリプトは、Solaris OS 2.5.1 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、JASS_HOME_DIR/Patches ディレクトリにあるパッチを JumpStart サーバーにインストールします。スクリプトが正常に実行できるように、推奨およびセキュリティパッチクラスタをダウンロードし、JASS_HOME_DIR/Patches ディレクトリに圧縮解除します。

install-sadmind-options.fin

注 – このスクリプトは、Solaris OS 2.5.1 ～ 9 を実行しているシステムでのみ使用します。

このスクリプトは、JASS_SADMIND_OPTIONS 環境変数で指定されているオプションを、/etc/inet/inetd.conf ファイルの sadmind デーモンのエントリに追加します。このサービスについての詳細は、sadmind(1M) のマニュアルページを参照してください。

install-security-mode.fin

注 – このスクリプトは、SPARC ベースのシステムにのみ使用します。

このスクリプトは、OpenBoot PROM セキュリティーモードの現在の状態を表示します。このスクリプトでは直接 EEPROM パスワードを設定しないため、JumpStart のインストール時に EEPROM パスワードの設定をスクリプト作成することはできません。スクリプトを出力すると、コマンド行から EEPROM パスワードを設定する方法がわかります。この機能についての詳細は、eeprom(1M) のマニュアルページを参照してください。

install-shells.fin

注 – このスクリプトでシェルを /etc/shells ファイルに追加するのは、シェルがシステム上に存在し、実行可能で、このファイルに含まれていない場合だけです。

このスクリプトは、JASS_SHELLS 環境変数で指定されているユーザーシェルを /etc/shells ファイルに追加します。Solaris OS 関数 getusershell(3C) は、システム上の有効なシェルを判定するときに /etc/shells ファイルで主に使用されます。詳細は、shells(4) のマニュアルページを参照してください。JASS_SHELLS 環境変数についての詳細は、277 ページの「JASS_SHELLS」を参照してください。

install-strong-permissions.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでは使用しないでください。

注意 – Solaris Security Toolkit ソフトウェアでは、このスクリプトによる変更を自動的に元に戻すことはできないため、このスクリプトを使用するときは注意してください。このスクリプトで設定するアクセス権が、使用する環境とアプリケーションに適していることを確認してください。

このスクリプトは、システム上でグループおよびユーザーのアクセスを制限することにより、各種のアクセス権と所有権を変更して、セキュリティーを向上させます。

Solaris 10 OS には多くのアクセス権と所有権の変更が組み込まれているため、このスクリプトは Solaris 10 OS には使用しません。このスクリプトを実行できないわけではありませんが、Solaris 10 OS への変更点を考慮すると、実行結果によりセキュリティーが改善されることはありません。

install-sulog.fin

このスクリプトは、すべてのスーパーユーザー (su)の試行動作を記録できるようにする /var/adm/sulog ファイルを作成します。この機能についての詳細は、sulog(4) のマニュアルページを参照してください。

install-templates.fin

注 – このスクリプトは特別な用途のためなので、ドライバから直接呼び出さないでください。

JASS_FILES パラメータまたはいずれかの OS 固有値が空でない場合に、このスクリプトが driver.run プログラムから直接呼び出されます。このスクリプトは、対象システム上にファイルテンプレートのコピーを自動的に作成します。当初、この機能

は driver.run スクリプトにありましたが、ファイルテンプレートの検証をよりサポートするために別スクリプトに分けられました。この終了スクリプトは、必要となったときに、JASS_FILES パラメータの内容に基づいて最初に実行されます。

印刷 (print) 終了スクリプト

この節では、以下の印刷 (print) 終了スクリプトについて説明します。

print-jass-environment.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでは使用しないでください。

このスクリプトは、Solaris Security Toolkit ソフトウェアで使用される環境変数をすべて印刷します。このスクリプトは診断を目的としたもので、多くの場合、環境変数を使用する前にその状態を記録できるように、ドライバの最初に呼び出されます。

print-jumpstart-environment.fin

このスクリプトは、JumpStart のインストールで使用されるすべての環境変数を印刷します。このスクリプトは診断を目的としたもので、JumpStart のインストール時に発生した問題のデバッグに役立ちます。

print-rhosts.fin

注 – スクリプトが必要とする追加の処理時間が受け入れられる場合、print-rhosts.fin スクリプトは手動で有効にする必要があります。

このスクリプトは、JASS_ROOT_DIR ディレクトリ内の任意のディレクトリに格納されている、.rhosts ファイルと hosts.equiv ファイルをすべて一覧表示します。JASS_RHOSTS_FILE 変数が定義されていない場合、結果は標準出力に表示されます。この変数が定義されている場合は、結果はすべて指定されているファイルに書き込まれます。

print-sgid-files.fin

このスクリプトは、設定グループ ID アクセス権を持つ JASS_ROOT_DIR ディレクトリ内の任意のディレクトリにあるファイルをすべて印刷します。JASS_SGID_FILE 変数が定義されていない場合、結果は標準出力に表示されます。この変数が定義されている場合は、結果はすべて指定されているファイルに書き込まれます。

print-suid-files.fin

このスクリプトは、設定ユーザー ID アクセス権を持つ JASS_ROOT_DIRディレクトリ内の任意のディレクトリにあるファイルをすべて印刷します。JASS_SUID_FILE 変数が定義されていない場合、結果は標準出力に表示されます。この変数が定義されている場合は、結果はすべて指定されているファイルに書き込まれます。

print-unowned-objects.fin

このスクリプトは、有効なユーザーやグループが割り当てられていないシステム上のファイル、ディレクトリ、およびそのほかのオブジェクトをすべて、JASS_ROOT_DIR から順に一覧表示します。JASS_UNOWNED_FILE 変数が定義されていない場合、結果は標準出力に表示されます。この変数が定義されている場合は、結果はすべて指定されているファイルに書き込まれます。

print-world-writable-objects.fin

このスクリプトは、システム上の world-writable オブジェクトをすべて、JASS_ROOT_DIR から順に一覧表示します。JASS_WRITABLE_FILE 変数が定義されていない場合、結果は標準出力に表示されます。この変数が定義されている場合は、結果はすべて指定されているファイルに書き込まれます。

削除 (remove) 終了スクリプト

この節では、以下の削除 (remove) 終了スクリプトについて説明します。

■ 171 ページの「remove-unneeded-accounts.fin」

remove-unneeded-accounts.fin

注 – このスクリプトは、Solaris OS 2.5.1 ～ 9 を実行しているシステムでのみ使用します。

remove-unneeded-accounts.fin スクリプトは、passmgmt コマンドを使用して、/etc/passwd ファイルと /etc/shadow ファイルから、未使用の Solaris OS アカウントを削除します。このスクリプトは、JASS_ACCT_REMOVE 変数で定義されているアカウントを削除します。

設定 (set) 終了スクリプト

この節では、以下の設定 (set) 終了スクリプトについて説明します。

set-banner-dtlogin.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、dtlogin サービスのサービスバナーをインストールします。このバナーは、共通デスクトップ環境 (CDE) または GNU ネットワークオブジェクトモデル環境 (GNOME) で提供されるようなグラフィカルインタフェースを使用して、システムへの認証に成功すると、ユーザーに表示されます。このスクリプトは、ファイルテンプレート JASS_ROOT_DIR/etc/dt/config/Xsession.d/0050.warning で指定されているファイルの内容を表示するようにシステムを構成します。デフォルトでは、/etc/motd ファイルの内容が表示されます。

set-banner-ftpd.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、次のファイル転送プロトコル (FTP) サービスバナーをインストールします。

- Solaris 8 OS 以前では、このバナーは /etc/default/ftpd ファイルで JASS_BANNER_FTPD 変数を使用して定義されます。
- Solaris 9 および 10 OS では、このバナーは /etc/ftpd/banner.msg ファイルを使用して定義されます。詳細は、in.ftpd(1M) または ftpaccess(4) (Solaris 9 または 10 OS の場合) のマニュアルページを参照してください。

注 – install-ftpaccess.fin スクリプトを使用していない場合、Solaris 9 または 10 OS システムでは、set-banner-ftpd.fin スクリプトを使って行った変更は有効になりません。

---

set-banner-sendmail.fin

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、変数 JASS_BANNER_SENDMAIL で定義されている Sendmail サービスバナーをインストールします。このバナーは、/etc/mail/sendmail.cf ファイルの SmtpGreetingMessage パラメータまたは De パラメータを使用して定義されます。Solaris OS バージョン 9 ～ 10 では、SmtpGreetingMessage パラメータが使用されます。

詳細は、sendmail(1M) のマニュアルページを参照してください。

set-banner-sshd.fin

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、システムの認証の前に `/etc/issue` の内容をユーザーに表示するよう Secure Shell サービスを構成することにより、Secure Shell サービスバナーをインストールします。このタスクは、`/etc/ssh/sshd_config` ファイルで `Banner` パラメータを `/etc/issue` に設定すれば実行されます。この機能についての詳細は、`sshd_config`(4) のマニュアルページを参照してください。

set-banner-telnet.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、変数 `JASS_BANNER_TELNET` で定義されている Telnet サービスバナーをインストールします。このバナーは、`/etc/default/telnetd` ファイルの `BANNER` 変数を使用して定義されます。詳細は、`in.telnetd`(1M) のマニュアルページを参照してください。

set-flexible-crypt.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

Solaris 10 OS には、システム上でパスワード暗号化に使用されるアルゴリズムを制御する、新しいチューニング可能属性がいくつか導入されています。新しいアルゴリズムは、LDAP、NIS+、および NIS を使用したネームサービスベースの格納だけでなく、ローカルのパスワード格納に使用できます。ネームサービスに対してこの機能を有効にする手順については、『Solaris 10 システム管理 (セキュリティサービス)』の「システム、ファイル、およびデバイスのセキュリティ」の章を参照してください。

このスクリプトは、ローカルに格納されたパスワードに対してさまざまなパスワードハッシュアルゴリズムを使用することで、強力なパスワードを使用できるようにしています。すべてのパスワードを失効させるのは `secure.driver` のみであるため、ユーザーは、新しい暗号化アルゴリズムで暗号化された新しいパスワードの選択を強制されます。

チューニング可能属性は、次のように /etc/security/policy.conf ファイルに追加されています。

コード例 5-7 Solaris Security Toolkit ドライバ用のパスワード暗号化チューニング可能属性

```
secure.driver:
        CRYPT_ALGORITHMS_ALLOW = 1,2a,md5
        CRYPT_DEFAULT = 1
        JASS_FORCE_CRYPT_EXPIRE = 1
server-secure.driver:
        CRYPT_ALGORITHMS_ALLOW = 1,2a,md5
        CRYPT_DEFAULT = 1
        JASS_FORCE_CRYPT_EXPIRE = 0
suncluster3x-secure:
        CRYPT_ALGORITHMS_ALLOW = 1,2a,md5
        CRYPT_DEFAULT = 1
        JASS_FORCE_CRYPT_EXPIRE = 0
sunfire_15k_sc-secure:
        CRYPT_ALGORITHMS_ALLOW = 1,2a,md5
        CRYPT_DEFAULT = 1
        JASS_FORCE_CRYPT_EXPIRE = 0
```

CRYPT_ALGORITHMS_ALLOW の値は、次のようにマッピングされています。

- 1 – BSD/Linux md5
- 2a – BSD Blowfish
- md5 – Sun md5

secure.driver のパスワードが失効するのは次の場合です。

- JASS_FORCE_CRYPT_EXPIRE が 1 であり、かつ
- Solaris Security Toolkit により最後の policy.conf 変更が行われてからパスワードが失効していない、または
- この処理中に構成が変更されている

そのほかすべてのドライバは、ユーザーがユーザーパスワードを変更する際に、新しい暗号化アルゴリズムを使用してパスワードが再度暗号化されることを知らせるメッセージを表示します。

set-ftpd-umask.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、FTP サービス用のデフォルトのファイル生成マスクを設定します。

- Solaris 9 OS より前のバージョンでは、JASS_FTPD_UMASK 変数で定義されている UMASK 値を /etc/default/ftpd ファイルに追加することで、デフォルトのファイル生成マスクを設定します。
- Solaris 9 および 10 OS では、/etc/ftpd/ftpaccess ファイルで定義されている defumask パラメータを設定します。詳細は、in.ftpd(1M) または ftpaccess(4) (Solaris 9 または 10 OS の場合) のマニュアルページを参照してください。

注 – install-ftpaccess.fin スクリプトを使用していない場合、Solaris 9 または 10 OS システムでは、set-ftpd-umask.fin スクリプトを使って行った変更は有効になりません。

---

set-login-retries.fin

このスクリプトは、/etc/default/login ファイルの RETRIES 変数を、JASS_LOGIN_RETRIES 変数で定義されている値に設定します。ログしきい値を下げると、より多くの情報を取得できます。install-loginlog.fin スクリプトを使用すると、試みて失敗したログインを記録できます。この機能についての詳細は、login(1) のマニュアルページを参照してください。

set-power-restrictions.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

注 – このスクリプトは、たとえば PROM プロンプトでの power off など、ソフトウェアで制御可能な電源装置に対してのみ機能します。

このスクリプトは、JASS_POWER_MGT_USER 変数と JASS_CPR_MGT_USER 変数を使用して、/etc/default/power の構成を変更し、電源管理機能へのユーザーアクセスを制限します。その結果、システムの電源管理と中断/再開機能へのアクセスが制御されます。

set-rmmount-nosuid.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。Solaris OS バージョン 8 ～ 10 はデフォルトで、nosuid オプションを指定してリムーバブルメディアをマウントするように構成されています。このスクリプトは、デフォルトの設定に関係なく、必要なチェックを実行します。

このスクリプトは、/etc/rmmount.conf ファイルに 2 つのエントリを追加して、Set-UID ファイルのマウントを無効にします。システムにアクセスできるユーザーがフロッピーディスクや CD-ROM を挿入して Set-UID バイナリを読み込むと、システムが危険にさらされる可能性があるため、マウントを無効にすることが重要です。この機能についての詳細は、rmmount.conf(4) のマニュアルページを参照してください。

set-root-group.fin

このスクリプトは、root ユーザーの一次グループを JASS_ROOT_GROUP へ、さらにグループ識別子 #1 (GID 1、other) からグループ識別子 #0 (GID 0、root) へ変更します。これにより、root ユーザーと非特権ユーザーが共通グループになることはありません。

set-root-home-dir.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

多くの Solaris セキュリティー強化スクリプトおよび手順では、root アカウントにシングルスラッシュ (/) 以外のホームディレクトリを与えることを推奨しています。Solaris OS の root アカウントのホームディレクトリを変更すると、次のように、セキュリティーとシステム管理の点で利点があり、また Solaris OS とそのほかの UNIX システム (Linux/*BSD を含む) との互換性が高くなります。

- root アカウントのホームディレクトリのアクセス権を自動的に 0700 にすることができる。
- / の次の 3 つの一般的な使用法を区別することができる。
  - uid 0、loginame root のホームディレクトリとしての /
  - ユーザーのホームディレクトリが見つからなかった場合に自動的に割り当てられるホームディレクトリの値としての /。

    ルートディレクトリを /root に変更することで、独自のドットファイルではなく、root ユーザーのドットファイルを取得するリスクがなくなります。
  - ディレクトリツリーの最上位としての /

このスクリプトは、root アカウントが /etc/passwd ファイルに / のホームディレクトリを持っているかどうかを確認し、持っている場合このスクリプトは次の動作を行います。

- 所有権が root:root でアクセス権が 0700 である新しいディレクトリ /root を作成する
- 次のドットファイルが root により所有されている場合、/root に移動する
  - /.cshrc
  - /.profile
  - /.login
  - /.ssh
- 上記すべてに対するアクセス権を確認する
- usermod を介してルートホームディレクトリの定義を変更する

set-root-password.fin

注 – このスクリプトは、JumpStart ソフトウェアのインストール時にのみ実行されます。Solaris Security Toolkit ソフトウェアがコマンド行から起動された場合は、実行されません。

このスクリプトは、root パスワードを JASS_ROOT_PASSWORD で定義された初期値に自動的に設定します。このスクリプトで使用されるパスワードは、インストール時にのみ使用し、JumpStart のインストール処理が正常に終了したら、直ちに変更する必要があります。デフォルトでは、JASS_ROOT_PASSWORD パラメータで使用されるパスワードは t00lk1t です。

注意 – Solaris Security Toolkit は JumpStart モードで実行されると、root パスワードを設定します。そのあとで元に戻す処理が実行されると、root パスワードは以前の設定である「パスワードなし」に戻ってしまいます。これは、誰でもパスワードなしで root アカウントにログインできることを意味します。JumpStart インストールの直後に元に戻す処理を実行する場合は、必ず passwd(1) コマンドを使用して root パスワードを設定してください。

set-strict-password-checks.fin

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、ローカル環境内のユーザーに対するより厳しいパスワード要件をインストールします。Solaris 10 OS の passwd(1) コマンドは、より強力なユーザーパスワード用の新しい一連の機能を定義しています。Solaris Security Toolkit ソフトウェアは、デフォルト設定よりも強力なこれらの値を多数設定しています。このスクリプトは、さまざまなパスワードチェック用の正しい値が /etc/default/passwd ファイルの JASS_PASS_* 環境変数で正しく定義されていることを確認します。これらの環境変数およびそのほかの環境変数の定義と値については第 7 章を参照してください。

set-sys-suspend-restrictions.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、JASS_SUSPEND_PERMS 変数に基づいて、/etc/default/sys-suspend の構成を変更し、中断および再開機能へのユーザーアクセスを制限します。詳細は、sys-suspend(1M) のマニュアルページを参照してください。

set-system-umask.fin

このスクリプトは、実行コントロールスクリプトすべてが、JASS_UMASK の設定に基づいた安全なファイル生成マスクで実行されるようにします。不適切に選択されたファイル生成マスクを使用すると、重要なファイルが任意のユーザーから書き込み可能状態になるおそれがあるため、この設定は重要です。

- Solaris 8 OS より前のバージョンでは、このスクリプトが各実行レベルで起動スクリプトを作成し、それによってファイル生成マスクを JASS_UMASK に設定します。
- Solaris OS バージョン 8 ～ 10 では、/etc/default/init の CMASK 変数は JASS_UMASK に設定されます。この機能についての詳細は、init(1M) のマニュアルページを参照してください。

set-term-type.fin

このスクリプトは、システムで dtterm が認識されない問題を回避するように、vt100 のデフォルトの端末タイプを設定します。このスクリプトは主に、グラフィカルコンソールを備えておらず、通常は端末コンソールまたはそのほかのシリアルリンクを介してアクセスするシステムで使用されます。このスクリプトは便宜上の理由から用意されているだけであり、システムのセキュリティーには影響を与えません。

set-tmpfs-limit.fin

注 – Solaris 2.5.1 OS を実行するシステムでは、この set-tmpfs-limit.aud スクリプトの機能はサポートされていないため使用しないでください。

このスクリプトは、tmpfs ファイルシステムの一部として使用可能なディスク容量の制限を設定します。この制限により、メモリーの使い果たしを防止することができます。このスクリプトがデフォルトで制限する使用可能ディスク容量は、JASS_TMPFS_LIMIT で定義されている値です。この機能についての詳細は、mount_tmpfs(1M) のマニュアルページを参照してください。

set-user-password-reqs.fin

このスクリプトで行う変更によって、次回システムでパスワードが変更される際のシステムのパスワードポリシーが構成されます。セキュリティー強化処理によってアプリケーションと操作機能に悪影響がないように、このプロファイルをさらに調整する必要があります。

このスクリプトでは、以下の機能を有効にすると、より厳密なパスワード要件を使用できます。

- パスワードの有効期限
- パスワードの最小変更間隔
- パスワードの最小文字数

このスクリプトは、次の変数で定義されている値を使用して、/etc/default/passwd ファイルに適切なエントリを設定することで、要件を実装します。

- JASS_AGING_MINWEEKS
- JASS_AGING_MAXWEEKS
- JASS_AGING_WARNWEEKS
- JASS_PASSLENGTH

特に、非特権ユーザーからのアクセスがあるシステムで、このスクリプトの使用をお勧めします。

このスクリプトで変更するのは、/etc/default/passwd ファイルの設定だけです。任意のユーザーに対するパスワードの有効期限は有効にしません。パスワードの有効期限要件は、次回のパスワード変更時に、各ユーザーに対して実装されます。パスワード変更まで待たずに、パスワードの有効期限を有効にするには、passwd(1) コマンドを使用してください。

set-user-umask.fin

このスクリプトは、デフォルトのファイル生成マスク (UMASK) を、ユーザー起動ファイル /etc/.login、/etc/profile、/etc/skel/local.cshrc、/etc/skel/local.login、/etc/skel/local.profile、および /etc/default/login の JASS_UMASK で定義されている値に設定します。

更新 (update) 終了スクリプト

この節では、以下の更新 (update) 終了スクリプトについて説明します。

- 180 ページの「update-at-deny.fin」
- 180 ページの「update-cron-allow.fin」
- 180 ページの「update-cron-deny.fin」
- 181 ページの「update-cron-log-size.fin」
- 181 ページの「update-inetd-conf.fin」

update-at-deny.fin

このスクリプトは、JASS_AT_DENY にリストされているアカウントを、/etc/cron.d/at.deny ファイルに追加します。このスクリプトは、これらのアカウントを持つユーザーの at 機能と batch 機能の使用を拒否します。at 機能と batch 機能へのアクセスを判定するときには、このスクリプトを install-at-allow.fin スクリプトとともに使用します。この機能についての詳細は、at(1) のマニュアルページを参照してください。

update-cron-allow.fin

このスクリプトは、JASS_CRON_ALLOW にリストされているアカウントを、/etc/cron.d/cron.allow ファイルに追加します。このスクリプトは、これらのアカウントを持つユーザーの cron 機能の使用を許可します。cron 機能へのアクセスを判定するときには、このスクリプトを update-cron-deny.fin スクリプトとともに使用します。この機能についての詳細は、crontab(1) のマニュアルページを参照してください。

update-cron-deny.fin

このスクリプトは、JASS_CRON_DENY にリストされているアカウントを、/etc/cron.d/cron.deny ファイルに追加します。このスクリプトは、これらのアカウントを持つユーザーの cron 機能の使用を拒否します。cron 機能へのアクセスを判定するときには、このスクリプトを update-cron-allow.fin スクリプトとと

もに使用します。このスクリプトでは、root アカウントユーザーのアクセスは無効にしません。この機能についての詳細は、crontab(1) のマニュアルページを参照してください。

update-cron-log-size.fin

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、cron ログ情報の格納に使用する最大サイズ制限を調整します。

- Solaris 9 OS より前のバージョンでは、/etc/cron.d/logchecker スクリプトの LIMIT 変数を調整します。
- Solaris 9 および 10 OS では、(/var/cron/log エントリについて) /etc/logadm.conf ファイルの -s パラメータを調整します。

このスクリプトで使用されるサイズ制限は、JASS_CRON_LOG_SIZE 環境変数によって決まります。Solaris OS で定義されているデフォルトの制限は 0.5 MB です。

update-inetd-conf.fin

このスクリプトは、JASS_SVCS_DISABLE 変数で定義されている、inetd で始まるすべてのサービスを無効にします。つまり、このスクリプトでは、JASS_SVCS_ENABLE 変数で表示されるサービスを有効にします。同じサービスが両方の変数に含まれている場合は、そのサービスは有効になります。JASS_SVCS_ENABLE 変数の方が優先されます。

in.telnetd、in.ftpd、in.rshd などの共通サービスを含む、基本 OS のすべてのサービスが、Solaris OS バージョン 2.5.1 ～ 10 ではデフォルトで無効になります。

- **Solaris 9 OS** およびそれ以前のバージョンでは、/etc/inet/inetd.conf ファイルのサービスエントリの各行の先頭に # を挿入することで、サービスを無効にしています。別パッケージまたはサン以外のソフトウェアによってインストールされた追加サービスは、無効になりません。
- **Solaris 10 OS** では、サービスはサービス管理機能 (SMF) およびそのコマンド (svcadm(1M) など) により制御されます。

製品固有の終了スクリプトの使用

特定の Sun 製品のセキュリティー強化を行うために、製品固有の終了スクリプトが存在します。これらのスクリプトは、Finish ディレクトリに格納されています。製品固有の終了スクリプトを 表 5-1 に示します。

サンの新製品や更新されたサン製品のセキュリティーを強化するために、定期的に新しい終了スクリプトがリリースされます。最新のスクリプトのリストについては、以下の Security Web サイトを参照してください。

```
http://www.sun.com/security/jass
```

表 5-1 製品固有の終了スクリプト

製品	ドライバ名
Sun Cluster 3.x ソフトウェア	suncluster3x-set-nsswitch-conf.fin
Sun Fire ハイエンドシステムドメイン	s15k-static-arp.fin
Sun Fire ハイエンドシステムシステムコントローラ	s15k-static-arp.fin s15k-exclude-domains.fin s15k-sms-secure-failover.fin

suncluster3x-set-nsswitch-conf.fin

注 – このスクリプトは Sun Cluster 3.x システムにのみ使用します。ほかのシステムでは実行されません。

このスクリプトは、システムを Sun Cluster 3.x ノードとして自動的に構成します。このスクリプトは、クラスターキーワードを /etc/nsswitch.conf ファイルに設定して、Sun Cluster 3.x システムの配布を簡略化します。キーワードはホストフィールドに格納する必要があります。詳細については、Sun BluePrints OnLine 掲載記事『Securing Sun Cluster 3.x Software』を参照してください。

s15k-static-arp.fin

注 – このスクリプトは Sun Fire ハイエンドシステムの SC およびドメインにのみ使用します。ほかのシステムでは実行されません。このスクリプトは、System Management Services (SMS) バージョン 1.2 ～ 1.4.1 でのみ使用します。

このスクリプトは、I1 MAN ネットワーク上の静的 ARP アドレスの使用を有効にします。I1 MAN ネットワークは Sun Fire ハイエンドシステムのシャーシ内蔵ネットワークで、SC とドメイン間での TCP/IP ベースの通信に使用されます。動的 ARP の代わりに静的 ARP を使用することにより、SC に対する ARP ベースの攻撃を無効にします。

Sun Fire ハイエンドシステムのオプションの `s15k-static-arp.fin` スクリプトでは、以下の 4 つのファイルが使用されます。

- `/etc/sms_sc_arp`
- `/etc/sms_domain_arp`
- `/etc/rc2.d/S73sms_arpconfig`
- `/etc/init.d/sms_arpconfig`

詳細については、Sun BluePrints OnLine 掲載記事『Securing the Sun Fire 12K and 15K System Controller』と『Securing the Sun Fire 12K and 15K Domains』を参照してください。

s15k-exclude-domains.fin

このスクリプトは、SC と 1 つまたは複数のドメイン間の TCP/IP 接続を無効にします。詳細については、Sun BluePrints OnLine 掲載記事『Securing the Sun Fire 12K and 15K System Controller』を参照してください。

s15k-sms-secure-failover.fin

注 – このスクリプトは Sun Fire ハイエンドシステムの SC にのみ使用します。ほかのシステムでは実行されません。

このスクリプトは、フェイルオーバーデーモン `fomd` による Secure Shell の使用を自動的に有効にします。このスクリプトは、Secure Shell 構成の多くを自動的に行うだけでなく、従来の `r*` サービスを無効にします。

詳細については、Sun BluePrints OnLine 掲載記事『Securing the Sun Fire 12K and 15K System Controller』を参照してください。

第6章

監査スクリプト

この章では、監査スクリプトの使用、追加、変更、および削除について説明します。監査スクリプトは、システムのセキュリティー状態を定期的にチェックするための簡単な方法です。システムのセキュリティーがセキュリティープロファイルに適合していることを確認するために、定期的にシステムをチェックしてください。

標準の監査スクリプトは、終了スクリプトによりシステムが変更されたことを確認し、強化処理後に発生した不一致があれば報告します。監査スクリプトには対応する終了スクリプトと同じ名前を使用しますが、接尾辞は異なります。終了スクリプトの接尾辞は `.fin` ですが、監査スクリプトには `.aud` という接尾辞を使用します。

この章では、以下の項目を説明します。

監査スクリプトのカスタマイズ

この節では、既存の監査スクリプトのカスタマイズと、新しい監査スクリプトの作成についての手順と推奨事項を説明します。また、監査スクリプト機能を使用する際のガイドラインについても説明します。

標準の監査スクリプトをカスタマイズする

監査スクリプトは、Solaris Security Toolkit ドライバおよび終了スクリプトと同じように、カスタマイズすることができます。

注意 – Solaris Security Toolkit ソフトウェアで提供されているスクリプトを変更するときは、注意が必要です。直接元のスクリプトを変更せずに、必ずスクリプトのコピーを作成して、それを変更するようにしてください。直接元のスクリプトを変更した場合、Solaris Security Toolkit ソフトウェアをアップグレードまたは削除するときに、機能が失われる可能性があります。

できる限りコードの変更は必要最低限にとどめ、その変更について記録に残しておいてください。

監査スクリプトをカスタマイズするには、環境変数を使用します。大部分のスクリプトの動作は環境変数を使用して大幅に変更できるため、直接スクリプトコードを変更する必要はありません。これが不可能な場合は、`user.run` スクリプトで使用する、カスタマイズ版のスクリプトを開発することで、関数を変更しなければならない場合があります。すべての環境変数のリストと、環境変数を定義する際のガイドラインについては、第 7 章を参照してください。

注意 – 標準の終了スクリプトをカスタマイズしたり、新しい終了スクリプトを作成したときは、必ず関連する監査スクリプトにも同様の変更を行なってください。

注 – 変更をより多くのユーザーのために役立てたいときは、拡張機能に関するバグレポートや要望を提出することをご検討ください。Solaris Security Toolkit 開発チームは、ユーザーに役立つソフトウェアの改善方法を常に求めております。

▼ 監査スクリプトをカスタマイズするには

使用しているシステムおよび環境に合わせて標準の監査スクリプトをカスタマイズするには、次の手順を実行します。元のファイルが更新されたときに、カスタマイズしたファイルが更新されたファイルで上書きされないようにするには、この手順を使用してください。`pkgrm` コマンドを使用して Solaris Security Toolkit ソフトウェアを削除しても、カスタマイズしたファイルは削除されません。

1. **カスタマイズする監査スクリプトとその関連ファイルをコピーします。**

   監査スクリプトとその関連ファイルについては、『Solaris Security Toolkit 4.2 管理マニュアル』の第 6 章を参照してください。

2. **コピーしたファイルを、カスタムスクリプトおよびカスタムファイルとして識別される名前に変更します。**

   命名規則については、『Solaris Security Toolkit 4.2 管理マニュアル』の第 1 章「ガイドライン」を参照してください。

3. カスタムスクリプトとファイルを変更します。

finish.init ファイルは、監査スクリプトの構成変数をすべて提供するファイルです。変数とその正しい値を user.init ファイルに追加することで、finish.init ファイルで指定されている変数のデフォルト値を無効にすることができます。このファイルには、各変数と、その変数の監査スクリプトでの影響と使用について、詳しい説明が記載されています。このファイルと、その環境変数の変更についての詳細は、第 3 章を参照してください。変更をすべてのドライバに適用するのではなく、変更を特定のドライバに限定したい場合は、そのドライバを変更します。

監査スクリプトをカスタマイズする場合、監査機能を正確なものにするためには、カスタマイズする機能に終了スクリプトと監査スクリプトのどちらもアクセスできることが重要となります。これは、他の init ファイルを変更したり、スクリプトを直接変更したりするのではなく、user.init スクリプトの環境変数を変更することで、最も簡単かつ効果的に実現できます。

コード例 6-1 では、ソフトウェアパッケージがインストールされて構成され、システム再起動時に実行されるように設定されていることをチェックすることで、install-openssh.audit スクリプトが正しいソフトウェアインストールをどのように検証するかを示しています。この例では、ソフトウェアパッケージがインストールされて構成され、システム再起動時に実行されるように設定されていることをチェックしています。

コード例 6-1 install-openssh.aud スクリプト例

```
#
#!/bin/sh
# Copyright (c) 2005 by Sun Microsystems, Inc.
# All rights reserved.
#
#ident  "@(#)install-openssh.aud        1.3     07/12/05 SMI"
#
# ****************************************************************
# Service definition section.
# ****************************************************************
#---------------------------------------------------------------
service="OpenSSH"
servfil="install-openssh.aud"
servhdr_txt="
#Rationale for Verification Check:
#This script will attempt to determine if the OpenSSH software is
#installed, configured and running on the system. Note that this
#script expects the OpenSSH software to be installed in package
#form in accordance with the install-openssh.fin Finish script.

#Determination of Compliance:

#It indicates a failure if the OpenSSH package is not installed,
#configured, or running on the system.
"
```

コード例 6-1　　install-openssh.aud スクリプト例 (続き)

```
#

#-----------------------------------------------------------------

servpkg="
   OBSDssh
"

#-----------------------------------------------------------------

servsrc="
   ${JASS_ROOT_DIR}/etc/rc3.d/S25openssh.server
"

#-----------------------------------------------------------------

servcfg="
   ${JASS_ROOT_DIR}/etc/sshd_config
"

#-----------------------------------------------------------------

servcmd="
   /opt/OBSDssh/sbin/sshd
"

#
****************************************************************
# Check processing section.
#
****************************************************************

start_audit "${servfil}" "${service}" "${servhdr_txt}"

logMessage "${JASS_MSG_SOFTWARE_INSTALLED}"

if check_packageExists "${servpkg}" 1 LOG ; then
 pkgName="`pkgparam -R ${JASS_ROOT_DIR} ${servpkg} NAME`"
 pkgVersion="`pkgparam -R ${JASS_ROOT_DIR} ${servpkg} VERSION`"
 pkgBaseDir="`pkgparam -R ${JASS_ROOT_DIR} ${servpkg} BASEDIR`"
 pkgContact="`pkgparam -R ${JASS_ROOT_DIR} ${servpkg} EMAIL`"

 logNotice "Package has description '${pkgName}'"
 logNotice "Package has version '${pkgVersion}'"
 logNotice "Package has base directory '${pkgBaseDir}'"
 logNotice "Package has contact '${pkgContact}'"
```

コード例 6-1　　install-openssh.aud スクリプト例 (続き)

```
#
 logMessage "\n${JASS_MSG_SOFTWARE_CONFIGURED}"
 check_startScriptExists "${servsrc}" 1 LOG
 check_serviceConfigExists "${servcfg}" 1 LOG

 logMessage "\n${JASS_MSG_SOFTWARE_RUNNING}"
 check_processExists "${servcmd}" 1 LOG
fi

finish_audit
```

新しい監査スクリプトを作成する

新しい監査スクリプトを作成して、使用している Solaris Security Toolkit ソフトウェアに統合することができます。通常、監査スクリプトは Bourne シェルや Solaris 10 OS の Perl で作成されるため、新機能を追加するのは比較的簡単です。UNIX シェルスクリプトの作成経験があまりない開発者は、同様の機能を実行する既存の監査スクリプトを調べて、目的のタスクを実行する方法を習得し、アクションの正しいシーケンスを理解してください。

終了スクリプトの新規作成と同じ規則が、監査スクリプトの新規作成にも適用されます。これらの規則については、131 ページの「終了スクリプトのカスタマイズ」を参照してください。

注 – 監査スクリプトと終了スクリプトは連携しています。新しい監査スクリプトを追加するときは、対応する終了スクリプトも必ず追加してください。

標準の監査スクリプトの使用

監査スクリプトは、Solaris Security Toolkit ソフトウェア内で、セキュリティー状態とあらかじめ定義されているセキュリティープロファイルを比較することにより、自動的にセキュリティー状態を検証する方法を提供します。セキュリティー変更が正しく行われたことを検証し、システムのセキュリティー状態と設定したセキュリティープロファイルの間に不一致がある場合に報告を受け取るときには、監査スクリプトを使用します。監査スクリプトを使用してシステムのセキュリティーを検証する方法についての詳細は、『Solaris Security Toolkit 4.2 管理マニュアル』の第 6 章を参照してください。

この節では、Audit ディレクトリに格納されている標準の監査スクリプトについて説明します。ここで説明するのは、監査スクリプトで実行される機能だけです。

Audit ディレクトリ内のスクリプトは、以下のカテゴリに分かれています。これは、Finish ディレクトリの終了スクリプトのカテゴリと同じです。

- 無効化 (disable)
- 有効化 (enable)
- インストール (install)
- 最小化 (minimize)
- 印刷 (print)
- 削除 (remove)
- 設定 (set)
- 更新 (update)

これらの標準の監査スクリプトに加え、Solaris Security Toolkit ソフトウェアでは、製品固有の監査スクリプトも提供しています。製品固有の監査スクリプトのリストについては、224 ページの「製品固有の監査スクリプトの使用」を参照してください。

無効化 (disable) 監査スクリプト

この節では、以下の無効化 (disable) 監査スクリプトについて説明します。

- 198 ページの「disable-picld.aud」
- 198 ページの「disable-power-mgmt.aud」
- 198 ページの「disable-ppp.aud」
- 198 ページの「disable-preserve.aud」
- 199 ページの「disable-remote-root-login.aud」
- 199 ページの「disable-rhosts.aud」
- 199 ページの「disable-routing.aud」
- 199 ページの「disable-rpc.aud」
- 200 ページの「disable-samba.aud」
- 200 ページの「disable-sendmail.aud」
- 201 ページの「disable-slp.aud」
- 201 ページの「disable-sma.aud」
- 201 ページの「disable-snmp.aud」
- 202 ページの「disable-spc.aud」
- 202 ページの「disable-ssh-root-login.aud」
- 202 ページの「disable-syslogd-listen.aud」
- 202 ページの「disable-system-accounts.aud」
- 203 ページの「disable-uucp.aud」
- 203 ページの「disable-vold.aud」
- 203 ページの「disable-wbem.aud」
- 204 ページの「disable-xfs.aud」
- 204 ページの「disable-xserver.listen.aud」

disable-ab2.aud

注 – Solaris OS バージョン 9 および 10 では AnswerBook2 ソフトウェアは使用されなくなったため、このスクリプトは、Solaris OS バージョン 2.5.1 ～ 8 を実行しているシステムでのみ使用します。

このスクリプトは、AnswerBook2 サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-apache.aud

注 – このスクリプトは、サンによってパッケージ化され、Solaris OS バージョン 8 または 9 に組み込まれている Apache Web Server についてのみチェックを行います。

このスクリプトは、Apache Web Server がシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-apache2.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、Apache 2 サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-appserv.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、Sun Java Application Server がシステムにインストールされ、構成され、実行されているか判定します。このソフトウェアが、インストールされているか、動作するよう構成されている場合は、失敗が表示されます。

disable-asppp.aud

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 8 を実行しているシステムでのみ使用します。Solaris 9 および 10 OS では、このサービスは PPP サービスに置き換わっており、検証は `disable-ppp.aud` スクリプトを使用して行われます。

このスクリプトは、ASPPP サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-autoinst.aud

このスクリプトは、自動インストール機能がシステムにインストールされ、有効になっているか判定します。このソフトウェアが、インストールされているか、動作するよう構成されている場合は失敗が表示されます。

disable-automount.aud

注 – 自動マウントサービスが必要な場合には、このスクリプトを使用しないでください。また、このサービスは RPC サービスに依存しているため、`disable-rpc.fin` スクリプトも使用しないでください。

このスクリプトは、自動マウントサービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-dhcpd.aud

注 – このスクリプトは、Solaris OS バージョン 8 ～ 10 の DHCP サーバーにのみ使用します。

このスクリプトは、DHCP サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-directory.aud

注 – このスクリプトは、Solaris 9 または 10 OS に付属している Sun Java System Directory Server に対してのみチェックを行います。別パッケージの製品や、Solaris OS の他のバージョンに付属している Sun Java System Directory Server の監査は行いません。

このスクリプトは、Sun Java System Directory サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-dmi.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、DMI サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-dtlogin.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、CDE ログインサーバー (`dtlogin`) がシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-face-log.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、`/usr/oasys/tmp/TERRLOG` ファイルが存在し、グループおよびその他のユーザーに対する書き込み権がないことを検証します。ファイルが グループおよびその他のユーザーによるグローバル書き込み権を持っている場合、失敗が表示されます。

disable-IIim.aud

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、`IIim` サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実際に実行されている場合は、失敗が表示されます。

disable-ipv6.aud

注 – このスクリプトは、Solaris OS バージョン 8、9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、IPv6 インタフェースを plumb させる IPv6 ホスト名ファイル /etc/hostname6.* が存在するかどうかをチェックします。また、このスクリプトでは、in.ndpd サービスが開始されているかどうかについてもチェックします。IPv6 インタフェースが構成されて plumb されている、またはサービスが実行されている場合は、失敗が表示されます。

disable-kdc.aud

注意 – Solaris 9 OS では、JASS_DISABLE_MODE が conf に設定されている場合は、kdc.conf ファイルが無効になるため、システムに Kerberos クライアントと KDC サーバー両方としての機能があるかどうかが判定されます。システムが Kerberos クライアントとして機能しなければならない場合は、このスクリプトをこのような方法では使用しないでください。

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、KDC サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-keyboard-abort.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

注 – 一部のシステムには、キースイッチが安全位置に設定できるものがあります。これらのシステムでは、キースイッチを安全位置に設定すると、kdb コマンドによるソフトウェアのデフォルト設定が無効になります。

このスクリプトは、キーボードのアボートシーケンスを無視するようにシステムが構成されているかどうか判定します。通常、キーボードのアボートシーケンスが開始されると、オペレーティングシステムが中断されて、コンソールは OpenBoot PROM モニターまたはデバッガに入ります。このスクリプトでは、システムがこの方法で中断されるかどうか判定します。

disable-keyserv-uid-nobody.aud

このスクリプトは、ユーザー nobody がデフォルトのキーを使用できないように keyserv サービスが構成されているかどうか判定します。keyserv プロセスが -d フラグ付きで実行されておらず、ENABLE_NOBODY_KEYS パラメータが NO に設定されていない場合に、失敗が表示されます (Solaris OS バージョン 9 および 10 の場合)。

disable-ldap-client.aud

注 – このスクリプトは、Solaris OS バージョン 8 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、LDAP クライアントサービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-lp.aud

このスクリプトは、ラインプリンタ (lp) サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。また、このスクリプトは、lp ユーザーが cron 機能の使用を許可されているか、または crontab ファイルがインストールされている場合は、失敗を表示します。

disable-mipagent.aud

注 – このスクリプトは、Solaris OS バージョン 8 ～ 10 にのみ使用します。

このスクリプトは、モバイル IP サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-named.aud

注 – このサービスを無効にしても、システムがドメインネームシステム (DNS) クライアントとして動作する機能には影響しません。

このスクリプトは、DNS サーバーがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、(構成ファイルにより) 動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

このスクリプトは、Sun Microsystems によってパッケージ化され、Solaris OS に組み込まれている DNS サーバーについてのみチェックを行います。

disable-nfs-client.aud

注意 – NFS クライアントサービスが必要な場合には、このスクリプトを使用しないでください。また、このサービスは RPC サービスに依存しているため、`disable-rpc.fin` スクリプトも使用しないでください。

このスクリプトは、NFS クライアントサービスがシステムで構成されて、実行されているか判定します。ソフトウェアがシステムで動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-nfs-server.aud

注意 – NFS サービスが必要な場合には、このスクリプトを使用しないでください。また、このサービスは RPC サービスに依存しているため、`disable-rpc.fin` スクリプトも使用しないでください。

このスクリプトは、NFS サービスがシステムで構成されて、実行されているか判定します。ソフトウェアがシステムで動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-nscd-caching.aud

このスクリプトは、passwd、group、host、または ipnodes サービスのいずれかで、値が 0 に設定されていない、正または負の数の生存時間があるかどうか判定します。値が 0 でない場合は、失敗が表示されます。

disable-picld.aud

注 – このスクリプトは、Solaris OS バージョン 8 および 9 を実行しているシステムでのみ使用します。

このスクリプトは、PICL サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-power-mgmt.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、電源管理サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-ppp.aud

注 – このサービスは Solaris 8 OS (7/01) で導入され、以前の ASPPP サービスを補完するものです。このスクリプトは、Solaris OS バージョン 8 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、PPP サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-preserve.aud

このスクリプトは、保持機能が有効になっているかどうか判定します。有効である場合は、失敗が表示されます。

disable-remote-root-login.aud

注 – Solaris Secure Shell の使用など、/bin/login を使用しないでシステムにアクセスする他のメカニズムでは、システムがこのテストにパスした場合でも、直接 root にアクセスできます。

このスクリプトは、root ユーザーが直接ログインを許されているのか、または telnet などの /bin/login を使用するプログラムにより遠隔でシステムに対してコマンドを実行している場合、それが判定され、失敗が表示されます。

disable-rhosts.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、rhosts と hosts.equiv の機能が、/etc/pam.conf の PAM 構成によって有効になっているかどうか判定します。この機能が /etc/pam.conf ファイルの pam_rhosts_auth.so.1 モジュールを使用して有効になっている場合は、失敗が表示されます。

disable-routing.aud

注 – このスクリプトは、Solaris OS バージョン 5.51 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、あるネットワークから別のネットワークへのネットワークパケットのルーティングまたはパケット転送が無効であるかどうかを判断します。

disable-rpc.aud

注意 – システムで、自動マウント、NFS、NIS、NIS+、CDE、およびボリューム管理 (Solaris 9 および 10 OS のみ) のいずれかのサービスを使用している場合は、RPC ポートマッパー機能を無効にしないでください。

このスクリプトは、RPC サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。また、各サービスが rpcbind ポートマッパーを使用して登録されている場合にも、このスクリプトは失敗が表示されます。

disable-samba.aud

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、Samba サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。チェック時に無効と判断されるのは、Solaris OS に付属の Samba サービスだけです。システムにインストールされているその他の Samba ソフトウェアには影響を与えません。

disable-sendmail.aud

注 – Solaris Security Toolkit ソフトウェアの変更では、Solaris OS システムが電子メールを受信するよう構成されていないことのみ検証されます。送信する電子メールは正常に処理されます。

デフォルトでは、sendmail サービスは、ローカルメールの転送と、遠隔発信元からの着信メールの受信の両方を行うように構成されています。システムをメールサーバーにしない場合は、着信メッセージを受け取らないように sendmail サービスを構成することができます。このスクリプトでは、sendmail サービスが着信メッセージを受け取らないように構成されているかどうかチェックします。

使用している Solaris OS のバージョンによって、このチェックを実行する方法が異なります。

- Solaris OS バージョン 9 および 10 では、/etc/mail/sendmail.cf ファイルに次のコードが含まれているかチェックします。

```
Name=NoMTA4, Family=inet, Addr=127.0.0.1
```

- Solaris OS バージョン 8 では、/etc/default/sendmail ファイルの MODE パラメータが "" (なし) に設定されているかどうかチェックします。

- それより前の Solaris OS バージョンでは、sendmail 実行コントロールスクリプトが無効になっており、root ユーザーの crontab ファイルに、キューに入っているメールの処理を自動化するエントリが追加されているかどうか判定します。

各 Solaris OS バージョンに対応したチェックの結果、sendmail サービスが無効でない場合に、失敗が表示されます。

disable-slp.aud

注 – このスクリプトは、Solaris OS バージョン 8、9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、SLP サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-sma.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、SMA サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステム上で呼び出される、動作するよう構成されている、あるいは実際に実行されている場合は、失敗が表示されます。

disable-snmp.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 に搭載されている SNMP エージェントのみチェックします。

このスクリプトは、SNMP サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。このスクリプトでは、Sun 以外の SNMP エージェントがシステムで機能しているかどうかを検証しません。

disable-spc.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、SPC サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-ssh-root-login.aud

注 – このスクリプトは、Solaris 9 または 10 OS を実行し、Solaris Secure Shell パッケージがインストールされて有効になっているシステムでのみ使用します。

このスクリプトでは、Solaris OS バージョン 9 および 10 に搭載されている Solaris Secure Shell サービスが `root` アカウントへの遠隔アクセスを制限していない場合に、失敗が表示されます。

disable-syslogd-listen.aud

注 – `SYSLOG` サーバーの機能は遠隔で生成された `SYSLOG` ログメッセージを受け取ることであるため、このスクリプトは `SYSLOG` サーバーでは使用しないでください。このスクリプトは、Solaris OS バージョン 8 ～ 10 を実行しているシステムでのみ使用します。

スクリプトでは、`syslogd` プロセスの遠隔ログ機能を許可しないオプションを設定します。このスクリプトは、`SYSLOG` サービスが遠隔ログ接続を受け付けるように構成されているかどうか判定します。`syslogd` プロセスが `-t` フラグ付きで実行されていない (Solaris OS 8)、および `LOG_FROM_REMOTE` パラメータが `NO` に設定されていない (Solaris OS バージョン 9 および 10) 場合に、失敗が表示されます。

disable-system-accounts.aud

このスクリプトは、`JASS_ACCT_DISABLE` 環境変数にリストされている各アカウント名を調べ、`JASS_SHELL_DISABLE` 変数で定義されているシェルを使用するように構成されていないアカウントに対して失敗が表示されます。また、`JASS_SHELL_DISABLE` 変数にリストされているシェルプログラムがシステム上に存在しない場合も、失敗が表示されます。

注 – このスクリプトでチェックするのは、/etc/passwd ファイルにリストされているアカウントだけです。他のネームサービス (NIS、NIS+、または LDAP) にリストされているアカウントはチェックしません。

disable-uucp.aud

このスクリプトは、UUCP サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。また、nuucp ユーザーが存在する (Solaris 9 OS およびそれ以前) またはこのユーザーがロックされてない (Solaris 10) 場合、in.uucpd が /etc/inetd.conf に存在する場合、あるいは uucp crontab ファイルがインストールされている場合も失敗が表示されます。

disable-vold.aud

注 – システムで、リムーバブルメディア (フロッピーディスク、CD-ROM など) の自動取り付けおよび取り外しを行う必要がある場合は、このスクリプトを使用しないでください。

このスクリプトは、VOLD サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-wbem.aud

注 – WBEM サービスが必要な場合には、このスクリプトを使用しないでください。また、このサービスは RPC サービスに依存しているため、disable-rpc.fin スクリプトも使用しないでください。Solaris Management Console を使用する必要がある場合は、このスクリプトを使用しないでください。このスクリプトは、Solaris OS バージョン 8 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、WBEM サービスがシステムにインストールされ、構成され、実行されているか判定します。ソフトウェアがシステムにインストールされている、動作するよう構成されている、あるいは実行されている場合は、失敗が表示されます。

disable-xfs.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、xfs サービスがシステムにインストールされ、有効にされ、実行されているか判定します。ソフトウェアがシステム上で、動作するよう有効である、あるいは実際に実行されている場合は、失敗が表示されます。

disable-xserver.listen.aud

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

X11 サーバーが TCP トランスポートを使用してクライアント接続を受け付けるように構成されている場合に、失敗が表示されます。また、TCP トランスポートの使用を許可する構成で X11 サーバーが実行されている場合にも、失敗が表示されます。

有効化 (enable) 監査スクリプト

この節では、以下の有効化 (enable) 監査スクリプトについて説明します。

enable-account-lockout.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、LOCK_AFTER_RETRIES の値が /etc/security/policy.conf ファイルで正しく定義されているかを検証します。また、/etc/user_attr で指定されている LOCK_AFTER_RETRIES 以外の値を持つユーザーが存在しないことを確認します。

enable-bart.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは BART が実行されていることを検証し、BART の規則とマニフェストファイルを比較します。

BART 規則ファイルが存在するかどうか、また存在する場合はその構成が実行中のドライバおよびその BART 規則ファイルと矛盾がないかどうかを判断します。BART 規則ファイルの構成が実行中のドライバおよびその BART 規則ファイルと矛盾する場合、スクリプトは $JASS_FILES/var/opt/SUNWjass/bart/rules から規則ファイルをコピーします。またこのスクリプトは、/var/opt/SUNWjass/BART/manifests に JASS_TIMESTAMP.txt という名前の新しいマニフェスト (20050711152248.txt など) も作成します。

さらにこのスクリプトは、新規作成されたマニフェストファイルと最新のマニフェストファイルとの間の違いを報告し、使用される BART マニフェストの名前を含む監査メッセージを生成し、検出された問題について以前のマニフェストファイルまたはフィンガープリントデータベースをチェックすることをユーザーに勧めます。

注 – enable-bart.aud スクリプトにより報告されるエラーは必ずしもアラームの原因にはなりません。追加、削除、変更されたファイルまたはファイルアクセス権など、スクリプトがチェックするディレクトリで変更が検出されると、必ずエラーが報告されます。ただし、enable-bart.aud スクリプトにより作成される出力で潜在的な問題がないかを確認する必要はありません。

enable-bsm.aud

注 – このスクリプトは、Solaris OS バージョン 8 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、SunSHIELD Solaris 基本セキュリティーモジュール (Solaris BSM) 監査機能が有効で、システム上で実行されているか、このサービスが /etc/system ファイルに読み込まれているか、および audit_warn 別名が /etc/mail/aliases で定義されているか判定します。これらのうちで 1 つ以上のチェックに失敗した場合は、失敗が表示されます。

enable-coreadm.aud

注 – このスクリプトは、Solaris OS バージョン 7 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、JASS_CORE_DIR で指定されているディレクトリに、生成されたコアファイルが格納されているか確認します。Solaris OS バージョン 7 ～ 9 に搭載されている coreadm 機能が構成されていない場合に、失敗が表示されます。また、JASS_CORE_PATTERN で指定されたタグがコアファイルに付けられていない場合も、エラー状態になります。

enable-ftp-syslog.aud

このスクリプトは、FTPサービスが、セッションと接続情報を記録するように構成されているかどうか判定します。FTPサービスの記録が有効になっていない場合は、失敗が表示されます。

enable-ftpaccess.aud

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、FTP サービスが /etc/ftpd/ftpaccess ファイルを使用するように構成されているかどうか判定します。FTP サービスが正しく構成されていない場合は、失敗が表示されます。

enable-inetd-syslog.aud

このスクリプトは、インターネットサービスデーモン (inetd) サービスがセッションと接続情報を記録するように構成されているかどうか判定します。

- **Solaris OS** バージョン **9** では、-t オプションが inetd コマンド行に追加されているか、/etc/default/inetd ファイルの ENABLE_CONNECTION_LOGGING 変数が YES に設定されているかチェックします。いずれかのチェックに失敗した場合は、失敗が表示されます。
- **Solaris 10 OS** では、このスクリプトは defaults/tcp-trace プロパティーが FMRI svc:/network/inetd に対して定義されているかどうかをチェックします。またこのスクリプトは、-t オプションが指定されている実行中の inetd プロセスもチェックします。

enable-ipfilter.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、すべての使用可能なネットワークインタフェースの ipfilter 構成を確認し、正しい IP フィルタ規則セットがインストールされていることを検証します。スクリプトは次の動作を行います。

- /etc/ipf/pfil.ap を解析し、ネットワークインタフェースがコメントアウトされているかどうかを判断する。一部のネットワークインタフェースがコメントアウトされている場合、スクリプトはセキュリティーポリシー違反のメッセージを生成します。
- システム上の既存の /etc/ip/ipf.conf ファイルを確認し、それがキーワード固有のドライバと同じであるかどうかを確認する。異なる場合、スクリプトはセキュリティーポリシー違反のメッセージを生成します。
- network/ipfilter サービスが有効であることを検証する。有効でない場合、スクリプトはセキュリティーポリシー違反のメッセージを生成します。

enable-password-history.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、システム上のパスワード履歴の正しい構成を検証します。/etc/default/passwd ファイルをチェックし、HISTORY 値が指定されているかどうかを判断します。

- HISTORY 値が /etc/default/passwd ファイルで指定されている場合、スクリプトはその値を JASS_PASS_HISTORY 環境変数内の値と照合し、その値が正しいかどうかを確認します。
- HISTORY 値が、JASS_PASS_HISTORY 環境変数で指定されている値とは異なる場合、スクリプトはその値を訂正します。

- HISTORY 値が正しく設定されていない場合、スクリプトはその値を訂正し、監査セキュリティー違反を出力します。

enable-priv-nfs-ports.aud

このスクリプトは、1024 未満の特権範囲のポートから発信されたクライアント通信のみを受け付けるように NFS サービスが構成されているかどうか判定します。NFS サービスが正しく構成されていない場合は、失敗が表示されます。

enable-process-accounting.aud

このスクリプトは、プロセスアカウンティングソフトウェアがシステムにインストールされているか、有効になっているか、あるいは実行されているか判定します。いずれにも該当しない場合は、失敗が表示されます。

enable-rfc1948.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、RFC 1948 を使用して TCP シーケンス番号を生成するようにシステムが構成されているかどうか判定します。保存されている構成と、実際の実行時設定の両方をチェックします。RFC 1948 準拠の TCP シーケンス番号生成を行うようにシステムが構成されていない場合は、失敗が表示されます。

enable-stack-protection.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、スタックの保護と例外のログを有効にするように、noexec_user_stack オプションと noexec_user_stack_log オプションが /etc/system ファイルで設定されているかどうか判定します。これらのオプションが有効になっていない場合は、失敗が報告されます。

enable-tcpwrappers.aud

注 – このスクリプトは、付属の TCP ラッパーパッケージを使用している Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、TCPラッパーが、Solaris Security Toolkit ソフトウェアに含まれている `hosts.allow|deny` テンプレートを使用してインストールまたは構成されているか、`ENABLE_TCPWRAPPERS` 変数を使用して有効になっているか判定します。システムで TCPラッパーが使用されていない場合は、失敗が報告されます。

Solaris 10 OS 専用:

また、このスクリプトは次の動作も行います。

- `inetd` が `tcp_wrappers` を使用していることを検証する
- `rpcbind` が `tcp_wrappers` を使用していることを検証する
- 関数 `check_fileContentsexist` を使用することで、関連するキーワード固有の `hosts.allow|deny` の内容を検証し、`$JASS_FILES` にあるキーワード固有のファイルをシステム上の `hosts.allow|deny` と比較して、内容が一致するかどうかを判定します。内容が一致しない場合は、スクリプトはエラーを記録します。

インストール (install) 監査スクリプト

この節では、以下のインストール (install) 監査スクリプトについて説明します。

install-at-allow.aud

このスクリプトは、JASS_AT_ALLOW 変数にリストされているユーザー名が、/etc/cron.d/at.allow ファイルに存在しているかどうか判定します。JASS_AT_ALLOW で定義されているユーザー名リストは、デフォルトでは空です。このチェックにパスするには、各ユーザー名が /etc/passwd ファイルと /etc/cron.d/at.allow ファイルの両方に存在している必要があります。さらに、ユーザー名は /etc/cron.d/at.deny ファイルに存在していてはなりません。どちらのファイルにもユーザー名がリストされていない場合は、失敗が表示されます。

install-fix-modes.aud

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 9 を実行しているシステムでのみ使用します。

このスクリプトは、Fix Modes プログラムがシステムにインストールされ、実行されていたか判定します。このソフトウェアがインストールされていない、あるいは実行されていなかった場合は、失敗が表示されます。さらにこのスクリプトは、デバッグモードで Fix Modes プログラムを使用して、その他のファイルシステムオブジェクトを変更する必要があるかについても判定します。

install-ftpusers.aud

このスクリプトは、JASS_FTPUSERS パラメータにリストされているユーザー名が ftpusers ファイルに存在しているかどうか判定します。

install-jass.aud

このスクリプトは、Solaris Security Toolkit (SUNWjass) パッケージがシステムにインストールされているかどうか判定します。このパッケージがインストールされていない場合は、失敗が報告されます。

install-loginlog.aud

このスクリプトは、/var/adm/loginlog ファイルが存在しているか、このファイルに対する適切な所有権とアクセス権があるかチェックします。このファイルが存在していない、アクセス権が無効である、あるいは所有者が root アカウントを持っていない場合は、失敗が報告されます。

install-md5.aud

このスクリプトは、MD5 ソフトウェアがシステムにインストールされているかどうか判定します。このソフトウェアがインストールされていない場合は、失敗が報告されます。

install-nddconfig.aud

このスクリプトは、Sun BluePrints OnLine 掲載記事『Solaris Operating Environment Network Settings for Security』で指定されている、Solaris Security Toolkit ソフトウェアに付属の `nddconfig` 実行コントロールスクリプトファイルが、対象システムにコピーされ、その設定が対象システム上でアクティブにされているか判定します。

このスクリプトは、オブジェクトごとに以下のチェックを行います。

1. ソースファイルとターゲットファイルのタイプ (通常ファイル、シンボリックリンク、またはディレクトリ) が一致していることを確認する
2. ソースファイルとターゲットファイルの内容が同じであることを確認する

このスクリプトでは、`nddconfig` スクリプトで定義されている設定が、実行中のシステムで実際に機能しているかについても確認します。このスクリプトは、特に、スクリプト名が変更されていた場合や、同様のことを実行するために他のスクリプトが使用される場合には、Solaris Security Toolkit の `nddconfig` スクリプトのコピーを使用してより正確な結果を報告します。

上記のチェックのいずれかが正しくなかった場合は、失敗が表示されます。

install-newaliases.aud

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 8 を実行しているシステムでのみ使用します。

このスクリプトは、`/usr/bin/newaliases` プログラムが存在するかどうかチェックします。このファイルが存在しないか、またはシンボリックリンクでない場合は、失敗が示されます。

install-openssh.aud

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 8 を実行しているシステムでのみ使用します。Solaris 9 および 10 OS には Secure Shell ソフトウェアが含まれているので、Solaris 9 および 10 OS をインストールする場合はこのスクリプトは使用しません。

このスクリプトは、このスクリプトで指定されている OpenSSH パッケージがインストールされ、構成されているか判定します。このパッケージがインストールされていない場合は、失敗が報告されます。

install-recommended-patches.aud

このスクリプトは、推奨およびセキュリティパッチクラスタファイルに記載されているパッチがシステムにインストールされているか判定します。パッチ情報は、テスト対象システムの Solaris OS バージョンに基づいて、JASS_HOME_DIR/Patches ディレクトリから収集されます。いずれかのパッチがインストールされていない場合は、失敗が表示されます。

インストールされているパッチのバージョンがパッチ順ファイルにリストされているバージョン以上である場合は、成功が表示されます。

install-sadmind-options.aud

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 9 を実行しているシステムでのみ使用します。

このスクリプトは、sadmind サービスが /etc/inet/inetd.conf ファイルに存在しているかどうか判定します。存在している場合は、JASS_SADMIND_OPTIONS 変数で定義されているとおりにオプションが設定されているかチェックします。デフォルトの設定は -S 2 です。

install-security-mode.aud

このスクリプトは、EEPROM セキュリティーモードの状態をチェックします。このモードが command または full でない場合は、警告を表示します。また、PROM の失敗したログインカウンタをチェックし、そのカウンタがゼロでない場合にも警告を表示します。

注 – install-security-mode.fin スクリプトではシステムのセキュリティーモードを変更できないため、スクリプトでは失敗を報告するのではなく、適合していないという警告だけを表示します。

install-shells.aud

このスクリプトは、JASS_SHELLS パラメータで定義されているシェルが、shells ファイルにリストされているか判定します。JASS_SHELLS で定義されているシェルを 表 6-1 に示します。

表 6-1 JASS_SHELLS で定義されているシェルのリスト

/usr/bin/sh	/usr/bin/csh
/usr/bin/ksh	/usr/bin/jsh
/bin/sh	/bin/csh
/bin/ksh	/bin/jsh
/sbin/sh	/sbin/jsh
/bin/bash	/bin/pfcsh
/bin/pfksh	/bin/pfsh
/bin/tcsh	/bin/zsh
/usr/bin/bash	/usr/bin/pfcsh
/usr/bin/pfksh	/usr/bin/pfsh
/usr/bin/tcsh	/usr/bin/zsh

JASS_SHELLS にリストされているシェルが shells ファイルにリストされていない場合は、失敗が表示されます。

install-strong-permissions.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでは使用しないでください。

このスクリプトは、install-strong-permissions.fin スクリプトで推奨されている変更が行われているか判定します。変更が行われていない場合は、失敗が表示されます。

Solaris 10 OS には多くのアクセス権と所有権の変更が組み込まれているため、このスクリプトは Solaris 10 OS には使用しません。このスクリプトを実行できないわけではありませんが、Solaris 10 OS への変更点を考慮すると、実行結果によりセキュリティーが改善されることはありません。

install-sulog.aud

このスクリプトは、/var/adm/sulog ファイルの適切な所有権とアクセス権についてチェックを行います。このファイルが存在していない、アクセス権が無効である、あるいは所有者が root アカウントを持っていない場合は、失敗が報告されます。

install-templates.aud

このスクリプトは、JASS_FILES 変数で定義されているファイルが、対象システムに正常にコピーされたか判定します。コピー元ファイルとコピー先ファイルのタイプ (通常ファイル、シンボリックリンク、またはディレクトリ) が一致しているか確認するテストと、両方のファイル内容が同一であるか確認するテストのいずれかに失敗した場合は、失敗が表示されます。

印刷 (print) 監査スクリプト

この節では、以下の印刷 (print) 監査スクリプトについて説明します。

- 214 ページの「print-jass-environment.aud」
- 215 ページの「print-jumpstart-environment.aud」
- 215 ページの「print-rhosts.aud」
- 215 ページの「print-sgid-files.aud」
- 215 ページの「print-suid-files.aud」
- 215 ページの「print-unowned-objects.aud」
- 215 ページの「print-world-writable-objects.aud」

これらのスクリプトは、監査用にカスタマイズされていることを除けば、印刷 (print) 終了スクリプトと機能は同じです。

print-jass-environment.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでは使用しないでください。

このスクリプトは、Solaris Security Toolkit で使用される変数とその内容を表示します。内容の検証やその他のチェックは行いません。

print-jumpstart-environment.aud

このスクリプトは、JumpStart モード専用です。JumpStart 環境変数の設定を印刷するときに使用します。このスクリプトは、監査チェックを行いません。

print-rhosts.aud

注 – スクリプトが必要とする追加の処理時間が許容される場合、print-rhosts.aud スクリプトは手動で有効にする必要があります。

このスクリプトは、.rhosts または hosts.equiv という名前を持つファイルに関する通知を表示します。さらに、詳細確認のためにファイルの内容も表示します。

print-sgid-files.aud

このスクリプトは、set-gid ビットセットを持つファイルに関する通知を表示するとともに、詳細確認のために完全な (長い) リストも表示します。

print-suid-files.aud

このスクリプトは、set-uid ビットセットを持つファイルに関する通知を表示するとともに、詳細確認のために完全な (長い) リストも表示します。

print-unowned-objects.aud

このスクリプトは、有効なユーザーとグループが割り当てられていないファイルに関する通知を表示するとともに、詳細確認のために完全な (長い) リストも表示します。

print-world-writable-objects.aud

このスクリプトは、world-writableである対応ファイルに関する通知を表示するとともに、詳細確認のために完全な (長い) リストも表示します。

削除 (remove) 監査スクリプト

この節では、以下の削除 (remove) 監査スクリプトについて説明します。

■ 216 ページの「remove-unneeded-accounts.aud」

remove-unneeded-accounts.aud

注 – このスクリプトは、Solaris OS バージョン 2.5.1 ～ 9 を実行しているシステムでのみ使用します。

remove-unneeded-accounts.aud スクリプトは、JASS_ACCT_REMOVE 変数で定義されている未使用の Solaris OS アカウントが、システムから削除されたことを検証します。

設定 (set) 監査スクリプト

この節では、以下の設定 (set) 監査スクリプトについて説明します。

■ 216 ページの「set-banner-dtlogin.aud」
■ 217 ページの「set-banner-ftpd.aud」
■ 217 ページの「set-banner-sendmail.aud」
■ 217 ページの「set-banner-sshd.aud」
■ 218 ページの「set-banner-telnet.aud」
■ 218 ページの「set-flexible-crypt.aud」
■ 218 ページの「set-ftpd-umask.aud」
■ 218 ページの「set-login-retries.aud」
■ 219 ページの「set-power-restrictions.aud」
■ 219 ページの「set-rmmount-nosuid.aud」
■ 219 ページの「set-root-group.aud」
■ 219 ページの「set-root-home-dir.aud」
■ 220 ページの「set-root-password.aud」
■ 220 ページの「set-strict-password-checks.aud」
■ 220 ページの「set-sys-suspend-restrictions.aud」
■ 220 ページの「set-system-umask.aud」
■ 221 ページの「set-term-type.aud」
■ 221 ページの「set-tmpfs-limit.aud」
■ 221 ページの「set-user-password-reqs.aud」
■ 221 ページの「set-user-umask.aud」

set-banner-dtlogin.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、CDE サービスまたは dtlogin サービス用のサービスバナーが定義されているか確認します。このスクリプトは、ファイルテンプレート JASS_ROOT_DIR/etc/dt/config/Xsession.d/0050.warning に /etc/motd ファイルをリストすると、システムがそのファイルの内容を表示するか確認します。

set-banner-ftpd.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、FTP サービスバナーが、JASS_BANNER_FTPD 変数で定義されている値と一致するかチェックします。サービスバナーが一致しない場合は、失敗が表示されます。変数の値は、Authorized Use Only です。

set-banner-sendmail.aud

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、JASS_BANNER_SENDMAIL 環境変数で定義されているとおりにサービスバナーを表示するよう sendmail サービスが構成されているか確認します。このバナーは、ネットワークを介して sendmail サービスに接続されているすべてのクライアントに表示されます。

set-banner-sshd.aud

注 – このスクリプトは、Solaris OS バージョン 9 および 10 を実行しているシステムでのみ使用します。

このスクリプトは、システムへのアクセス認証の前に、Secure Shell サービスがユーザーに /etc/issue の内容を表示することで、Secure Shell サービスバナーが表示されることを確認します。

set-banner-telnet.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、Telnet サービスバナーが、JASS_BANNER_TELNETD 変数で定義されている値と一致するかチェックします。サービスバナーが一致しなかった場合は、失敗が表示されます。変数の値は、Authorized Use Only です。

set-flexible-crypt.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、174 ページのコード例 5-7 で説明している各 Solaris Security Toolkit ドライバの変更が正しく行われていることをチェックすることで、強力なパスワードの使用を検証します。

このスクリプトによる監査時に Perl がシステムにインストールされている場合、Solaris Security Toolkit 4.2 ソフトウェアは Perl を使おうとします。システムに Perl が存在しない場合、スクリプトはエラーを出力します。

set-ftpd-umask.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、FTP サービスバナーが、JASS_FTPD_UMASK 変数で定義されている値と一致するかチェックします。ファイル生成マスク値が一致しない場合は、失敗が表示されます。変数の値は 022 です。

set-login-retries.aud

このスクリプトは、ログイン RETRIES パラメータに、JASS_LOGIN_RETRIES 変数で定義されている値が割り当てられているか判定します。変数のデフォルト値は 3 に設定されています。変数がデフォルト値に設定されていない場合は、失敗が表示されます。

set-power-restrictions.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 〜 10 を実行しているシステムでのみ使用します。

このスクリプトは、/etc/default/power ファイルの PMCHANGEPERM パラメータと CPRCHANGEPERM パラメータに値としてハイフン (-) が含まれているか確認します。含まれていない場合は失敗が表示されます。

set-rmmount-nosuid.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 〜 10 を実行しているシステムでのみ使用します。Solaris OS バージョン 8 〜 10 は、デフォルトで nosuid オプションを指定してリムーバブルメディアをマウントするように構成されています。このスクリプトは、デフォルトの設定に関係なく、必要なチェックを実行します。

このスクリプトは、nosuid パラメータを設定することによって、/etc/rmmount.conf ファイルで、リムーバブル Unix File System (UFS) または High Sierra File System (HSFS) ファイルシステムのマウントが制限されるかどうか判定します。/etc/rmmount.conf ファイルでこの制限が定義されていない場合は、失敗が表示されます。

set-root-group.aud

このスクリプトは、root アカウントの一次グループが、JASS_ROOT_GROUP 変数で定義されている値に設定されているかどうか判定します。正しく定義されていない場合は、失敗が表示されます。

set-root-home-dir.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、root アカウントが /etc/passwd ファイルに / のホームディレクトリを持っているかどうかを確認します。

- ホームディレクトリが / である場合は、スクリプトは監査エラーを出力します。
- ホームディレクトリが /root である場合、スクリプトは次のことをチェックします。
  - ディレクトリの所有権は root:root である

- ディレクトリのアクセス権は 0700 である
- ドット付きファイル (/.cshrc、/.profile、/llogin、/.ssh) は / から /root へ移動されている
- ドット付きファイルのアクセス権はすべて 0700 である

■ ホームディレクトリが / と /root のどちらでもない場合は、スクリプトは警告を生成しますが、監査エラーは生成しません。

set-root-password.aud

このスクリプトは、root アカウントのパスワードをチェックします。パスワードの値が JASS_ROOT_PASSWORD 変数の値と同じ場合は、失敗が表示されます。このチェックは、root パスワードを、JASS_ROOT_PASSWORD で定義されている値からできるだけ早く変更するようユーザーに促すために行われます。

set-strict-password-checks.aud

注 – このスクリプトは、Solaris 10 OS を実行しているシステムでのみ使用します。

このスクリプトは、さまざまなパスワードチェック用の正しい値が /etc/default/passwd ファイルで正しく定義されていることを確認します。

set-sys-suspend-restrictions.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、/etc/default/sys-suspend ファイルのチェックを行います。PERMS パラメータに値としてハイフン (-) が含まれていない場合は、失敗が表示されます。

set-system-umask.aud

このスクリプトは、システムのデフォルトのファイル生成マスクが、JASS_UMASK 変数で定義されている値に設定されているかどうか判定します。デフォルトの値は 022 に設定されています。変数が正しく定義されていない場合は、失敗が表示されます。

set-term-type.aud

このスクリプトは、/etc/profile ファイルと /etc/login ファイルでデフォルトの端末タイプが vt100 に設定されているかどうか判定します。デフォルトの端末タイプが正しく定義されていない場合は、失敗が表示されます。このスクリプトは便宜上の理由から用意されているだけで、失敗が表示されてもシステムのセキュリティーには影響ありません。

set-tmpfs-limit.aud

注 – Solaris 2.5.1 OS では、この set-tmpfs-limit.aud スクリプトはサポートされていないため実行できません。

このスクリプトは、/etc/vfstab ファイルで定義されている tmpfs ファイルシステムが、JASS_TMPFS_SIZE 変数にサイズが制限されているかどうか判定します。この変数は、デフォルトで 512 MB に設定されています。tmpfs ファイルシステムのサイズが JASS_TMPFS_SIZE 値に適合していない場合は、失敗が報告されます。

set-user-password-reqs.aud

このスクリプトは、システム上のパスワードポリシー設定が、あらかじめ定義されている設定どおりであるかどうかを確認します。設定値が、Solaris Security Toolkit で定義されている次のデフォルト値と一致しない場合は、エラーが表示されます。

- MINWEEKS - 1
- MAXWEEKS - 8
- WARNWEEKS - 1
- PASSLENGTH - 8

デフォルト値は、以下の環境変数に含まれます。

- JASS_AGING_MINWEEKS
- JASS_AGING_MAXWEEKS
- JASS_AGING_WARNWEEKS
- JASS_PASS_LENGTH

set-user-umask.aud

このスクリプトは、以下のファイルの umask パラメータが、(デフォルト値は 022 に設定されている) JASS_UMASK 変数で定義されている値に設定されているかどうか判定します。

- /etc/.login
- /etc/profile

- /etc/skel/local.cshrc
- /etc/skel/local.login
- /etc/skel/local.profile
- /etc/default/login

これらのファイルの umask パラメータが適切に設定されていない場合は、失敗が表示されます。

更新 (update) 監査スクリプト

この節では、以下の更新 (update) 監査スクリプトについて説明します。

update-at-deny.aud

このスクリプトは、JASS_AT_DENY 変数にリストされているユーザーアカウントが、/etc/cron.d/at.deny ファイルにリストされているかどうか判定します。JASS_AT_DENY 変数で定義されているユーザーアカウントのリストは、次のとおりです。

- root
- daemon
- bin
- sys
- adm
- lp
- uucp
- smmsp
- nobody
- noaccess

このチェックにパスするには、すべてのユーザーアカウントが、/etc/passwd ファイルと /etc/cron.d/at.deny ファイルの両方に存在している必要があります。ユーザーアカウントは /etc/cron.d/at.allow ファイルに格納しないでください。このファイルは優先度が高いので、設定を無効にすることがあるためです。いずれかのチェックに失敗した場合は、失敗が表示されます。

update-cron-allow.aud

このスクリプトは、JASS_CRON_ALLOW 変数にリストされているユーザーアカウントが、/etc/cron.d/cron.allow ファイルにあるかどうか判定します。デフォルトでは、ユーザーアカウントの値は root ユーザーだけです。このチェックに失敗すると、失敗が表示されます。

update-cron-deny.aud

このスクリプトは、JASS_CRON_DENY 変数にリストされているユーザーアカウントが、/etc/cron.d/cron.deny ファイルにあるかどうか判定します。JASS_CRON_DENY 変数で定義されているユーザーアカウントのリストは、次のとおりです。

- daemon
- bin
- sys
- adm
- lp
- uucp
- smmsp
- nobody
- noaccess

このチェックにパスするには、すべてのユーザーアカウントが、/etc/passwd ファイルと /etc/cron.d/cron.deny ファイルの両方に存在している必要があります。なお、ユーザーアカウントは /etc/cron.d/cron.allow ファイルに格納しないでください。このファイルは優先度が高いので、設定を無効にすることがあるためです。いずれかのチェックに失敗した場合は、失敗が表示されます。

update-cron-log-size.aud

注 – このスクリプトは、Solaris OS バージョン 2.6 ～ 10 を実行しているシステムでのみ使用します。

このスクリプトは、ログファイルのデフォルトのサイズ制限を増加するように cron 機能が構成されているかどうか判定します。チェック方法は、Solaris OS のバージョンと JASS_CRON_LOG_SIZE 変数の値によって決まります。JASS_CRON_LOG_SIZE 変数で定義されているサイズ制限は 20480 KB です。サイズ制限が正しくない場合は、失敗が表示されます。

update-inetd-conf.aud

このスクリプトは、JASS_SVCS_DISABLE 変数にリストされているサービスが /etc/inetd.conf ファイルで無効になっているかどうか判定します。また、JASS_SVCS_ENABLE 変数にリストされているサービスが、/etc/inetd.conf ファイルで有効になっているかどうかもチェックします。サービスが両方の変数にリストされている場合には、JASS_SVCS_ENABLE 変数で有効になっているサービスが残されます。いずれかのチェックに失敗した場合は、失敗が表示されます。

JASS_SVCS_DISABLE パラメータは、表 6-2 のように表示されます。

表 6-2　JASS_SVCS_DISABLEの出力例

100068	100083	100087	100134	100146	100147
100150	100155	100166	100221	100229	100230
100232	100234	100235	100242	100424	300326
536870916	chargen	comsat	daytime	discard	dtspc
echo	eklogin	exec	finger	fs	ftp
kerbd	klogin	kshell	login	name	netstat
printer	rexd	rquotad	rstatd	rusersd	rwalld
shell	smtp	sprayd	sun-dr	systat	talk
telnet	tftp	time	ufsd	uucp	uuidgen
walld	xaudio				

デフォルトでは、JASS_SVCS_ENABLE 変数は空です。suncluster3x-secure.driver などの一部のドライバがこの変数を使用することがあります。

製品固有の監査スクリプトの使用

特定のサン製品用の製品固有の監査スクリプトを、表 6-3 に示します。これらのスクリプトは、Audit ディレクトリに格納されています。

サンの新製品や更新されたサン製品のセキュリティーを強化するために、定期的に新しい監査スクリプトがリリースされます。最新のスクリプトのリストについては、以下の Security Web サイトを参照してください。

http://www.sun.com/security/jass

表 6-3　製品固有の監査スクリプト

製品	ドライバ名
Sun Cluster 3.x ソフトウェア	suncluster3x-set-nsswitch-conf.aud
Sun Fire ハイエンドシステムドメイン	s15k-static-arp.aud
Sun Fire ハイエンドシステムシステムコントローラ	s15k-static-arp.aud s15k-exclude-domains.aud s15k-sms-secure-failover.aud

suncluster3x-set-nsswitch-conf.aud

注 – このスクリプトは Sun Cluster 3.x システムにのみ適用され、ほかのシステムでは実行されません。

このスクリプトは、ホストのデータベースに対する最初のソースとして、cluster キーワードが /etc/nsswitch.conf ファイルにリストされているかどうか判定します。リストされていない場合は、失敗が表示されます。

詳細については、Sun BluePrints OnLine 掲載記事『Securing Sun Cluster 3.x Software』を参照してください。

s15k-static-arp.aud

System Management Services (SMS) バージョン 1.2 ～ 1.4.1 では、このスクリプトは、静的 ARP 構成ファイルが、Sun Fire ハイエンドシステムのシステムコントローラ (SC) とドメインにインストールされているか確認します。システムコントローラでは、このファイルは /etc/sms_sc_arp に格納されます。ドメインでは、このファイルは /etc/sms_domain_arp に格納されます。

このスクリプトは、すべての既存ドメインが、SC の静的 ARP 起動スクリプトと、それに対応するデータファイルにリストされているとおりに Ethernet アドレスを持っているかチェックします。

詳細については、Sun BluePrints OnLine 掲載記事『Securing the Sun Fire 12K and 15K System Controller』と『Securing the Sun Fire 12K and 15K Domains』を参照してください。

s15k-exclude-domains.aud

SMS バージョン 1.2 以降では、このスクリプトは、/etc/opt/SUNWSMS/SMS/config/MAN.cf ファイルが存在しているかどうか判定します。存在している場合は、このファイルにリストされているドメインがすべて I1 MAN から除外されているかチェックします。そして、すべてのドメインを I1 MAN から除外します。サイトでドメインの一部のみを除外するようにスクリプトを変更した場合は、ドメインが I1 MAN に含まれているという警告が表示されます。

詳細については、Sun BluePrints OnLine 掲載記事『Securing the Sun Fire 12K and 15K System Controller』を参照してください。

s15k-sms-secure-failover.aud

SMS バージョン 1.2 ～ 1.4.1 では、このスクリプトは、Sun BluePrints OnLine 掲載記事『Securing the Sun Fire 12K and 15K System Controller』の推奨事項に従って、Sun Fire ハイエンドシステムシステムコントローラが構成されているかどうか判定します。SMS_SVCS_DISABLE 変数にリストされているサービスのいずれかが、/etc/inet/inetd.conf で有効になっている場合は、失敗が表示されます。

第7章

環境変数

この章では、環境変数の使用について説明します。この章で取り上げる変数は、Solaris Security Toolkit ソフトウェアで使用されるすべての変数です。また、これらの変数値をカスタマイズする際のヒントとテクニックについても説明します。

この章では、以下の項目を説明します。

変数のカスタマイズと割り当て

Solaris Security Toolkit ソフトウェアには、ドライバとスクリプトの動作をカスタマイズまたは指示するための簡単な方法を提供する環境変数が用意されています。この環境変数は Bourne シェル変数なので、シェル変数に適用されるすべてのルールがこの Solaris Security Toolkit の変数にも適用されます。この節では、変数のカスタマイズと割り当てについて説明します。

Solaris Security Toolkit ソフトウェアには、次の 4 種類の環境変数があります。

- フレームワーク関数の変数
- 終了および監査スクリプト変数
- JumpStart モード変数
- ユーザー変数

注 – 上記の変数はすべて、割り当てとカスタマイズを行うことができます。

変数をカスタマイズするときは、それぞれのタイプの変数が Solaris Security Toolkit ソフトウェア内で果たす役割とその目的をあらかじめ理解しておくことが重要です。変数の設定とカスタマイズは、使用しているシステム、環境、およびセキュリティー

ポリシーに合うように Solaris Security Toolkit ソフトウェアを構成するための重要ポイントです。変数の使用についての詳細は、232 ページの「環境変数の使用」を参照してください。

場合によっては、標準の変数、ドライバ、スクリプトをカスタマイズしても、特定のニーズに対応できないことがあります。このような場合は、使用環境に合った変数、ドライバ、スクリプトを作成してください。変数の作成についての詳細は、231 ページの「環境変数の作成」を参照してください。

この節では、以下の項目を説明します。

静的変数の割り当て

静的変数とは、決まった値または静的な値が割り当てられる変数のことです。この値は Solaris Security Toolkit の初期化前にすでに設定されており、外的要因によってこの値が変更されない限り、Solaris Security Toolkit の実行中は同じ値のままです。実行されている状況や環境に応じて、これらの変数の値が変わることはありません。

静的変数は、システムの種類、ネットワーク設定、インストールされているアプリケーションなどの外部要因にポリシー設定が依存していない場合には便利です。たとえば、通常、パスワードの有効期限は、企業や部門のポリシーによって定義されます。この有効期限に静的変数を割り当てると、企業や部門内のすべてのシステムとデバイスにこの設定が適用されます。パスワードの有効期限は外部要因に依存しないため、通常、システム管理者はパスワードの有効期限を静的変数として設定します。

以下に静的変数の割り当て例を示します。

```
JASS_AGING_MAXWEEKS="8"
JASS_AGING_MINWEEKS="1"
```

この例では、ユーザーパスワードが最後に変更されてから 8 週間で期限が切れるように構成されています。また、同様に静的変数として定義されている 2 番目の変数では、ユーザーパスワードの変更を 1 週間に 1 回までと制限しています。

動的変数の割り当て

動的変数とは、一般に柔軟性を必要とする変数であり、その値がコマンドの出力結果やファイルの内容によって変わる変数のことです。この場合、動的変数は実行されている環境を認識して、より効果的に環境に適応することができます。以下に動的変数の割り当て例を示します。

```
JASS_AT_DENY="`awk -F: '{ print $1 }' ${JASS_PASSWD}`"
```

この例では、JASS_PASSWD (たとえば、JASS_ROOT_DIR/etc/passwd) ファイルで定義されている各ユーザーを、変数 JASS_AT_DENY に追加しています。ユーザーのリストは、Solaris Security Toolkit ソフトウェアが実行されているシステムによって変わります。このように、より使用環境に対応することができます。同様に、あらかじめ定義されている例外を除いたすべてのユーザーを含めるように構築することもできます。次の例では、root アカウントと ORACLE® アカウントを除き、システム上のすべてのユーザーを JASS_CRON_DENY 変数に追加する場合を示します。

```
JASS_CRON_DENY="`awk -F: '{ print $1 }' ${JASS_PASSWD} |\
   egrep -v '^root|^oracle'`"
```

複合置換変数の割り当て

さらに変数の割り当てについて考えていくと、複合置換という方法になります。この方法を使用すると、ポリシー、ファイルの内容、あるいはその他のメカニズムに基づいて、より高度な値を変数に割り当てることができます。

複合置換変数の割り当ての一例には、静的変数と動的変数の割り当ての組み合わせがあります。次の例では、静的リストの値と JASS_ROOT_DIR/etc/passwd ファイルの出力結果の両方に基づいた値を JASS_FTPUSERS に割り当てています。

```
JASS_FTPUSERS="`awk -F: '$1 !~ /^ftp/ { print $1 }' \
${JASS_PASSWD}` guest"
```

この例では、JASS_FTPUSERS 変数に guest アカウントが必ず追加されます。また、ログイン名が接頭辞 ftp から始まらないユーザーでも JASS_PASSWD にリストされていれば、JASS_FTPUSERS 変数に追加されます。これらの方法を組み合わせて使用すると、たいていの組織のニーズを満たすことができる構成のほとんどを実現することができます。

また別の高度な方法としては、シェルスクリプトやシェル関数に基づいて置換ポリシーを定義する方法もあります。この例については、Drivers/finish.init ファイルの JASS_SHELLS 変数の宣言を参照してください (コード例 7-1)。次の例では、OS のバージョンによって変数の割り当てが決まります。

コード例 7-1　　OS のバージョンに基づいた変数の割り当て

```
#
if [ -z "${JASS_SHELLS}" ]; then
# These shells are by default found in Solaris 2.5.1 to Solaris 7
JASS_SHELLS="
      /usr/bin/sh     /usr/bin/csh     /usr/bin/ksh
      /usr/bin/jsh    /bin/sh          /bin/csh
      /bin/ksh        /bin/jsh         /sbin/sh
      /sbin/jsh"
# This is to handle special cases by OS.
case ${JASS_OS_REVISION} in
      5.8 | 5.9)
         JASS_SHELLS="${JASS_SHELLS}
            /bin/bash       /bin/pfcsh       /bin/pfksh
            /bin/pfsh       /bin/tcsh        /bin/zsh
            /usr/bin/bash   /usr/bin/pfcsh   /usr/bin/pfksh
            /usr/bin/pfsh   /usr/bin/tcsh    /usr/bin/zsh"
      ;;
esac
fi
export JASS_SHELLS
# This function could be further enhanced, for example, to remove
# those shell entries that do not exist on the system. This
# could be done by adding the following code:
tmpShells="${JASS_SHELLS}"
JASS_SHELLS=""
for shell in ${tmpShells}; do
    if [ -x "${JASS_ROOT_DIR}${shell}" ]; then
        if [ -z "${JASS_SHELLS}" ]; then
            JASS_SHELLS="${JASS_SHELLS}/${shell}"
        fi
    fi
done
```

このような機能は、/usr/bin/bash や /usr/bin/tcsh など、一部のシェルが使用できない最小化されたシステムにおいて役立ちます。(なお、これらのシェルは、それぞれ SUNWbash パッケージと SUNWtcsh パッケージに存在するものです。) この方法を使用すれば、不適切な変数の割り当てが原因で生成される通知と警告のメッセージ数を減らすことができます。

グローバル変数およびプロファイルベース変数の割り当て

グローバル変数を割り当てると、Solaris Security Toolkit 変数の多くのデフォルト値を無効にすることができます。Solaris Security Toolkit ソフトウェアを実行するたびに、デフォルト値を無効にする変数を定義して割り当てるには、`user.init` ファイルをカスタマイズします。このファイルは、Solaris Security Toolkit ソフトウェアが起動されるたびに、`driver.init` プログラムによって読み取られます。

また、プロファイルベースの変数を割り当てて、デフォルト値を無効にすることもできます。この無効化は、`driver.init` ファイルを呼び出したあと、プロファイルそのものの内部で行われます。プロファイル内部で変数を割り当てると、すべてのプロファイルではなく特定のプロファイルについて、変数を更新、拡張、および優先指定することができます。たとえば、ファイル `server-secure.driver` に、次のプロファイルベースの変数の優先指定が含まれているとします。

```
JASS_SVCS_ENABLE="telnet ftp dtspc rstatd 100155"
```

この例では、Telnet、FTP、`dtspc`、`rstatd`、および `rpc.smserverd` (`100155`) サービスのエントリを含むように、`JASS_SVCS_ENABLE` 変数が割り当てられています。この割り当てでは、Solaris Security Toolkit ソフトウェアにこれらのサービスを有効にしておく (無効であった場合は有効にする) ように指示しています。通常、デフォルトの動作では、これらのサービスは `JASS_SVCS_DISABLE` 変数によって無効にされているサービスです。

環境変数の作成

標準の Solaris Security Toolkit ソフトウェアからは必要な変数が提供され、使用しているシステムと環境に合わせてその変数をカスタマイズすることができますが、ときには、新しい変数を作成する必要がある場合があります。変数の作成が必要となることが多いのは、独自のスクリプトを作成する場合です。サイト固有のスクリプトやカスタムスクリプトをサポートする新しい変数を作成して割り当てることができます。新しい変数を作成することで、Solaris Security Toolkit ソフトウェアが持っているフレームワークとモジュール性を生かすことができます。

迅速かつ容易に新しい機能を作成したり、カスタマイズした機能を実装するときは、Solaris Security Toolkit ソフトウェアの既存の機能を利用してください。標準の変数をサンプルとして使用し、そこから新しい変数を作成します。可能ならば、新しい変数を作成せずに、標準の変数をカスタマイズするようにしてください。このようにソフトウェアのフレームワークを使用すると、あまりカスタマイズされないコードを作成してサポートすることができます。

注 – 接頭辞 JASS_ は、Solaris Security Toolkit ソフトウェア開発者が使用するために予約されています。新しい変数を作成したときには、この接頭辞を使用しないでください。企業や組織に固有の接頭辞を使用するようにしてください。

移植性と構成上の問題を簡略化するために、各種 .init スクリプトで定義されている環境変数は Solaris Security Toolkit ソフトウェア全体を通して使用されます。

変数を追加する必要がある場合は、その変数を環境変数として user.init スクリプトに追加してください。

新しい変数を追加するには、デフォルト値を指定して変数宣言を追加し、その宣言を user.init ファイルにエクスポートします。この処理ではグローバルなデフォルト値がエクスポートされますが、セキュリティープロファイル (ドライバ) 内でこの値を無効にすれば、後で必要に応じて変更が可能です。たとえば、次のコードでは、デフォルト値 0 を指定して新しい変数 ABC_TESTING を user.init ファイルに追加しています。

```
ABC_TESTING="0"
export ABC_TESTING
```

変数の値が現在未定義である場合には、その値の設定だけを行う必要があるときがあります。この方法は、管理者にログインシェルから値を変更できるようにする場合に最も便利です。これを実行するには、上記のコーディング例を次のように変更します。

```
if [ -z "${ABC_TESTING}" ]; then
   ABC_TESTING="0"
fi
export ABC_TESTING
```

環境変数の使用

この節では、Solaris Security Toolkit ソフトウェアで定義されているすべての標準の変数をアルファベット順に説明します。これらの変数をより効果的に活用できる方法がある場合には、それに関する推奨事項などの役立つ情報も説明します。

Solaris Security Toolkit ソフトウェアには、次の 4 種類の環境変数があります。

- フレームワーク変数
- 終了および監査スクリプト変数
- JumpStart モード変数

■ ユーザー変数

この節で説明する各変数は、Solaris Security Toolkit ソフトウェアでの機能に応じて、以下のいずれかのファイルで定義されています。(すでに説明したように、機能はその目的に基づいて分類されています。)

■ `driver.init` (フレームワーク変数と JumpStart モード変数)
■ `finish.init` (終了および監査スクリプト変数)
■ `user.init` (ユーザー変数とグローバル優先変数)

これらのファイルについての詳細は、第 3 章を参照してください。

移植性と構成上の問題を簡略化するために、各種 `.init` スクリプトで定義されている環境変数は Solaris Security Toolkit ソフトウェア全体を通して使用されます。

変数を追加する必要がある場合は、その変数を環境変数として `user.init` スクリプトに追加してください。詳細については、231 ページの「環境変数の作成」を参照してください。

注 – スクリプトで使用される環境変数のデフォルト値は、`finish.init` スクリプトで定義されます。

この節では、変数について次の構成で説明します。

フレームワーク変数の定義

フレームワーク変数とは、Solaris Security Toolkit ソフトウェアで定義および使用される変数のことで、構成状態の維持またはコア変数の提供を行います。通常、これらの変数はグローバル変数であり、ソフトウェアのフレームワーク、コア関数、およびスクリプトの中に含まれます。

フレームワーク変数を変更することで、動的にソフトウェアの動作を変更できます。そこで、フレームワーク変数を変更するのは、どうしても必要な場合に限定してください。変更による影響を明確に理解し、問題が発生した場合に解決することができる経験豊富な管理者だけが変更を行うようにしてください。

注 – すべてのフレームワーク変数を変更できるわけではありません。配備された Solaris Security Toolkit ソフトウェア間での整合性を高め、それぞれのソフトウェア構成をサポート可能にするために、この制限が設けられています。

注意 – 変更しなければ無効にできないフレームワーク変数は、直接変更しないでください。

ここでは、以下のフレームワーク変数について説明します。

JASS_AUDIT_DIR

注 – 通常、この変数を変更する必要は**ありません**。

Solaris Security Toolkit ソフトウェアの規則では、監査スクリプトはすべて `Audit` ディレクトリに格納されます。しかし、格納先に柔軟性をもたせるため、管理者は、この `JASS_AUDIT_DIR` 環境変数を使用すれば、別の場所に監査スクリプトを格納することができます。デフォルトでは、この変数は `JASS_HOME_DIR/Audit` に設定されています。

JASS_CHECK_MINIMIZED

この変数は、監査処理でのみ使用されます。この変数の値によって、大部分の監査スクリプトに含まれている `check_minimized` 関数の実行方法が決定されます。この変数が `0` (デフォルト値) に設定されている場合、または値を持っていない場合、`check_minimized` 関数はチェックを実行しません。この変数の値が `1` の場合、スクリプトはチェックを実行します。

最小化されていないシステムで、チェックを行わずにソフトウェアを実行するときに、この変数を使用します。使用しない場合、最小化されていないシステムで `check_minimized` 関数から失敗メッセージが表示されて、監査処理には合格しません。

JASS_CONFIG_DIR

Solaris Security Toolkit ソフトウェアのバージョン 0.3 から、用途がわかりやすいように、変数 JASS_CONFIG_DIR は JASS_HOME_DIR に名前が変更されました。JASS_CONFIG_DIR 変数は、現在使用されていません。242 ページの「JASS_HOME_DIR」を参照してください。

JASS_DISABLE_MODE

注 – この環境変数は、Solaris 10 OS を実行しているシステムには使用しません。

この変数は、run-control スクリプトから開始されるサービスを無効にするために、Solaris Security Toolkit ソフトウェアで使用される方法を定義します。Solaris OS 9 では、この変数にはデフォルトで conf の値が割り当てられます。一方、それ以前のすべてのバージョンでは、デフォルトの値は script です。

注 – 特定のサービスで構成ファイルを使用していなかったり、開始前に構成ファイルの存在をチェックしない場合には、そのサービスを無効にするときに script 方法が使用されます。

JASS_DISABLE_MODE 変数が conf に設定されている場合は、その構成ファイルを使用しないことによってサービスを無効にします。この方法は、開始前に構成ファイルが存在するかどうかを最初にチェックするサービスにおいては有効です。これらの無効にされた構成ファイルは Solaris OS パッチで置換されることはほとんどないため、この方法を使用すると、よりサポートしやすく維持しやすい構成になります。

この変数が script に設定されている場合は、それぞれの実行コントロールスクリプトを使用しないことによってサービスを無効にします。サービスの開始が許可されないと、サービスを実行することはできないので、この方法も有効です。ただし、Solaris OS パッチは実行コントロールスクリプトをインストールし、無効にされていたサービスを再有効化するので、あまりサポートしやすい構成ではありません。

注 – デフォルトの設定は変更しないでください。

注 – セキュリティースキャナを使用している場合は、この構成を使用してスキャナを適切にテストする必要があります。ほとんどのスキャナは一般に (しかも誤って) 実行コントロールスクリプトの存在のみをチェックするので、この変数を conf に設定すると、誤検出になる可能性があります。監査機能にはこの制約はありません。

JASS_DISPLAY_HOST_LENGTH

JASS_DISPLAY_HOSTNAME 変数が設定されている場合、この変数はホスト名に対して出力される文字数を設定します。

JASS_DISPLAY_HOSTNAME

注 – JASS_DISPLAY_HOSTNAME 変数は、JASS_VERBOSITY が 2 (要約モード) 以下の場合にのみ使用されます。

この変数は、監査処理中のホスト名情報の表示を制御します。Solaris Security Toolkit ソフトウェアで使用する詳細レベルを選択することができます。単一行出力モード (257 ページの「JASS_VERBOSITY」を参照) には、実行されているシステムのホスト名を各行に付けるオプションがあります。この値は、JASS_HOSTNAME と同じです。この情報を含めておくと、複数のシステムからの実行を処理するときに便利です。この変数を 1 に設定した場合、出力される各行の先頭に対象システムのホスト名が付加されます。それ以外の場合は、この情報は含まれません。デフォルトでは、この情報は表示されません。

JASS_DISPLAY_SCRIPT_LENGTH

JASS_DISPLAY_SCRIPTNAME 変数が設定されている場合、この変数はスクリプト名に対して出力される文字数を設定します。

JASS_DISPLAY_SCRIPTNAME

注 – JASS_DISPLAY_SCRIPTNAME 変数は、JASS_VERBOSITY が 2 (要約モード) 以下の場合にのみ使用されます。

この変数は、監査処理中の現在のスクリプト名の表示を制御します。Solaris Security Toolkit ソフトウェアで使用する詳細レベルを選択することができます。単一行出力モード (257 ページの「JASS_VERBOSITY」を参照) には、実行されている現在の監査スクリプト名を各行に付けるオプションがあります。この情報を含めておくと、失敗メッセージの発信元を判定するときに便利です。この変数を 1 に設定した場合、出力される各行の先頭に現在の監査スクリプト名が付加されます。それ以外の場合は、この情報は含まれません。デフォルトでは、この情報は含まれています。

JASS_DISPLAY_TIME_LENGTH

JASS_DISPLAY_TIMESTAMP 変数が設定されている場合、この変数はタイムスタンプに対して出力される文字数を設定します。

JASS_DISPLAY_TIMESTAMP

注 – JASS_DISPLAY_TIMESTAMP 変数は、JASS_VERBOSITY が 2 (要約モード) 以下の場合にのみ使用されます。

この変数は、監査処理中のタイムスタンプ情報の表示を制御します。Solaris Security Toolkit ソフトウェアで使用する詳細レベルを選択することができます。単一行出力モード (257 ページの「JASS_VERBOSITY」を参照) には、実行されているソフトウェアに関連するタイムスタンプを各行に付けるオプションがあります。この値は、JASS_TIMESTAMP と同じです。この情報を含めておくと、単一システムまたは一連のシステムからの複数の実行を処理するときに便利です。この変数を 1 に設定した場合、出力される各行の先頭に実行のタイムスタンプが付加されます。それ以外の場合は、この情報は含まれません。デフォルトでは、この情報は表示されません。

JASS_FILE_COPY_KEYWORD

この変数には、ファイルコピーに使用される、キーワード固有の接尾辞が含まれています。copy_files() 関数によって使用され、JASS_FILES ディレクトリ構造からさまざまなドライバ用のさまざまなファイルを取得します。

JASS_FILES

この変数は、対象システムにコピーするファイルシステムオブジェクトのリストを指定します。この変数にリストする各オブジェクトは、絶対パス名を使用して指定してください。各オブジェクトは、JASS_HOME_DIR/Files のルートディレクトリ内のファイルシステム階層に格納されます。

注 – JASS_FILES は user.init ファイルには追加できません。この変数を変更するには、関連する .driver ファイルを新しい名前にコピーし、その新しいファイルを変更します。

JASS_FILES 変数を使用したファイルの指定

注 – この機能は基本的には、JASS_FILES "+" 関数と同じです。

ファイルリストが一般ファイルリストの内容に追加されるのは、定義されている Solaris OS バージョンで Solaris Security Toolkit ソフトウェアが実行されている場合だけです。バージョン固有リストは、JASS_FILES 変数の末尾にオペレーティングシステムのメジャーおよびマイナーバージョン番号を下線で区切って付加して作成します。現在 Solaris Security Toolkit ソフトウェアでサポートしているオプションを、表 7-1 に示します。

表 7-1 JASS_FILES 変数でサポートする OS バージョン

変数	OS バージョン
JASS_FILES	すべてのバージョンの Solaris OS に適用され、追加ではなく上書きする
JASS_FILES_5_5_1	Solaris 2.5.1 OS のみに適用
JASS_FILES_5_6	Solaris 2.6 OS のみに適用
JASS_FILES_5_7	Solaris 7 OS のみに適用
JASS_FILES_5_8	Solaris 8 OS のみに適用
JASS_FILES_5_9	Solaris 9 OS のみに適用
JASS_FILES_5_10	Solaris 10 OS のみに適用

たとえば、/etc/logadm.conf ファイルは、Solaris 9 OS にのみ適用されます。Files/etc/logadm.conf ファイルだけを Solaris 9 OS にインストールするには、次の構文を使用してください。

```
JASS_FILES_5_9="
               /etc/logadm.conf
"
```

JASS_FILES 変数を使用すると、以下の方法でファイルを指定できます。

- Solaris Security Toolkit ソフトウェアからクライアントにコピーするファイルを指定する。

次の例は、hardening.driver の一部です。

```
JASS_FILES="
                /etc/dt/config/Xaccess
                /etc/init.d/set-tmp-permissions
                /etc/issue
                /etc/motd
                /etc/rc2.d/S00set-tmp-permissions
                /etc/rc2.d/S07set-tmp-permissions
                /etc/syslog.conf
"
```

このファイルを含めるように JASS_FILES 環境変数を定義することにより、クライアントにある /etc/motd ファイルが、Solaris Security Toolkit ソフトウェアの JASS_HOME_DIR/Files/etc/motd ファイルに置き換えられます。ファイル、ディレクトリ、シンボリックリンクは、Files ディレクトリに含めて、対応するドライバの JASS_FILES 定義に追加すればコピーできます。

- ホスト固有のファイルを指定する

  ホスト固有のファイルとは、対象システムのホスト名が Files ディレクトリのオブジェクトに割り当てられているホスト名と一致する場合にのみコピーされるファイルのことです。この機能を使用するには、次の形式で Files ディレクトリのファイルを作成します。

```
/etc/syslog.conf.$HOSTNAME
```

  このシナリオでは、ホスト名が *HOSTNAME* で定義されている値と一致する場合にのみ、JASS_HOME_DIR/Files/etc/syslog.conf.*HOSTNAME* ファイルが対象システム上の JASS_ROOT_DIR/etc/syslog.conf にコピーされます。syslog.conf と syslog.conf.*HOSTNAME* がどちらも存在する場合は、ホスト固有のファイルが優先されます。

- OS リリース固有のファイルを指定する

  OS リリース固有のファイルはホスト固有のファイルと概念は同じですが、対象システムの Solaris OS バージョンが、Files ディレクトリのオブジェクトに割り当てられている値と一致する場合にのみ、対象システムにコピーされます。この機能を使用するには、次の形式で Files ディレクトリのファイルを作成します。

```
/etc/syslog.conf+$OS
```

  この例では、対象システムの Solaris OS バージョンが OS で定義されている値と一致する場合にのみ、JASS_HOME_DIR/Files/etc/syslog.conf+OS ファイルが対象システムに JASS_ROOT_DIR/etc/syslog.conf としてコピーされます。

OS 変数は、uname -r コマンドで生成された出力をミラー化する必要があります。たとえば、Solaris 8 OS がセキュリティー保護されていた場合には、JASS_HOME_DIR/Files/etc/syslog.conf+5.8 という名前を持つファイルがコピーされます。このファイルが他の OS リリースにコピーされることはありません。OS 固有のファイルは汎用ファイルより優先されますが、OS 固有のファイルよりもホスト固有のファイルの方が優先されます。

JASS_FILES 変数は、OS 固有の拡張子もサポートします。この拡張子は、特定の Solaris OS バージョンにのみコピーするファイルシステムオブジェクトのリストを指定するときに使用します。OS 固有の JASS_FILES 拡張子は、Solaris OS バージョン 5.5.1、5.6、7、8、9、および 10 でサポートされます。たとえば、Solaris 8 OS に対してのみファイルのリストをコピーするには、JASS_FILES_5_8 変数を定義して、この変数にコピーするファイルのリストを割り当てます。

JASS_FILES 変数のカスタマイズ

この節では、JASS_FILES 環境変数のカスタマイズ方法について説明します。以下のコーディング例は、Drivers/config.driver ファイルの一部です。このプロファイルファイルは、プラットフォーム上で基本の構成を行います。デフォルトでは config.driver ファイルはどのような状態であるかを次に示します。

```
JASS_FILES="
"
```

次のプロファイル例には、ファイルテンプレート、ドライバ、および終了スクリプトの使用方法例が示されています。driver.run 関数が呼び出されたときに、JASS_HOME_DIR/Files/ ディレクトリから /.cshrc ファイルと /.profile ファイルを対象プラットフォームにコピーするよう config.driver が構成されています。

```
JASS_FILES="
/.cshrc
/.profile
"
```

いずれかのファイルの内容を変更するには、JASS_HOME_DIR/Files/ ディレクトリに格納されているファイルのコピーを変更します。ファイルテンプレートを追加または削除するだけならば、そのように JASS_FILES 変数を変更します。変更管理メカニズムを使用して、Solaris Security Toolkit 構成の変更を追跡してください。詳細は、『Solaris Security Toolkit 4.2 管理マニュアル』の第 1 章「バージョンの管理」を参照してください。

Solaris Security Toolkit ソフトウェアでは、OS バージョン固有のファイルリストをサポートしています。詳細は、前節の 239 ページの「JASS_FILES 変数を使用したファイルの指定」を参照してください。

JASS_FILES_DIR

注 – 通常、この変数を変更する必要はありません。

この変数は、JASS_HOME_DIR にある Files ディレクトリの場所を指します。このディレクトリには、クライアントにコピーできるファイルシステムオブジェクトがすべて格納されています。

オブジェクトをシステムにコピーするには、JASS_FILES 変数またはその OS 固有の拡張子のいずれかに、ファイルをリストする必要があります。install-templates.fin スクリプトによって、セキュリティー強化処理中に、これらのオブジェクトがクライアントにコピーされます。JASS_FILES 変数は、個々のドライバ内で設定してください。この変数は、他の構成ファイルによっては定義されません。この変数を使用してファイルをコピーするその他の方法については、238 ページの「JASS_FILES」を参照してください。デフォルトでは、この変数は JASS_HOME_DIR/Files に設定されています。

JASS_FINISH_DIR

注 – 通常、この変数を変更する必要は**ありません**。

Solaris Security Toolkit ソフトウェアの規則では、終了スクリプトはすべて Finish ディレクトリに格納されます。しかし、格納先に柔軟性を持たせるために、JASS_FINISH_DIR 環境変数を使用すれば、別の場所に終了スクリプトを格納することができます。デフォルトでは、この変数は JASS_HOME_DIR/Finish に設定されています。

JASS_HOME_DIR

注 – Solaris Security Toolkit ソフトウェアがすでに存在している JumpStart インストールのサブディレクトリにインストールされる場合を除き、通常、この変数を変更する必要はありません。上記のケースに該当する場合は、SI_CONFIG_DIR の最後に Solaris Security Toolkit ソースのパスを追加して、SI_CONFIG_DIR/jass-*n.n* のように変更してください (*n.n*は、ソフトウェアの現在のバージョン番号)。

この変数は、Solaris Security Toolkit のソースツリーの場所を定義します。JumpStart モードでは、JASS_HOME_DIR 変数は、JumpStart SI_CONFIG_DIR 変数によって設定されます。スタンドアロンモードでは、ベースディレクトリに含まれている jass-execute スクリプトによって設定されます。

JASS_HOSTNAME

注意 – フレームワークのいくつかのコンポーネントが、正しく設定されているこの変数に依存しているため、この変数を**変更しないでください**。

この変数には、Solaris Security Toolkit ソフトウェアが実行されているシステムのホスト名を指定します。この変数は、driver.init スクリプト内で Solaris OS uname -n コマンドを使用して、実行中に設定されます。

JASS_ISA_CAPABILITY

注 – この環境変数は、バージョン 4.2 の時点で Solaris Security Toolkit ソフトウェアから削除されました。

注 – 通常、この変数を変更する必要は**ありません**。

この変数は、対象システムの Solaris OS 命令セットの能力を定義します。システムが 32 ビットと 64 ビットのどちらのモードで動作する能力があるか判定するときに、この変数を使用します。この判定が行われるのは、終了スクリプトで使用する命令セットアーキテクチャー (ISA) 情報を提供するためです。この変数の値は、Solaris OS パッケージ SUNWkvmx が存在しているかどうかのチェック結果に基づいて決まります。このパッケージがインストールされている場合には、システムは 64 ビット対応と見なされ、この変数は 64 に設定されます。インストールされていない場合は、システムは 32 ビットのみに対応と見なされ、この変数は 32 に設定されます。

JASS_LOG_BANNER

注 – logBanner 関数は、JASS_VERBOSITY変数が 3 (完全モード) 以上、かつ JASS_LOG_BANNER 変数が 0 でない場合にのみ、出力を表示します。

この変数は、logBanner 関数の動作を制御します。logBanner 関数は、Solaris Security Toolkit ソフトウェアで使用されるバナーメッセージをすべて生成する関数です。この変数が 0 に設定されている場合、logBanner 関数は何も情報を表示しません。それ以外の場合は、logBanner 関数は引数として渡された情報を表示します。この変数は、出力メッセージをよりニーズに合うように変更するときに使用します。デフォルトではこの変数は値を持たないため、logBanner 関数は通常どおり動作します。

JASS_LOG_ERROR

この変数は、logError 関数の動作を制御します。logError 関数は、接頭辞 [ERR] が付いたメッセージを生成する関数です。この変数が 0 に設定されている場合、logError 関数は何も情報を表示しません。それ以外の場合は、logError 関数は引数として渡された情報を表示します。この変数は、出力メッセージをよりニーズに合うように変更するときに使用します。デフォルトではこの変数は値を持たないため、logError 関数は通常どおり動作します。

JASS_LOG_FAILURE

この変数は、logFailure 関数の動作を制御します。logFailure 関数は、接頭辞 [FAIL] が付いたメッセージを生成する関数です。この変数が 0 に設定されている場合、logFailure 関数は何も情報を表示しません。それ以外の場合は、logFailure 関数は引数として渡された情報を表示します。この変数は、出力メッセージをよりニーズに合うように変更するときに使用します。デフォルトではこの変数は値を持たないため、logFailure 関数は通常どおり動作します。

JASS_LOG_NOTICE

この変数は、logNotice 関数の動作を制御します。logNotice 関数は、接頭辞 [NOTE] が付いたメッセージを生成する関数です。この変数が 0 に設定されている場合、logNotice 関数は何も情報を表示しません。それ以外の場合は、logNotice 関数は引数として渡された情報を表示します。この変数は、出力メッセージをニーズに合うように変更するときに使用します。デフォルトではこの変数は値を持たないため、logNotice 関数は通常どおり動作します。

JASS_LOG_SUCCESS

この変数は、logSuccess 関数の動作を制御します。logSuccess 関数は、接頭辞 [PASS] が付いたメッセージを生成する関数です。この変数が 0 に設定されている場合、logSuccess 関数は何も情報を表示しません。それ以外の場合は、logSuccess

関数は引数として渡された情報を表示します。この変数は、出力メッセージをよりニーズに合うように変更するときに使用します。デフォルトではこの変数は値を持たないため、logSuccess 関数は通常どおり動作します。

JASS_LOG_SUMMARY

この変数は、logSummary 関数の動作を制御します。logSummary 関数は、接頭辞 [SUMMARY] が付いたメッセージを生成する関数です。この変数が 0 に設定されている場合、logSummary 関数は何も情報を表示しません。それ以外の場合は、logSummary 関数は引数として渡された情報を表示します。この変数は、出力メッセージをよりニーズに合うように変更するときに使用します。デフォルトではこの変数は値を持たないため、logSummary 関数は通常どおり動作します。

JASS_LOG_WARNING

この変数は、logWarning 関数の動作を制御します。logWarning 関数は、接頭辞 [WARN] が付いたメッセージを生成する関数です。この変数が 0 に設定されている場合、logWarning 関数は何も情報を表示しません。それ以外の場合は、logWarning 関数は引数として渡された情報を表示します。この変数は、出力メッセージをよりニーズに合うように変更するときに使用します。デフォルトではこの変数は値を持たないため、logWarning 関数は通常どおり動作します。

JASS_MODE

注意 – この変数は**変更しないでください**。

この変数は、Solaris Security Toolkit の動作方法を定義します。この変数は次のいずれかの値を取ります。

- APPLY
- UNDO
- AUDIT
- CLEAN
- HISTORY_LAST
- HISTORY_FULL

スタンドアロンモードでは、jass-execute コマンドによってこの変数は APPLY に設定されています。JumpStart モードでは、デフォルトで APPLY に設定されています。この変数においては、APPLY はセキュリティー強化処理を意味します。

JASS_OS_REVISION

注意 – この変数は自動的に設定されるので、**変更しないでください**。

この変数は、Solaris Security Toolkit ソフトウェアが使用されているクライアントの OS バージョンを示すグローバル変数です。この変数は、`uname -r` コマンドにより `driver.init` スクリプト内で自動的に設定され、他のすべてのスクリプトがアクセスできるようにエクスポートされます。

JASS_OS_TYPE

この変数は、強化または監査が行われているシステムが Solaris OS システムであるかどうかを判定します。システムで Solaris OS の汎用バージョンが実行されている場合は `Generic` に設定されます。この変数は `driver.init` ファイルで設定されます。

JASS_PACKAGE_DIR

注 – 通常、この変数を変更する必要は**ありません**。

Solaris Security Toolkit ソフトウェアの規則では、インストールするソフトウェアパッケージはすべて `Packages` ディレクトリに格納されます。しかし、格納先に柔軟性をもたせるため、`JASS_PACKAGE_DIR` 変数を使用すれば、別の場所にパッケージを格納することができます。デフォルトでは、スタンドアロンモードで、この変数は `JASS_HOME_DIR/Packages` に設定されています。

ただし JumpStart モードでは、この変数は一時的なマウントポイント `JASS_ROOT_DIR/tmp/jass-packages` として定義されます。JumpStart インストール時に、JumpStart サーバーに格納されているパッケージディレクトリが、このクライアントでのパッケージディレクトリとしてマウントされます。

JASS_PATCH_DIR

注 – 通常、この変数を変更する必要は**ありません**。

Solaris Security Toolkit ソフトウェアの規則では、インストールするソフトウェアパッチはすべて `Patches` ディレクトリに格納されます。しかし、格納先に柔軟性をもたせるため、`JASS_PATCH_DIR` 変数を使用すれば、別の場所にパッチを格納することができます。デフォルトでは、スタンドアロンモードで、この変数は `JASS_HOME_DIR/Patches` に設定されています。

ただし、JumpStart モードでは、この変数は一時的なマウントポイント JASS_ROOT_DIR/tmp/jass-patches として定義されます。JumpStart インストール時に、JumpStart サーバーに格納されている実際のパッケージディレクトリが、このクライアントでのパッケージディレクトリとしてマウントされます。

JASS_PKG

注意 – この変数は**変更しないでください**。

この変数は、Solaris Security Toolkit ソフトウェアの Solaris OS パッケージ名を定義します。この変数には、SUNWjass という値が設定されています。

JASS_REPOSITORY

注意 – この変数は**変更しないでください**。

この変数は、実行ログ機能と元に戻す機能の一部です。JASS_REPOSITORY で指定したパスによって、必要な実行情報を格納するディレクトリが定義されます。この機能により、実行される各スクリプトに関連する情報、各スクリプトの結果の出力、および実行中にインストール、変更、または削除されたファイルのリストを簡単に入手できます。

この変数は、ソフトウェアの実行中に動的に変更されます。いずれかの init ファイルでこの変数に割り当てられていた値は上書きされます。デフォルトでは、以下の値がこの変数に割り当てられています。

JASS_ROOT_DIR/var/opt/JASS_PKG/run/JASS_TIMESTAMP

JASS_ROOT_DIR

注意 – この変数は自動的に設定されるので、変更しないでください。

この変数は、対象ファイルシステムのルートディレクトリを定義します。JumpStart モードでは、このディレクトリは常に /a です。スタンドアロンモードでは、この変数に / またはシステムのルートディレクトリを指定します。

Solaris Security Toolkit ソフトウェアバージョン 0.2 以降では、この変数の値は jass-execute スクリプトの中で自動的に設定されるので、手動で変更する必要はありません。

JASS_ROOT_HOME_DIR

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

デフォルトではこの変数は、Solaris 10 OS のルートホームディレクトリを `/root` に定義します。

- Solaris 10 OS では、使用するルートホームディレクトリを / から `/root` に変更する必要がない場合は、この変数を / に設定します。
- Solaris OS のそのほかのバージョンでは、この変数はデフォルトで / です。

JASS_RUN_AUDIT_LOG

注意 – この変数は**変更しないでください**。

この変数は、実行ログの一部です。この変数は、監査処理中に生成された出力を格納するファイルの名前とそのファイルへの絶対パスを定義します。処理過程でテストされた各監査チェックの出力だけでなく、実行されたスクリプトについても記録するために、この情報が収集されます。

このファイルには、生成されたエラーや警告が格納されます。このファイルに格納される情報は、監査処理中に画面に表示される出力と同じです。デフォルトでは、この変数は `JASS_REPOSITORY/jass-audit-log.txt` に設定されています。

JASS_RUN_CHECKSUM

注意 – この変数は**変更しないでください**。

この変数は、実行ログ機能と元に戻す機能の一部です。`JASS_HOME_DIR` に含まれている `jass-check-sum` プログラムでも、この変数は使用されます。この変数は、Solaris Security Toolkit ソフトウェアで使用されるチェックサム情報をすべて格納するファイルの名前とそのファイルへの絶対パスを定義します。この情報は、変更前後のファイルの状態を記録したものです。最後に Solaris Security Toolkit ソフトウェアによってファイルが変更されてから、ファイルが変更されたかどうかを判定するときに、この情報が使用されます。この情報は `JASS_REPOSITORY` ディレクトリ構造内に格納され、デフォルト値は次のとおりです。

```
JASS_REPOSITORY/jass-checksums.txt
```

JASS_RUN_CLEAN_LOG

注意 – この変数は**変更しないでください**。

この変数は、実行ログの一部です。この変数は、クリーンアップ処理中に生成された出力を格納するファイルの名前とそのファイルへの絶対パスを定義します。実行中にインストール、変更、または削除されたファイルをリストするだけでなく、実行されたスクリプトについても記録するために、この情報が収集されます。

このファイルには、生成されたエラーや警告が格納されます。このファイルに格納される情報は、クリーンアップ処理中に画面に表示される出力と同じです。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-cleanup-log.txt
```

JASS_RUN_FINISH_LIST

この変数の名前は、Solaris Security Toolkit 4.0 ソフトウェアのリリース前に変更されました。250 ページの「JASS_RUN_SCRIPT_LIST」を参照してください。

JASS_RUN_INSTALL_LOG

注意 – この変数は**変更しないでください**。

この変数は、実行ログの一部です。この変数は、強化処理中に生成された出力を格納するファイルの名前とそのファイルへの絶対パスを定義します。実行中にインストール、変更、または削除されたファイルをリストするだけでなく、実行されたスクリプトについても記録するために、この情報が収集されます。

このファイルには、生成されたエラーや警告が格納されます。このファイルに格納される情報は、強化処理中に画面に表示される出力と同じです。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-install-log.txt
```

JASS_RUN_MANIFEST

注意 – この変数は**変更しないでください**。

この変数は、実行ログ機能と元に戻す機能の一部です。この変数は、実行に関連するマニフェスト情報を格納するファイルの名前とそのファイルへの絶対パスを定義します。マニフェストファイルは、強化処理の一部として実行された操作を記録するファイルです。このファイルは元に戻す処理にも使用され、システムを復元するために、どのファイルをどのような順序で移動しなければならないか判定します。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-manifest.txt
```

JASS_RUN_SCRIPT_LIST

注意 – この変数は**変更しないでください**。

この変数は、実行ログの一部です。この変数は、処理時に実行された終了スクリプトと監査スクリプトすべてのリストを格納するファイルの名前とそのファイルへの絶対パスを定義します。この情報は、情報提供とデバッグのために収集され、JASS_REPOSITORY ディレクトリ構造内に格納されます。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-script-list.txt
```

JASS_RUN_UNDO_LOG

注意 – この変数は**変更しないでください**。

この変数は、実行ログの一部です。この変数は、元に戻す処理中に生成された出力を格納するファイルの名前とそのファイルへの絶対パスを定義します。実行中にインストール、変更、または削除されたファイルをリストするだけでなく、実行されたスクリプトについても記録するために、この情報が収集されます。

このファイルには、生成されたエラーや警告が格納されます。このファイルに格納される情報は、元に戻す処理中に画面に表示される出力と同じです。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-undo-log.txt
```

JASS_RUN_VALUES

注意 – この変数は**変更しないでください**。

この変数は、set/get_stored_keyword_val 関数を使用した処理中に保存される変数を保持するファイルの名前とそのファイルへの絶対パスを定義します。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-values.txt
```

注 – JASS_REPOSITORY/jass-values.txt ファイルは編集しないでください。

JASS_RUN_VERSION

注意 – この変数は**変更しないでください**。

この変数は、実行ログの一部です。この変数は、バージョンと実行関連の情報を格納するファイルの名前とそのファイルへの絶対パスを定義します。通常、このファイルには、実行中に Solaris Security Toolkit ソフトウェアで使用されるバージョン、モード、およびセキュリティープロファイルに関する情報が含まれます。この情報は、システムでソフトウェアが使用された方法を記録するために収集されます。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-version.txt
```

JASS_SAVE_BACKUP

注意 – JASS_SAVE_BACKUP の値を 0 に設定すると、Solaris Security Toolkit の元に戻す機能は使用できません。

この変数は、強化処理中のバックアップファイルの作成を制御します。デフォルトの値は 1 で、クライアントで変更されたファイルのバックアップコピーが作成されます。この値を 0 に変更すると、処理中に作成されたすべてのバックアップコピーが処理完了時に削除されます。

ファイルのバックアップコピーを作成したくない場合は、user.run スクリプトを変更してください。user.run スクリプトの値は、変数に設定された値よりも優先されます。

JASS_SCRIPT

注意 – この変数は**変更しないでください**。

この変数には、現在実行中の終了または監査スクリプトの名前が含まれています。

JASS_SCRIPT_ERROR_LOG

注意 – この変数は**変更しないでください。**

この変数には、処理実行中にエラーが発生したスクリプトのリストを保持するファイルのセットが含まれています。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-script-errors.txt
```

JASS_SCRIPT_FAIL_LOG

注意 – この変数は**変更しないでください。**

この変数には、処理実行中に失敗が発生したスクリプトのリストを保持するファイルのセットが含まれています。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-script-failures.txt
```

JASS_SCRIPT_NOTE_LOG

注意 – この変数は**変更しないでください。**

この変数には、処理実行中に注記 (NOTE) が発生したスクリプトのリストを保持するファイルのセットが含まれています。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-script-notes.txt
```

JASS_SCRIPT_WARN_LOG

注意 – この変数は**変更しないでください。**

この変数には、処理実行中に警告が発生したスクリプトのリストを保持するファイルのセットが含まれています。デフォルトでは、この変数は次の値に設定されています。

```
JASS_REPOSITORY/jass-script-warnings.txt
```

JASS_SCRIPTS

この変数は、特定のドライバを使用する場合に、対象システムで実行する終了スクリプトのリストを指定します。各エントリには、JASS_FINISH_DIR ディレクトリにある終了スクリプトと同じ名前の終了スクリプトを指定してください。

また、JASS_FINISH_DIR に格納されている各終了スクリプトに対応している監査スクリプトを JASS_AUDIT_DIR に格納してください。

注 – JASS_SCRIPTS は user.init ファイルには追加できません。この変数を変更するには、関連する .driver ファイルを新しい名前にコピーし、その新しいファイルを変更します。

JASS_SCRIPTS 変数を使用したファイルの指定

JASS_SCRIPTS 変数は、OS 固有の拡張子もサポートします。対象システムで特定のバージョンの Solaris OS が実行されているときにだけ実行する終了スクリプトのリストを指定するときに、この拡張子を使用します。バージョン固有リストは、JASS_SCRIPTS 変数の末尾にオペレーティングシステムのメジャーおよびマイナーバージョン番号を下線で区切って付加して作成します。Solaris Security Toolkit ソフトウェアでサポートしているオプションを、表 7-2に示します。

表 7-2 JASS_SCRIPTS 変数でサポートする OS バージョン

変数	OS バージョン
JASS_SCRIPTS	すべてのバージョンの Solaris OS に適用され、追加ではなく上書きする
JASS_SCRIPTS_5_5_1	Solaris 2.5.1 OS のみに適用
JASS_SCRIPTS_5_6	Solaris 2.6 OS のみに適用
JASS_SCRIPTS_5_7	Solaris 7 OS のみに適用
JASS_SCRIPTS_5_8	Solaris 8 OS のみに適用
JASS_SCRIPTS_5_9	Solaris 9 OS のみに適用
JASS_SCRIPTS_5_10	Solaris 10 OS のみに適用

たとえば、Solaris 9 OS でのみ disable-something.fin スクリプトを使用するには、次のコードをドライバに追加します。

```
JASS_SCRIPTS_5_9="
disable-something.fin
"
```

この例では、オペレーティングシステムを Solaris 9 OS と想定して、JASS_SCRIPTS の最後に disable-something.fin スクリプトを追加しています。

注 – OS 固有のファイルとスクリプトのリストは、常にファイルとスクリプトの汎用リストの最後に追加されます。結果的に、OS 固有のファイルとスクリプトは、一般的なファイルとスクリプトのあとで実行されることになります。たとえば、JASS_SCRIPTS が a b で、JASS_SCRIPTS_5_9 が c d である場合、追加すると、JASS_SCRIPTS は a b c d となり、JASS_SCRIPTS_5_9 は自動的に破棄されます。

JASS_SCRIPTS 変数のカスタマイズ

ドライバに対して終了スクリプトを追加または削除するには、必要に応じて JASS_SCRIPTS 変数を変更します。ドライバには、ファイルテンプレートとスクリプトを 1 つのセキュリティープロファイルにまとめるメカニズムがあります。これらのプロファイルを使用すると、論理的にカスタマイズする内容をグループ化することができます。たとえば、組織内のすべてのシステムに対して適用するベースラインを、1 つのプロファイルを使って定義することができます。あるいは、データベースサーバーとして動作しているセキュリティー保護されたシステムに対する変更も、1 つのプロファイルで定義することができます。これらのプロファイルは単独で使用することも、より複雑なプロファイルに組み込むこともできます。

```
JASS_SCRIPTS="
print-jass-environment.fin
install-recommended-patches.fin
install-jass.fin
set-root-password.fin
set-term-type.fin
"
```

この例では、driver.run 関数が実行されるときに、5 つの異なるスクリプトが構成されます。(driver.run についての詳細は、113 ページの「ドライバの関数と処理について」を参照してください。) これらの 5 つのスクリプトは、強化に直接関連していないシステム構成の変更を表しているので、config.driver にグループ化されています。

JASS_STANDALONE

注 – 通常、この変数を変更する必要は**ありません**。

この変数は、Solaris Security Toolkit ソフトウェアがスタンドアロンモードと JumpStart モードのどちらで実行されるか制御します。この変数は、JumpStart インストールの場合はデフォルトで `0` に設定され、`jass-execute` コマンドを使用して実行を開始した場合は `1` に設定されます。

JASS_SUFFIX

注意 – この変数は**変更しないでください**。

この変数は、ファイルのバックアップコピーに付加する必要がある接尾辞を決定します。デフォルトでは、この変数は次の値に設定されています。

```
JASS.JASS_TIMESTAMP
```

実行中に、タイムスタンプフィールドの値が、ファイルが作成された時間に合わせて変更されます。この処理により、すべてのバックアップファイル名が確実に一意のファイル名になります。

この変数は、実行中に動的に変更されます。この変数に `init` ファイルで割り当てられていた値は上書きされます。

JASS_TIMESTAMP

注 – 通常、この変数を変更する必要は**ありません**。

この変数により、次の `JASS_REPOSITORY` ディレクトリが作成されます。

```
/var/opt/SUNWjass/run/JASS_TIMESTAMP
```

すでに説明したように、このディレクトリには、Solaris Security Toolkit ソフトウェアの各処理に関するログとマニフェスト情報が含まれます。この変数には処理開始時のタイムスタンプが格納され、その値は処理中ずっと保持されます。結果として、各処理に対してその値は一意となります。この一意の値により、個々の処理に関する情報を、実行開始時刻に基づき、その他のすべての情報と明確に区別することができます。デフォルトでは、この変数は次の `date` に設定されています。

```
+%EY%m%d%OH%OM%S
```

このコマンドでは、YYYYMMDDHHMMSS という形式のタイムスタンプが作成されます。たとえば、2005 年 7 月 1 日午前 1 時 30 分に開始された処理は、20050701013000 という値で表されます。

JASS_UNAME

この変数は、Solaris Security Toolkit 4.0 リリースに先立ち、JASS_OS_REVISION に名前が変更されました。246 ページの「JASS_OS_REVISION」を参照してください。

JASS_UNDO_TYPE

注意 – この変数は**変更しないでください。**

この変数には、jass-execute command の起動に -b、-f、または -k オプションのいずれが使用されたか、あるいはこれらのオプションがまったく使用されなかったかに関する情報が含まれています。取り得る値は次のとおりです。

- BACKUP
- FORCE
- KEEP
- ASK

JASS_USER_DIR

この変数は、構成ファイル user.init と user.run の場所を指定します。デフォルトでは、この 2 つのファイルは JASS_HOME_DIR/Drivers ディレクトリに格納されています。これらの構成ファイルは、組織のニーズに合うように Solaris Security Toolkit ソフトウェアをカスタマイズするときに使用します。

Solaris Security Toolkit ソフトウェアをカスタマイズする必要がある場合、今後の Solaris Security Toolkit ソフトウェアのアップグレードの影響を最小限に抑えるために、これらの構成ファイル内でカスタマイズを行うようにしてください。

グローバル変数は、user.init ファイルまたはドライバ内のいずれかで作成して割り当ててください。新しい関数や、既存の関数の優先指定は、user.run ファイルで実行してください。変数や関数の優先指定はすべて、Solaris Security Toolkit ソフトウェアで定義されている変数や関数の優先指定より優先されます。

JASS_VERBOSITY

注意 – この変数は直接**変更しないでください**。その代わりに、-V オプションを付けて jass-execute コマンドを実行してください。

この変数は、Solaris Security Toolkit ソフトウェアで監査処理の実行結果を表示する方法を制御します。現在、0 ～ 4 の 5 段階の詳細レベルがサポートされています。この変数に 0 ～ 4 のいずれかの値を設定するには、-V オプションを付けて jass-execute コマンドを実行してください。

注 – 強化処理などのその他の操作では、この変数は 3 (完全モード) に設定されています。通常は、変更しないでください。

監査処理で使用される詳細レベルを、表 7-3 に示します。

表 7-3 監査処理の詳細レベル

レベル	説明
0	最終モード。このモードでは、検証全体の総合結果を示す 1 行のみの出力が生成されます。PASS または FAIL の結果だけが必要な場合に、このモードは便利です。
1	統合モード。このモードでは、各監査スクリプトごとに、その結果を示す 1 行の出力が生成されます。また、監査終了時にスコアの総合計が生成されるだけでなく、各スクリプトの終了時にもスコアの小計が生成されます。
2	要約モード。このモードは、統合詳細レベルの内容に、各監査スクリプト内の個々のチェックの結果を組み合わせたモードです。1 つの監査スクリプト内で成功したチェックと失敗したチェックを即座に判定する場合に、このモードは便利です。このモードの形式も、1 行に 1 つの結果が表示されます。
3	完全モード。初めての複数行詳細モードです。このモードでは、実行されているチェック、そのチェックの目的、および結果の判定方法がもっとよくわかるように、バナーとヘッダーが出力されます。このモードはデフォルトの詳細レベルで、Solaris Security Toolkit の検証機能を初めて使用するユーザーに適しています。
4	デバッグモード。このモードは、完全詳細モードを拡張したモードで、logDebug ログ関数によって生成される項目がすべて含まれます。現在、このモードは Solaris Security Toolkit 監査スクリプトでは使用されていませんが、完全を期すためと、管理者がコード内部にデバッグ文を埋めこめるように用意されています。

最低の詳細モードであるレベル 0 では、処理の全結果を表す 1 行だけが表示されます。このレベルでの出力は次のようになります。

```
# ./jass-execute -a secure.driver -V 0
secure.driver [PASS] Grand Total : 0 Error(s)
```

JASS_VERSION

注意 – この変数は**変更しないでください。**

この変数は、使用されているソフトウェアに関連付けられている Solaris Security Toolkit ソフトウェアのバージョンを定義します。この変数は、ソフトウェアのバージョンを記録して、ログ関数などの関数でそのバージョンを使用できるようにします。

JASS_ZONE_NAME

注 – Solaris OS バージョン 9 およびそれ以前のバージョンでは Solaris でゾーンを使用できないため、JASS_ZONE_NAME は自動的に global に設定されます。

ゾーンが使用できる Solaris 10 OS では、一部の Solaris Security Toolkit スクリプトはこの変数を使用して、スクリプトが global ゾーン内に存在しているかどうかをチェックします。ゾーンに対応する終了および監査スクリプトのリストを次に示します。

- disable-power-mgmt
- enable-bsm
- enable-ipfilter
- enable-priv-nfs-ports
- enable-rfc1948
- enable-stack-protection
- install-nddconfig
- install-security-mode

ゾーンに対応するスクリプトについての詳細は、表 1-4を参照してください。

スクリプトが global ゾーンでは実行されていない場合、スクリプトはその情報を logNotGlobalZone 関数を使用して記録し、終了します。

JASS_ZONE_NAME 変数は、/usr/bin/zonename を使用して、初期化時に Solaris Security Toolkit スクリプト内に設定されます。このコマンドが存在しない場合、変数は global に設定されます。

スクリプト動作変数を定義する

スクリプト動作変数は、終了スクリプトと監査スクリプトの動作に影響を与えるように、Solaris Security Toolkit ソフトウェアで定義および使用する変数です。Solaris Security Toolkit ソフトウェアでは、個々のサイトの要件に合わせて機能をカスタマイズするために、堅牢かつ柔軟性のあるフレームワークを提供しています。Toolkit ソフトウェアにより、ユーザーがサイト固有のカスタマイズを行うために変更しなければならないソースコード量が制限されます。スクリプト変数は、スクリプトのソースコードを変更せずにスクリプトの動作を変更できる、使いやすい方法を提供します。

これらの変数は、`JASS_HOME_DIR/Drivers/finish.init` ファイルで定義されます。これらはグローバル変数ですが、通常、その使用は終了スクリプトと監査スクリプトの小さなセットに制限されています。この章ですでに説明したように、これらの変数は、`user.init` ファイルまたは個々のドライバ内のいずれかで、静的、動的、複合的割り当てなどの方法でカスタマイズすることができます。

組織またはサイトのセキュリティーポリシーと要件に合うように、必要に応じてこれらの変数を調整してください。変数の調整を行うと、使用環境のセキュリティー状態を向上および維持するための最高の値が得られます。

ここでは、以下のスクリプト動作変数について説明します。

JASS_ACCT_DISABLE

この変数には、システムで無効にするユーザーアカウントのリストを指定します。強化処理では、`disable-system-accounts.fin` スクリプトによって、これらのアカウントが無効になります。監査処理では、`disable-system-accounts.aud` スクリプトによって、この変数で指定されているアカウントが無効になっているかどうかが検証されます。

デフォルトでは、次のアカウントが `JASS_ACCT_DISABLE` 変数に割り当てられています。

- daemon
- bin
- adm
- lp
- uucp
- nuucp
- nobody
- smtp
- listen
- noaccess
- nobody4
- smmsp

JASS_ACCT_REMOVE

この変数には、システムから削除するユーザーアカウントのリストを指定します。強化処理では、remove-unneeded-accounts.fin スクリプトによって指定したアカウントが削除されます。監査処理では、remove-unneeded-accounts.aud スクリプトによって、指定したアカウントがシステムに存在していないことが検証されます。

デフォルトでは、次のアカウントが JASS_ACCT_REMOVE 変数に割り当てられています。

- smtp
- listen
- nobody4

JASS_AGING_MAXWEEKS

この変数には、ユーザーはパスワードを変更しなければならなくなるまでにパスワードが有効である最大週数を示す数値を指定します。この変数のデフォルト値は 8 (週) です。この変数は次のスクリプトにより使用されます。

- set-user-password-reqs.fin
- set-user-password-reqs.aud

JASS_AGING_MINWEEKS

この変数には、ユーザーがパスワードを変更できるまでに経過しなければならない最小週数を示す数値を指定します。この変数のデフォルト値は 1 (週) です。この変数は次のスクリプトにより使用されます。

- set-user-password-reqs.fin
- set-user-password-reqs.aud

JASS_AGING_WARNWEEKS

この変数には、パスワードの有効期限が切れて、ユーザーが警告を受けるまでの週数を示す数値を指定します。この警告は、警告期間中にユーザーがログインすると表示されます。この変数のデフォルト値は 1 (週) です。

この変数は次のスクリプトにより使用されます。

- set-user-password-reqs.fin
- set-user-password-reqs.aud

JASS_AT_ALLOW

この変数には、at 機能と batch 機能の使用を許可するユーザーアカウントのリストを指定します。強化処理では、install-at-allow.fin スクリプトによって、この変数で指定されている各ユーザーアカウントが JASS_ROOT_DIR/etc/cron.d/at.allow ファイルに追加されます (存在していない場合)。同様に、監査処理では、install-at-allow.aud スクリプトによって、この変数で指定されている各ユーザーアカウントが at.allow ファイルにリストされているかどうかが判定されます。

注 – 追加またはチェックを行うユーザーアカウントは、JASS_PASSWD にも存在していなければなりません。

デフォルトでは、この変数にはユーザーアカウントは指定されていません。

JASS_AT_DENY

この変数には、at 機能と batch 機能の使用を拒否するユーザーアカウントのリストを指定します。強化処理では、update-at-deny.fin スクリプトによって、この変数で指定されている各ユーザーアカウントが JASS_ROOT_DIR/etc/cron.d/at.deny ファイルに追加されます (存在していない場合)。同様に、監査処理では、update-at-deny.aud スクリプトによって、この変数で指定されている各ユーザーアカウントが at.deny ファイルにリストされているかどうかが判定されます。

注 – 追加またはチェックを行うユーザーアカウントは、JASS_PASSWD にも存在していなければなりません。

デフォルトでは、この変数には、JASS_PASSWD ファイルで定義されている、システム上のユーザーアカウントがすべて指定されています。

注 – JASS_AT_DENY 変数定義を、変更することなく finish.init ファイルから user.init ファイルにコピーする場合、終了スクリプトまたは監査スクリプトでこの変数を使用すると、入力を待機しているため Solaris Security Toolkit ソフトウェアはハングしているように見えます。このような事態を防止するには、user.init ファイル内で JASS_AT_DENY 変数の前で JASS_PASSWD 変数を定義するか、JASS_PASSWD への参照を削除します。

JASS_BANNER_DTLOGIN

この変数には、CDE へのログイン後にユーザーに表示するバナーメッセージが含まれているファイル名を表す文字列値を指定します。強化処理では、set-banner-dtlogin.fin スクリプトによって、このバナーがインストールされます。監査処理では、set-banner-dtlogin.aud スクリプトによって、このバナーの存在がチェックされます。この変数のデフォルト値は /etc/motd です。

JASS_BANNER_FTPD

注 – この変数は、Solaris OS バージョン 2.6 ～ 8 を実行しているシステムにのみ使用されます。

この変数には、FTP サービスの認証前にユーザーに表示するバナーとして使用される文字列値を指定します。強化処理では、set-banner-ftpd.fin スクリプトによって、このバナーがインストールされます。監査処理では、set-banner-ftpd.aud スクリプトによって、このバナーの存在がチェックされます。この変数のデフォルト値は \"Authorized Use Only\" です。

注 – 引用符文字がコマンドシェルと解釈されないように、直前の文字列をバックスラッシュ文字で囲む必要があります。該当する FTP 構成ファイルにインストールされている場合には、この文字列は “Authorized Use Only” と表示されます。

JASS_BANNER_SENDMAIL

この変数には、sendmail サービスへの接続直後にクライアントに表示するバナーとして使用される文字列値を指定します。強化処理では、set-banner-sendmail.fin スクリプトによって、このバナーがインストールされます。監査処理では、set-banner-sendmail.aud スクリプトによって、このバナーの存在がチェックされます。この変数のデフォルト値は Mail Server Ready です。

JASS_BANNER_SSHD

この変数には、Secure Shell サービスの認証前にユーザーに表示するバナーメッセージが含まれているファイル名を表す文字列値を指定します。強化処理では、set-banner-sshd.fin スクリプトによって、このバナーがインストールされます。監査処理では、set-banner-sshd.aud スクリプトによって、このバナーの存在がチェックされます。この変数のデフォルト値は /etc/issue です。

JASS_BANNER_TELNETD

この変数には、Telnet サービスの認証前にユーザーに表示するバナーとして使用される文字列値を指定します。強化処理では、set-banner-telnetd.fin スクリプトによって、このバナーがインストールされます。監査処理では、set-banner-telnetd.aud スクリプトによって、このバナーの存在がチェックされます。この変数のデフォルト値は \"Authorized Use Only\" です。

注 – 引用符文字がコマンドシェルと解釈されないように、直前の文字列をバックスラッシュ文字で囲む必要があります。該当する Telnet 構成ファイルにインストールされている場合には、この文字列は “Authorized Use Only” と表示されます。

JASS_CORE_PATTERN

この変数には、coreadm 機能で使用されるパス名とコアファイルの命名パターンを表す文字列を指定します。システム上で生成されるコアファイルを、指定したディレクトリに制限するように coreadm を構成し、この変数で指定したファイルパターンに基づいて命名するときに、この変数が使用されます。強化処理では、enable-coreadm.fin スクリプトによって coreadm が構成されます。監査処理では、enable-coreadm.aud スクリプトによって coreadm 構成がチェックされます。この変数のデフォルト値は次のとおりです。

```
/var/core/core_%n_%f_%u_%g_%t_%p
```

ファイルの命名オプションについての詳細は、coreadm(1M) のマニュアルページを参照してください。

JASS_CPR_MGT_USER

この変数には、システム上でチェックポイント機能と再開機能の実行が許可されるユーザーを定義する文字列値を指定します。強化処理では、set-power-restrictions.fin スクリプトによってこの制限が実装されます。監査処理では、set-power-restrictions.aud スクリプトによってこの制限がチェックされま

す。デフォルト値は "-" で、これは、root アカウントだけにこれらの管理機能の実行が許可されることを示します。詳細は、第 3 章の「/etc/default/power」を参照してください。

JASS_CRON_ALLOW

この変数には、cron 機能の使用を許可するユーザーアカウントのリストを指定します。強化処理では、update-cron-allow.fin スクリプトによって、この変数で指定されている各ユーザーが JASS_ROOT_DIR/etc/cron.d/cron.allow ファイルに追加されます (存在していない場合)。同様に、監査処理では、update-cron-allow.aud スクリプトによって、この変数で指定されている各ユーザーが cron.allow ファイルにリストされているかどうかが判定されます。

注 – 追加またはチェックを行うユーザーアカウントは、JASS_PASSWD にも存在していなければなりません。

デフォルトでは、この変数には root アカウントのみ指定されています。

JASS_CRON_DENY

この変数には、cron 機能の使用を拒否するユーザーアカウントのリストを指定します。強化処理では、update-cron-deny.fin スクリプトによって、この変数で指定されている各ユーザーが JASS_ROOT_DIR/etc/cron.d/cron.deny ファイルに追加されます (存在していない場合)。同様に、監査処理では、update-cron-deny.aud スクリプトによって、この変数で指定されている各ユーザーが cron.deny ファイルにリストされているかどうかが判定されます。

注 – 追加またはチェックを行うユーザーアカウントは、JASS_PASSWD にも存在していなければなりません。

デフォルトでは、この変数には JASS_PASSWD ファイルで定義され、100 より小さく、かつ 60000 より大きいユーザー ID を持つユーザーアカウントがすべて指定されています。一般に、100 より小さく 60000 より大きい範囲は、管理アクセス用に予約されています。デフォルトでは、root アカウントは明示的にこのリストから除外されています。

注 – JASS_CRON_DENY 変数定義を、変更することなく finish.init ファイルから user.init ファイルにコピーする場合、終了スクリプトまたは監査スクリプトでこの変数を使用すると、入力を待機しているため Solaris Security Toolkit ソフトウェアはハングしているように見えます。このような事態を防止するには、user.init ファイル内で JASS_CRON_DENY 変数の前で JASS_PASSWD 変数を定義するか、JASS_PASSWD への参照を削除します。

JASS_CRON_LOG_SIZE

この変数には、ローテーション前の cron 機能のログファイルの最大サイズをブロックで表した数値を指定します。強化処理では、update-cron-log-size.fin スクリプトによってこの設定がインストールされます。監査処理では、update-cron-log-size.aud スクリプトによってこの設定がチェックされます。この変数のデフォルト値は 20480 (20 MB) です。このサイズは、デフォルトの Solaris OS 値 1024 (0.5 MB) よりも増加しています。

JASS_CRYPT_ALGORITHMS_ALLOW

この変数は、許可されたパスワード暗号化アルゴリズムを格納します。値は、次の中の 1 つまたは複数です。

- 1 – BSD/Linux md5
- 2a – BSD Blowfish
- md5 – Sun md5

JASS_CRYPT_DEFAULT

この変数には、システムに対して構成されているデフォルトの暗号アルゴリズムが含まれています。デフォルトの設定は 1 で、BSD MD5 に対応します。この変数は、CRYPT_DEFAULT 変数の /etc/security/crypt.conf ファイルで Solaris OS のデフォルトを変更するために、set-flexible-crypt.fin スクリプトで使用されます。

JASS_CRYPT_FORCE_EXPIRE

この変数は、暗号設定の変更のあと、すべてのパスワードの変更を強制するかどうかを Solaris Security Toolkit に通知します。1 に設定されている場合、set-flexible-crypt.fin スクリプトは passwd -f コマンドを使用して、次回ログイン時にパスワードを変更するようすべてのユーザーに強制します。デフォルトは次のとおりです。

- 汎用ドライバまたは secure.driver = 1

- server ドライバ = 0
- suncluster ドライバ = 0
- sunfire_15k_sc ドライバ = 0

JASS_FIXMODES_DIR

この変数には、FixModes ソフトウェアが存在する場合に、FixModes ソフトウェアへの絶対パスを表す文字列を指定します。FixModes ソフトウェアは、Solaris Security Toolkit によってソフトウェア配布からインストールされている場合は、この変数で指定されているディレクトリにインストールされます。強化処理では、FixModes ソフトウェアをインストールおよび実行するときに、install-fix-modes.fin スクリプトでこの変数が使用されます。監査処理では、install-fix-modes.aud スクリプトで FixModes ソフトウェアが実行されます。この変数のデフォルト値は /opt です。

JASS_FIXMODES_OPTIONS

この変数には、FixModes ソフトウェアが install-fix-modes.fin スクリプトから強化処理中に実行されるときに、FixModes ソフトウェアに渡されるオプションのリストを指定します。監査処理では、この変数は使用されません。デフォルトでは、この変数にはオプションは指定されていません。

JASS_FTPD_UMASK

この変数には、FTP サービスで使用されるファイル生成マスク (umask) を表す 8 進数の数値を指定します。強化処理では、set-ftpd-umask.fin スクリプトによってこの設定がインストールされます。監査処理では、set-ftpd-umask.aud スクリプトによってこの設定がチェックされます。この変数のデフォルト値は 022 です。

JASS_FTPUSERS

この変数には、FTP サービスの使用を拒否するユーザーアカウントのリストを指定します。強化処理では、install-ftpusers.fin スクリプトによって、この変数で指定されている各ユーザーが次のいずれかに追加されます。

Solaris 8 OS またはそれ以前では、JASS_ROOT_DIR/etc/ftpusers ファイル

Solaris 9 または 10 OS では、JASS_ROOT_DIR/etc/ftpd/ftpusers ファイル (まだ存在してない場合)

同様に、監査処理では、install-ftpusers.aud スクリプトによって、この変数で指定されている各ユーザーアカウントが ftpusers ファイルにリストされているかどうかが判定されます。デフォルトでは、この変数には JASS_PASSWD ファイルで定

義され、100 より小さく、かつ 60000 より大きいユーザー ID を持つユーザーアカウントがすべて指定されています。一般に、100 より小さく 60000 より大きい範囲は、管理アクセス用に予約されています。

注 – JASS_FTPUSERS 変数定義を、変更することなく finish.init ファイルから user.init ファイルにコピーする場合、終了スクリプトまたは監査スクリプトでこの変数を使用すると、入力を待機しているため Solaris Security Toolkit ソフトウェアはハングしているように見えます。このような事態を防止するには、user.init ファイル内で JASS_FTPUSERS 変数の前で JASS_PASSWD 変数を定義するか、JASS_PASSWD への参照を削除します。

JASS_KILL_SCRIPT_DISABLE

注 – Solaris 10 OS では実行コントロールスクリプトがサービス管理機能 (SMF) により管理されるため、この変数は Solaris 10 OS を実行しているシステムでは使用されません。

この 変数には、サービスが無効のときに kill 実行コントロールスクリプトを無効にするか、そのままにしておくかを判定するブール型の値を指定します。start 実行コントロールスクリプトは常に無効です。手動で開始したサービスが、システムのシャットダウンまたはリ再起動時に正しく終了するように、kill スクリプトをそのままにしておくことを希望する管理者もいます。デフォルトでは、この変数は 1 に設定され、kill 実行コントロールスクリプトは無効になっています。この変数を 0 に設定すると、kill 実行コントロールスクリプトをそのままにしておく構成になります。

JASS_LOGIN_RETRIES

この変数には、ログインプロセスが失敗を記録して接続を終了するまでに繰り返すことができる、ログイン失敗回数を示す数値を指定します。また Solaris 10 OS を実行するシステムでは、これ以上のログイン試行を防止するため、アカウントをロックします。強化処理では、set-login-retries.fin スクリプトによってこの設定がインストールされます。監査処理では、set-login-retries.aud スクリプトによってこの設定がチェックされます。この変数のデフォルト値は 3 です。

JASS_MD5_DIR

注 – Solaris 10 OS では /usr/bin/digest コマンドが MD5 機能を提供するため、この変数は Solaris 10 OS を実行しているシステムでは使用されません。

この変数には、MD5 ソフトウェアが存在する場合に、MD5 ソフトウェアへの絶対パスを表す文字列を指定します。MD5 ソフトウェアは、Solaris Security Toolkit によってソフトウェア配布からインストールされている場合は、この変数で指定されているディレクトリにインストールされます。強化処理では、install-md5.fin スクリプトによって MD5 ソフトウェアをインストールするときにこの変数が使用されます。監査処理では、install-md5.aud スクリプトによって、この変数で指定されている場所に MD5 ソフトウェアが存在するかどうかがチェックされます。この変数のデフォルト値は /opt です。

JASS_NOVICE_USER

この変数は、Solaris Security Toolkit を初めて使用するユーザーに対する情報の表示を制御します。この変数を使用すると、経験の浅い管理者に追加ガイダンスを表示できます。デフォルトは 1 で、初心者ユーザーであることを意味します。JASS_HOME_DIR/Drivers/user.init ファイルの JASS_NOVICE_USER 変数を 0 (ゼロ) に設定すれば、この機能を無効にできます。

JASS_PASSWD 環境変数

特別に指定されていない限り、この節の JASS_PASS_ 環境変数は set-strict-password-checks.[fin|aud] スクリプトにより使用されます。これらの変数は Solaris Security Toolkit ソフトウェアにより使用され、Solaris 10 OS の JASS_PASS_ 接頭辞のない対応する変数の /etc/default/passwd ファイルにある値を監査します。(JASS_PASS_ 接頭辞のない) 基本的な変数の詳細については、passwd(1) のマニュアルページを参照してください。

JASS_PASS_DICTIONDBDIR

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には、生成された辞書データベースが存在するディレクトリが含まれます。デフォルトは次のとおりです。

- secure.driver = /var/password
- server ドライバ = /var/password
- suncluster ドライバ = /var/password
- sunfire_15k_sc ドライバ = /var/password

(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

JASS_PASS_DICTIONLIST

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には、JASS_PASS_DICTIONLIST=file1,file2,file3 のように、コンマで区切った辞書ファイルのリストを含めることができます。デフォルトは次のとおりです。

- secure.driver = /usr/share/lib/dict/words
- server ドライバ = /usr/share/lib/dict/words
- suncluster ドライバ = /usr/share/lib/dict/words
- sunfire_15k_sc ドライバ = /usr/share/lib/dict/words

(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

JASS_PASS_HISTORY

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には特定のドライバ用の HISTORY 値が含まれ、enable-password-history.fin および enable-password-history.aud スクリプトによりドライバ上のパスワード履歴をチェックするために使用されます。デフォルトは次のとおりです。

- secure.driver = 10
- server ドライバ = 4
- suncluster ドライバ = 4
- sunfire_15k_sc ドライバ = 4

JASS_PASS_LENGTH

この変数には、ユーザーパスワードの最小文字数を示す数値を指定します。この変数のデフォルト値は 8 (文字) です。この変数は、set-user-password-reqs.[fin|aud] スクリプトで使用されます。

JASS_PASS_MAXREPEATS

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には、パスワード内で許容される連続する繰り返し文字の最大数が含まれます。デフォルトは次のとおりです。

- `secure.driver` = 1
- `server` ドライバ = 2
- `suncluster` ドライバ = 2
- `sunfire_15k_sc` ドライバ = 2

(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

JASS_PASS_MINALPHA

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には、パスワード内で必要なアルファベット文字の最小数が含まれます。デフォルトは次のとおりです。

- `secure.driver` = 4
- `server` ドライバ = 3
- `suncluster` ドライバ = 3
- `sunfire_15k_sc` ドライバ = 3

(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

JASS_PASS_MINDIFF

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には、古いパスワードと新しいパスワードの間で必要な最低限の差異が含まれます。デフォルトは次のとおりです。

- `secure.driver` = 7
- `server` ドライバ = 5
- `suncluster` ドライバ = 5
- `sunfire_15k_sc` ドライバ = 5

(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

JASS_PASS_MINDIGIT

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には、パスワードに必要な数字の最小数が含まれます。デフォルトは次のとおりです。

- secure.driver = 1
- server ドライバ = 1
- suncluster ドライバ = 1
- sunfire_15k_sc ドライバ = 1

(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

注 – JASS_PASS_MINNONALPHA が設定されている場合、Solaris Security Toolkit はその値を使用し、JASS_PASS_MINDIGIT と JASS_PASS_MINSPECIAL を無視します。

JASS_PASS_MINLOWER

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には、必要な小文字の最小数が含まれます。デフォルトは次のとおりです。

- secure.driver = 2
- server ドライバ = 2
- suncluster ドライバ = 2
- sunfire_15k_sc ドライバ = 2

(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

JASS_PASS_MINNONALPHA

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には、パスワードに必要な (数字と特殊文字を含む) アルファベット以外の文字の最小数が含まれます。デフォルトは次のとおりです。

- secure.driver = なし
- server ドライバ = 1
- suncluster ドライバ = 1
- sunfire_15k_sc ドライバ = 1

(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

注 – JASS_PASS_MINNONALPHA が設定されている場合、Solaris Security Toolkit はその値を使用し、JASS_PASS_MINDIGIT と JASS_PASS_MINSPECIAL を無視します。

JASS_PASS_MINSPECIAL

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には、パスワードに必要な (アルファベットや数字以外の) 特殊文字の最小数が含まれます。デフォルトは次のとおりです。

- secure.driver = 1
- server ドライバ = 1
- suncluster ドライバ = 1
- sunfire_15k_sc ドライバ = 1

(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

注 – JASS_PASS_MINNONALPHA が設定されている場合、Solaris Security Toolkit はその値を使用し、JASS_PASS_MINDIGIT と JASS_PASS_MINSPECIAL を無視します。

JASS_PASS_MINUPPER

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数には、パスワードに必要な大文字の最小数が含まれます。デフォルトは次のとおりです。

- secure.driver = 2
- server ドライバ = 2
- suncluster ドライバ = 2
- sunfire_15k_sc ドライバ = 2

(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

JASS_PASS_NAMECHECK

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数は、パスワードとログイン名の照合を有効/無効にするために使用されます。すべてのドライバでデフォルト値は YES で、照合が有効であることを意味します。(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

JASS_PASS_WHITESPACE

注 – この変数は、Solaris 10 OS を実行しているシステムでのみ使用します。

この変数は、パスワード内で空白文字を許可するかどうかを決定するために使用されます。すべてのドライバでデフォルト値は YES で、空白文字が使用できることを意味します。(詳細は、269 ページの「JASS_PASSWD 環境変数」を参照してください。)

JASS_PASSWD

注 – この変数を変更する必要は**ありません**。

この変数には、対象システムでのパスワードファイルの場所を示す文字列を指定します。この変数は、多くのスクリプトで使用されるだけでなく、多数の変数を動的に割り当てる場合にも使用されます。この変数のデフォルト値は、次のとおりです。

```
JASS_ROOT_DIR/etc/passwd
```

JASS_POWER_MGT_USER

この変数には、システム上で電源管理機能の実行が許可されるユーザーを定義する文字列値を指定します。強化処理では、set-power-restrictions.fin スクリプトによってこの制限が実装されます。監査処理では、set-power-restrictions.aud スクリプトによってこの制限がチェックされます。デフォルト値は “-” で、これは、root アカウントだけにこれらの管理機能の実行が許可されることを示します。詳細は、第 3 章の「/etc/default/power」を参照してください。

JASS_REC_PATCH_OPTIONS

この変数には、Solaris 推奨およびセキュリティパッチクラスタをシステムにインストールするときに、patchadd コマンドまたは installpatch コマンドに渡されるオプションを示す文字列を指定します。使用可能なオプションについては、patchadd(1M) マニュアルページまたは installpatch プログラムコードを参照してください。強化処理では、パッチクラスタをシステムにインストールするときに、install-recommended-patches.fin スクリプトでこの変数が使用されます。監査処理では、この変数は使用されません。デフォルトでは、この変数にはオプションは割り当てられていません。

JASS_RHOSTS_FILE

この変数には、システム上にある .rhosts ファイルまたは hosts.equiv ファイルのリストが格納されているファイルを示す文字列値を指定します。強化処理では、print-rhosts.fin スクリプトでこの変数が使用されます。監査処理では、この変数は使用されません。デフォルトでは、この変数にはファイル名は割り当てられていません。その結果、print-rhosts.fin スクリプトの出力が画面に表示されます。

JASS_ROOT_GROUP

この変数には、root ユーザーの一次グループ識別子の値として使用される数値を指定します。強化処理では、set-root-group.fin スクリプトによってこの設定がインストールされます。監査処理では、set-ftpd-umask.aud スクリプトによってこの設定がチェックされます。デフォルトでは、この変数は 0 (つまり root) に設定されています。この値は、Solaris OS のデフォルト値 1 (つまり other) よりも優先されます。

JASS_ROOT_PASSWORD

注意 – この文字列の値は、Solaris Security Toolkit ソフトウェア出荷時のデフォルトの値から変更してください。変更しない場合、デフォルトのパスワードは一般に知られているため、システムが攻撃を受ける可能性があります。

注 – このスクリプトは、システムが JumpStart インストール時に miniroot から実行された場合にのみ動作し、root パスワードが一般によく知られた値で不用意に上書きされるのを防止します。

この変数には、root アカウントの暗号化パスワードとして使用される文字列値を指定します。強化処理では、set-root-password.fin スクリプトによってこの設定がインストールされます。監査処理では、set-root-password.aud スクリプトによってこの設定がチェックされます。デフォルトでは、この変数は次の値に設定されています。

```
JdqZ5HrSDYM.o
```

この暗号化文字列は、クリアテキスト文字列 t00lk1t と同じです。

JASS_SADMIND_OPTIONS

この変数には、inetd プロセスから実行される sadmind デーモンで使用するオプションを示す文字列値を指定します。強化処理では、install-sadmind-options.fin スクリプトによってこの設定がインストールされます。監査処理では、install-sadmind-options.aud スクリプトによってこの設定がチェックされます。この変数のデフォルト値は -S 2 で、これは sadmind デーモンに、クライアントとの通信に強力な認証 (AUTH_DES) を使用するように指示します。

JASS_SENDMAIL_MODE

注 – sendmail のバージョンと構成が変更されたため、この変数は Solaris 8 OS にのみ使用されます。Solaris 8 OS バージョン以外で同じ処理を実行するときは、他のメカニズムが使用されます。詳細は、150 ページの「disable-sendmail.fin」を参照してください。

この変数には、sendmail デーモンで操作を決定するときに使用されるオプションを示す文字列値を指定します。強化処理では、disable-sendmail.fin スクリプトによって、この変数で指定されている操作を行うようにデーモンが構成されます。監査処理では、disable-sendmail.aud スクリプトによって、sendmail デーモンが正しい操作を行うように構成されているかチェックが行われます。この変数のデフォルト値は \"\" です。この値は、sendmail デーモンがキュー処理モードでのみ動作することを示します。この値は、sendmail デーモンがデーモンとして動作し、着信メールを受信するように構成されているデフォルト値よりも優先されます。

注 – 引用符文字がコマンドシェルと解釈されないように、直前の文字列をバックスラッシュ文字で囲む必要があります。該当する sendmail 構成ファイルでインストールされるときには、この文字列は "" として表示されます。

JASS_SGID_FILE

この変数には、システム上にある set-group-id ファイルのリストが格納されているファイルを示す文字列値を指定します。強化処理では、print-sgid-files.fin スクリプトでこの変数が使用されます。監査処理では、この変数は使用されません。デフォルトでは、この変数にはファイル名は割り当てられていません。その結果、print-sgid-files.fin スクリプトの出力が画面に表示されます。

JASS_SHELLS

この変数には、JASS_ROOT_DIR/etc/shells ファイルに追加するシェルのリストを指定します。強化処理では、install-shells.fin スクリプトによって、この変数で指定されている各シェルが JASS_ROOT_DIR/etc/shells ファイルに追加されます (存在していない場合)。同様に、監査処理では、install-shells.aud スクリプトによって、この変数で指定されている各シェルが shells ファイルにリストされているかどうかが判定されます。

この変数のデフォルト値は次のとおりです。

- /bin/csh
- /bin/jsh
- /bin/ksh
- /bin/sh
- /sbin/sh
- /sbin/jsh
- /usr/bin/csh
- /usr/bin/jsh
- /usr/bin/ksh
- /usr/bin/sh

Solaris OS 8 以降では、次のシェルがデフォルト値に追加されています。

- /bin/bash
- /bin/pfcsh
- /bin/pfksh
- /bin/pfsh
- /bin/tcsh
- /bin/zsh
- /usr/bin/bash
- /usr/bin/pfcsh
- /usr/bin/pfksh
- /usr/bin/pfsh
- /usr/bin/tcsh
- /usr/bin/zsh

JASS_SUID_FILE

この変数には、システム上にある set-user-id ファイルのリストが格納されているファイルを示す文字列値を指定します。強化処理では、print-suid-files.fin スクリプトでこの変数が使用されます。監査処理では、この変数は使用されません。デフォルトでは、この変数にはファイル名は割り当てられていません。その結果、print-suid-files.fin スクリプトの出力が画面に表示されます。

JASS_SUSPEND_PERMS

この変数には、システムの中断または再開機能の実行が許可されるユーザーを定義する文字列値を指定します。強化処理では、set-sys-suspend-restrictions.fin スクリプトによってこの制限が実装されます。監査処理では、set-sys-suspend-restrictions.aud スクリプトによってこの制限がチェックされます。デフォルト値は "-" で、これは、root アカウントだけにこれらの管理機能の実行が許可されることを示します。詳細は、/etc/default/sys-suspend ファイルを参照してください。

JASS_SVCS_DISABLE

注意 – サービスのデフォルトのリストを使用する場合には、Telnet、RSH、および RLOGIN サーバーは Solaris Security Toolkit に含まれる大半のドライバですべて無効になっているため、システムへのコンソールによるアクセス、Secure Shell アクセス (Solaris 9 または 10 OS の場合)、あるいはデフォルトでない遠隔アクセスのいずれかの機能を確保してください。

Solaris 10 OS では、この変数には無効にする inetd の制御下にある SMF 対応サービスのリストが含まれています。JASS_SVCS_DISABLE スクリプトは、リスト上の SMF 対応でシステムにインストールされているすべてのサービスを無効にします。このリスト上のエントリは、第 1 章の表 1-1 にリストされている FMRI の形式である必要があります。このリストには、レガシーの SMF 以外のサービスを含めることもできます。これらが有効であるためには、サービスを JASS_ROOT_DIR/etc/inet/inetd.conf ファイルで定義する必要があります。定義されていない場合、エントリは無視されます。

バージョン **10** よりも前の **Solaris OS** では、この変数は、JASS_ROOT_DIR/etc/inet/inetd.conf ファイルからさまざまなサービスを簡単に削除できるようにします。強化処理では、そのサービスが JASS_SVCS_ENABLE 変数にもリストされていない限り、この変数で指定されている各 inetd サービスが update-inetd-conf.fin スクリプトによって無効になります。同様に、監査処理では、update-inetd-conf.aud スクリプトによって、正しい inetd サービスが

システムで無効になっているか判定されます。デフォルトでは、この変数によって無効にされるサービスのリストには、Solaris OS からデフォルトで提供されているエントリがすべて含まれています。

JASS_SVCS_DISABLE 変数と JASS_SVCS_ENABLE 変数は、スクリプトそのものを変更する必要なしに update-inetd-conf.fin のデフォルトの動作を変更する、簡単かつ使いやすいメカニズムを提供します。これらの変数を変更する場合は、次の 4 種類の構成パターンが考えられます。

例 **1 :**

JASS_SVCS_DISABLE (定義されている)

JASS_SVCS_ENABLE (定義されていない)

この例は、Solaris Security Toolkit ソフトウェアの旧バージョンとの下位互換性を保つためのデフォルトのケースです。このケースでは、update-inetd-conf.fin スクリプトが実行されているときは、JASS_SVCS_DISABLE にリストされているサービスは無効になっています。

例 **2 :**

JASS_SVCS_DISABLE (定義されていない)

JASS_SVCS_ENABLE (定義されている)

JASS_SVCS_ENABLE にリストされているサービスだけが有効になります。Sun 固有でないサービスも含め、その他のサービスはすべて無効になります。この例では、「明示的に許可されていないものはすべて拒否する」という原則を実行できます。

例 **3 :**

JASS_SVCS_DISABLE (定義されている)

JASS_SVCS_ENABLE (定義されている)

JASS_SVCS_DISABLE のサービスは無効になり、JASS_SVCS_ENABLE のサービスは有効のままです。リストに含まれていないサービスは影響を受けません。サービスが JASS_SVCS_ENABLE とJASS_SVCS_DISABLE の両方にリストされている場合は、JASS_SVCS_ENABLE が優先されるので、そのサービスは有効になります。

例 **4 :**

JASS_SVCS_DISABLE (定義されていない)

JASS_SVCS_ENABLE (定義されていない)

この例では、明示的に定義されている指示はないので、どのサービスの状態も変更されません。

JASS_SVCS_ENABLE

Solaris 10 OS では、この変数には有効にする inetd の制御下にある SMF 対応サービスのリストが含まれています。このリスト上のエントリは、第 1 章の表 1-1 にリストされている FMRI の形式である必要があります。コード例 7-2 に、Solaris 10 OS を実行するシステムの rlogin を有効にするために user.init ファイルに追加するコード例を示します。このリストには、レガシーの SMF 以外のサービスを含めることもできます。これらが有効であるためには、サービスを JASS_ROOT_DIR/etc/inet/inetd.conf ファイルで定義する必要があります。定義されていない場合、エントリは無視されます。

コード例 7-2 rlogin の JASS_SVCS_ENABLE リストへの追加

```
if [ -z "${JASS_SVCS_ENABLE}" ];then
   if [ -f${JASS_HOME_DIR}/Drivers/finish.init ];then
      ./${JASS_HOME_DIR}/Drivers/finish.init
   fi
   JASS_SVCS_ENABLE="${JASS_SVCS_ENABLE} svc:/network/login:rlogin "
   export JASS_SVCS_ENABLE
fi
```

バージョン **10** よりも前の **Solaris OS** では、この変数には、システムで有効にする inetd サービスのリストを指定します。強化処理では、update-inetd-conf.fin 終了スクリプトによって、この変数で指定されている現在無効のサービスが有効になります。サービスがすでに有効になっている場合は、何も処理は行われません。同様に、監査処理では、update-inetd-conf.aud スクリプトによって、この変数で指定されているサービスがシステムで有効になっているか判定されます。デフォルトでは、この変数にはサービスは指定されていません。結果として、update-inetd-conf.fin スクリプトと update-inetd-conf.aud スクリプトの動作は、JASS_SVCS_DISABLE 変数の内容によってのみ制御されます。

JASS_TMPFS_SIZE

注 – システムの機能とシステムで実行されているアプリケーションに対して、現在および今後予想される /tmp ファイルシステムのニーズを満たすだけの十分な空間容量を確保するように、この変数を調整してください。

この変数には、/tmp (tmpfs) ファイルシステムに割り当てる空間容量を表す文字列値を指定します。強化処理では、set-tmpfs-limit.fin スクリプトによってこの設定がインストールされます。監査処理では、set-tmpfs-limit.aud スクリプトによってこの設定がチェックされます。この変数のデフォルト値は、512 MB です。

JASS_UMASK

この変数には、ファイル生成マスク (umask) を表す 8 進数の数値を指定します。強化処理では、set-system-umask.fin スクリプトと set-user-umask.fin スクリプトによってこの設定が使用されます。監査処理では、set-system-umask.aud スクリプトと set-user-umask.aud スクリプトによってこの設定がチェックされます。この変数のデフォルト値は 022 です。

JASS_UNOWNED_FILE

この変数には、システム上にある所有者のいないファイルのリストが格納されているファイルを示す文字列値を指定します。ファイルのユーザーまたはグループ割り当てが、システム上の有効なユーザーやグループと一致していない場合には、そのファイルは所有者がいないものと見なされます。強化処理では、print-unowned-objects.fin スクリプトでこの変数が使用されます。監査処理では、この変数は使用されません。デフォルトでは、この変数にはファイル名は割り当てられていません。その結果、print-unowned-objects.fin スクリプトの出力が画面に表示されます。

JASS_WRITABLE_FILE

この変数には、システム上にある world-writable ファイルのリストが格納されているファイルを示す文字列値を指定します。強化処理では、print-world-writable-objects.fin スクリプトでこの変数が使用されます。監査処理では、この変数は使用されません。デフォルトでは、この変数にはファイル名は割り当てられていません。その結果、print-world-writable-objects.fin スクリプトの出力が画面に表示されます。

JumpStart モード変数を定義する

JumpStart モード変数とは、Solaris Security Toolkit ソフトウェアが JumpStart モードで実行されているときにのみ、Solaris Security Toolkit ソフトウェアで定義および使用される変数のことです。この変数を使用すると、Solaris Security Toolkit ソフトウェアを JumpStart フレームワークとして使用することも、より大規模なビルド環境の中に組み込むことも容易になります。この JumpStart モード変数は JumpStart インストール時にのみ関連するため、この節で別途説明します。

JumpStart モード変数は、JASS_HOME_DIR/Drivers/user.init ファイルで定義されます。これらの変数は、変更せずに使用することができる他の大部分の変数とは異なり、通常は変更が必要となるため、user.init ファイルに置かれています。

注 – 場合によっては、マルチホーム JumpStart サーバーでのように、特別なカスタマイズが必要となることもあります。

Solaris Security Toolkit ソフトウェアを使用する環境に最も適するように、必要に応じてこれらの変数を調整してください。

ここでは、以下の JumpStart モード変数について説明します。

- 282 ページの「JASS_PACKAGE_MOUNT」
- 282 ページの「JASS_PATCH_MOUNT」

JASS_PACKAGE_MOUNT

この変数は、クライアントへのインストールが必要なソフトウェアパッケージが検索される名前付きリソースまたは場所を定義します。このリソースは、*host name*:*/path/to/software* という形式の NFS パスで定義されます。このリソースは、`driver.run` スクリプトの実行時に、`mount_filesystems` 関数によって `JASS_PACKAGE_DIR` にマウントされます。

このリソースの場所はホスト名または IP アドレスで指定する必要があり、実行中にディレクトリをマウントするのに必要な情報を NFS デーモンに提供するために、絶対パスがリストされていなければなりません。ホスト名や IP アドレスは環境変数の値に指定されることがあるため、頻繁に変更が必要となります。

Solaris Security Toolkit ソフトウェアでは、正しいホスト名とディレクトリパスの構成を自動的に試みます。ただし、この自動構成は使用環境に適用できない場合もあります。デフォルトでは、この変数は次の値に設定されています。

```
HOSTNAME:/jumpstart/Packages
```

`HOSTNAME` 変数は、クライアントがマウントした `/cdrom` ファイルシステムが格納されている NFS サーバーのアドレスに動的に割り当てられます。

JASS_PATCH_MOUNT

この変数は、クライアントへのインストールが必要なソフトウェアパッチが検索される名前付きリソースまたは場所を定義します。このリソースは、*host name*:*/path/to/patches* という形式の NFS パスで定義されます。このリソースは、`driver.run` スクリプトの実行時に、`mount_filesystems` 関数によって `JASS_PATCH_DIR` にマウントされます。

このリソースの場所はホスト名または IP アドレスで指定する必要があり、Toolkit 実行中にディレクトリをマウントするのに必要な情報を NFS デーモンに提供するために、絶対パスがリストされていなければなりません。ホスト名や IP アドレスは環境変数の値に指定されることがあるため、頻繁に変更が必要となります。

Solaris Security Toolkit ソフトウェアでは、正しいホスト名とディレクトリパスの構成を自動的に試みます。ただし、この自動構成は使用環境に適用できない場合もあります。デフォルトでは、この変数は次の値に設定されています。

```
HOSTNAME:/jumpstart/Patches
```

HOSTNAME 変数は、クライアントがマウントした /cdrom ファイルシステムが格納されている NFS サーバーのアドレスに動的に割り当てられます。

用語集

ここでは、Solaris Security Toolkit で使用されている略語と頭字語を一覧にまとめています。

A

ab2	AnswerBook2
ABI	Application Binary Interface (アプリケーションバイナリインタフェース)
ARP	Address Resolution Protocol (アドレス解決プロトコル)
ASPPP	Asynchronous Point-to-Point Protocol (非同期ポイントツーポイントプロトコル)

B

BART	Basic Auditing and Reporting Tool (基本監査報告機能)
BIND	Berkeley Internet Name Domain
BSD	Berkeley Software Distribution
BSM	Basic Security Model *(Solaris)*

C

CD compact disc (コンパクトディスク)

CD-ROM compact disc–read-only memory (コンパクトディスク読み取り専用メモリー)

CDE Common Desktop Environment (共通デスクトップ環境)

cp(1) ファイルコピーコマンド

cron(1M) クロックデーモンコマンド

D

DHCP Dynamic Host Configuration Protocol (動的ホスト構成プロトコル)

DMI Desktop Management Interface (デスクトップ管理インタフェース)

DMTF Distributed Management Task Force

DNS Domain Name System (ドメインネームシステム)

E

EEPROM electronically erasable programmable read-only memory (電気的に消去できるプログラム可能な読み取り専用メモリー)

F

FACE Framed Access Command Environment

FMRI Fault Management Resource Identifier (障害管理リソース識別子)

FTP File Transfer Protocol (ファイル転送プロトコル)

G

GID group identifier (グループ識別子)

GNOME GNU Network Object Model Environment (GNU ネットワークオブジェクトモデル環境)

GUI graphical user interface (グラフィカルユーザーインタフェース)

H

HSFS High Sierra File System (ハイシエラファイルシステム)

HTT HyperText Transfer (ハイパーテキスト転送)

HTTP HyperText Transfer Protocol (ハイパーテキスト転送プロトコル)

I

ID identifier (識別子)

IETF Internet Engineering Task Force (インターネット特別技術調査委員会)

Ilim Internet-Intranet Input Method

INETD Internet service daemon (インターネットサービスデーモン)

IP Internet Protocol (インターネットプロトコル)

IPF Internet Protocol Filter (インターネットプロトコルフィルタ)

ISA instruction set architecture (命令セットアーキテクチャー)

J

JASS JumpStart Architecture and Security Scripts (現在は Solaris Security Toolkit)

K

KDC　Kerberos Key Distribution (Kerberos 鍵配布)

L

LDAP　Lightweight Directory Access Protocol

`lp`(1)　ラインプリンタコマンド *(印刷要求の発行)*

M

MAN　management network (管理ネットワーク) *(Sun Fire ハイエンドシステムの内部 I1 ネットワーク)*

MD5　message-digest 5 algorithm (メッセージダイジェスト 5 アルゴリズム)

MIP　Mobile Internet Protocol (モバイルインターネットプロトコル)

MSP　midframe service processor (ミッドフレームサービスプロセッサ)

`mv`(1)　ファイル移動コマンド

N

NFS　Network File System (ネットワークファイルシステム)

NG　Next Generation (次世代)

NGZ　non-global zone (非大域ゾーン)

NIS、NIS+　Network Information Services (ネットワーク情報サービス)

NP　no password

NSCD　name service cache daemon (ネームサービスキャッシュデーモン)

O

OEM Original Equipment Manufacturer

OS Operating System (オペレーティングシステム)

P

PAM Pluggable Authentication Module

PDF Portable Document Format

Perl Practical Extraction and Report Language

PICL Platform Information and Control Library

PPP Point-to-Point Protocol (ポイントツーポイントプロトコル)

PROM programmable read-only memory (プログラム可能な読み取り専用メモリー)

Q

QA quality assurance (品質保証)

R

RBAC role-based access control (ロールベースのアクセス制御)

rc run-control (実行コントロール) (**ファイルまたはスクリプト**)

`rlogin`(1) リモートログインコマンド

RFC Remote Function Call

RPC Remote Procedure Call (遠隔手続き呼び出し)

`rsh`(1) リモートシェルコマンド

S

SA system administrator (システム管理者)

SC system controller (システムコントローラ) (*Sun Fire* ハイエンドシステムおよびミッドレンジシステム)

`scp`(1) 安全コピーコマンド (遠隔ファイルコピープログラム)

SCCS Source Code Control System (ソースコード制御システム)

SLP Service Location Protocol (サービスロケーションプロトコル)

SMA System Management Agent (システム管理エージェント)

SMC Solaris Management Console

SMF Service Management Facility (サービス管理機能)

SMS System Management Services (システム管理サービス)

SNMP Simple Network Management Protocol (簡易ネットワーク管理プロトコル)

SP service provider (サービスプロバイダ)

SPARC Scalable Processor Architecture

SPC SunSoft Print Client

SSH Secure Shell (*Solaris 9* および *10 OS*)

T

TCP Transmission Control Protocol

`tftp`(1) trivial file transfer program

`ttl` time-to-live (生存時間)

U

UDP User Diagram Protocol

UFS Unix File System (Unix ファイルシステム)

UID user identifier (ユーザー識別子)

UUCP UNIX-to-UNIX Copy (UNIX-to-UNIX コピー)

V

VOLD Volume Management daemon (ボリューム管理デーモン)

W

WBEM Web-based Enterprise Management

X

XFS X Font Server

索引

記号

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

く

け

こ

さ

し

す

せ

そ

た

ち

へ

ほ

ま

み

む

め

も

や

ゆ

ら

り

る

れ

ろ