Vol.2011-CSEC-53 No.13
Vol.2011-IOT-13 No.13
2011/5/13

SELinuxの不要なセキュリティポリシ削減手法の設計と評価

矢儀真也[†1]　中村雄一[†2]　山内利宏[†1]

SELinuxのセキュリティポリシは設定が難しいため，汎用的なポリシを利用することが多い．しかし，このポリシは，個々のシステムに必要のない権限を許可している可能性がある．また，ポリシが占有するメモリ使用量が多く，組み込み機器には適していない．これらの問題への対処として，不要なポリシを自動で検出し，削減する手法を提案する．提案手法は，SELinuxが出力するログを利用して不要なポリシを検出する．また，システム管理者にポリシの修正を提案し，システムのセキュリティを向上させ，ポリシのメモリ使用量を削減できる．本論文では，SELinuxのポリシの問題点と対処方法を示し，設計と評価について報告する．

Design and Evaluation of a Method to Reduce Redundant Security Policy of SELinux

Shinya Yagi,[†1] Yuichi Nakamura[†2]
and Toshihiro Yamauchi[†1]

In many cases, general security policy is used because of the difficulty of creating security policy. However, this security policy is possible to allow excessive rights in system. In addition, it is difficult to use this security policy in embedded systems because of the memory footprint. To deal with these problems, we propose a method system automatically detects redundant security policies by using log SELinux outputs and deletes them. The proposed system also suggests system administrator and improves security of the system and reduces the memory footprint. This paper shows the problems of security policy and dealing with them. This paper also shows design and evaluation.

†1 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University
†2 日立ソリューションズ
Hitachi Solutions

1. はじめに

ソフトウェアの脆弱性を利用した様々な攻撃により，被害が発生している．その中でもゼロデイ攻撃は，被害が大きくなる可能性が高い．また，攻撃者が権限昇格によってroot権限を取得した場合は，Linuxにおいて，攻撃者が全ての権限を得るため，被害が大きい．

これらの問題を解決する手段として，Security-Enhanced Linux（以降，SELinuxと略す）[1]に代表されるセキュアOSの利用が挙げられる．

しかし，SELinuxのセキュリティポリシ（以降，ポリシと略す）には，設定の難しさ，最小特権実現の難しさ，およびメモリ使用量の問題がある．ポリシの設定が難しいため，自分でポリシを作成することなく，ポリシ開発者が配布しているポリシを利用する場合が多い．しかし，このポリシは利用しないデーモンやアプリケーションに関するポリシを含む汎用的なポリシであるため，個々のシステムには不要なポリシが含まれている可能性が高いことが挙げられる．また，配布されているポリシのメモリ使用量は5MBを超えるため，組み込み機器のようにメモリのサイズが限られる場合，利用が難しい．

これらの問題点のうち，設定の難しさとメモリ使用量は，SELinux Policy Editor（以降，SEEditと略す）[2][3]などのポリシを設定するツールにより対処されつつある．しかし，設定ツールを利用したとしても計算機システムの知識が必要であり，設定工数が多いため，ポリシの作成は簡単ではない．また，最小特権実現の難しさは文献4)で示されているものの，カーネルに修正を加えなければならず，実現されていない．

そこで，利用する計算機に含まれる不要なポリシを自動的に検出し，削除する手法を提案する．提案手法は，SELinuxが出力するログを利用して，不要なポリシを検出し，システム管理者にポリシの修正を提案する．システム管理者が修正を許可した場合，自動でポリシを修正する．これにより，最小特権を持つポリシの設定におけるシステム管理者の負担を軽減し，実行時にポリシが占有するメモリ使用量の削減を実現する．

2. SELinuxのセキュリティポリシと問題点

2.1 SELinuxのアクセス制御機構

SELinuxは，National Security Agencyを中心するコミュニティで開発されているセキュアOSである．セキュアOSとは，MACと最小特権(Least Privilege)を実現するOSま

Vol.2011-CSEC-53 No.13
Vol.2011-IOT-13 No.13
2011/5/13

たはカーネルモジュールである．MAC は，OS におけるアクセス権限の管理者が定めたポリシのもとで，全てのファイルやプログラムのアクセス権限が一元的に制御され，所有者が設定を変更できないアクセス制御である．最小特権は，サブジェクトに必要最小限のアクセス権を与えることができる機能である．SELinux は，Multi Level Security，Role Based Access Control，および Type Enforcement を実現している[5]．

2.2 SELinux のセキュリティポリシ

本論文では，利便性の高さから Reference Policy[6] を扱う．Reference Policy は，ポリシがモジュール化されており，ポリシの運用中でも，モジュール単位でポリシの追加や削除が可能である．Reference Policy は以下の 3 つのファイルから構成される．

(1) fc（file context）ファイル
(2) if（interface）ファイル
(3) te（type enforcement）ファイル

fc ファイルには，プロセスやシステム資源のパス名とラベルの対応付けを記述する．if ファイルには，te ファイルで使用するマクロを記述する．te ファイルには，アクセス権限の付与を記述する．te ファイルに記述するポリシを以下に示す．

policyrule subj_t obj_t:tclass{av};

マクロを使用しない場合は，ポリシルール（policyrule: allow，auditallow，dontallow，neverallow），ドメイン（subj_t），タイプ（obj_t），オブジェクトクラス（tclass），およびアクセスベクタパーミッション（av）で構成される．ポリシルールのうち，allow は許可を意味し，auditallow はアクセスを許可した際に出力するログ (以降，許可ログと略す) の出力を意味する．ドメインはプロセスのラベルであり，タイプは操作対象となるファイルやネットワーク等のシステム資源のラベルである．オブジェクトクラスとは，例えばファイルの場合は，ファイル，ディレクトリ，パイプのように，オブジェクトの種類を分類するものである．アクセスベクタパーミッションとは，read や write のようなアクセスパーミッションであり，オブジェクトクラスごとに定義されている．

2.3 SELinux のポリシの問題点

2.3.1 ポリシの設定の難しさ

SELinux のポリシの設定の難しさは，以下の 3 つに分類される[7]．

(1) ラベル設定の難しさ

利用する計算機システムについて詳しく知らなければ，どのプログラムや資源にどのラベルを設定すべきか判断が難しい．

(2) パーミッション設定の難しさ

アクセス権限を与えるべき全てのドメインとタイプの組み合わせに対して，ルールを記述する必要がある．また，700 を超える種類のパーミッションから，適切なパーミッションを設定する必要がある．

(3) アプリケーションの振る舞いの理解が必要

アプリケーションがどのシステムコールを利用するかを知らなければならない．また，システムコールとパーミッションの対応付けも知らなければならない．

2.3.2 最小特権の実現の難しさ

SELinux のポリシが採用するホワイトリスト方式は，誤って必要以上の権限を与えることがある．必要以上の権限を与えてしまう原因の 1 つとして，配布されているポリシは，利用していないデーモンやアプリケーションに関するポリシを含んでいるため，個々のシステムに必要のない権限を許可している可能性がある．また，利用しているデーモンやアプリケーションでも，多くの環境で問題なく動作するように，多くの権限が与えられている．このため，動作しているシステムの必要最小限の権限とは差が生じる．

2.3.3 メモリ使用量

配布されているポリシのメモリ使用量は，Fedora 13 の時点で約 5.6MB ほどであり，増加傾向にある．組み込みのようにメモリが限られている場合には利用が難しい．

3. SELinux の不要なセキュリティポリシ削減手法の設計

3.1 設 計 方 針

SELinux のポリシの問題点を解決するために，不要なポリシ削減手法を提案する．提案手法は，システムが自動で不要なポリシを発見し，修正することで，ポリシを最小特権に近づけ，ポリシのメモリ使用量を減らす．以下に, 本システムの設計方針を示す．

(1) ポリシが必要か不要かの判断はシステム管理者が行う．
(2) ポリシの修正時に，システム管理者がポリシを削除する際の判断を支援する．
(3) 誤って必要なポリシを削除した場合，できるだけ早くポリシの復元を提案する．
(4) カーネルに修正を加えない．

提案手法では，不要なポリシを検出してもすぐに修正せずに，ポリシの修正をシステム管理者に提案するにとどめ，最終的にはシステム管理者に判断を委ねる．また，誤って必要なポリシを削除してしまった場合に備えて，ポリシのバックアップをとり，復元する機能を備える．これにより，システムの信頼性を高める．さらに，カーネルに修正を加えないことによ

Vol.2011-CSEC-53 No.13
Vol.2011-IOT-13 No.13
2011/5/13

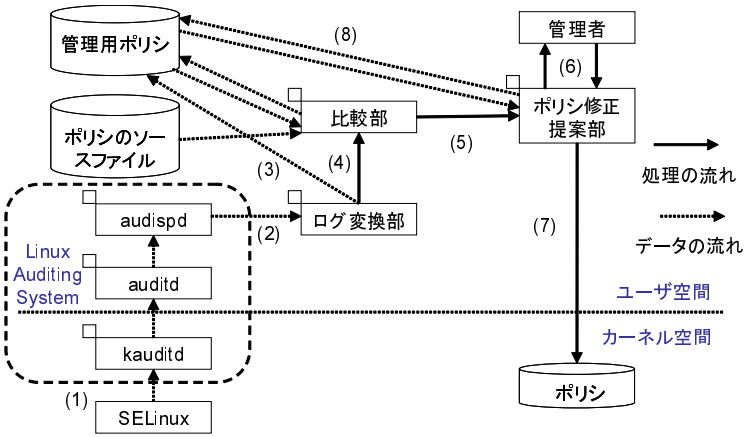

図 1 システムの全体像

り，導入時の手間を省くだけでなく，汎用性を高める．

3.2 設計内容

3.2.1 基本構成

本手法では，SELinux が出力するログを利用して不要なポリシを発見し，取り除く．また，誤ってポリシを削除した場合，削除したポリシを復元する．本手法の全体像を**図 1** に示し，以下で不要なポリシを削除する場合と，誤って削除したポリシを復元する場合について述べる．図 1 の番号は，不要なポリシを削除する場合に対応している．

図 1 の管理用ポリシとは，利用されたポリシや削除したポリシなどの情報を保存しているファイルのことである．ログ変換部，比較部，およびポリシ修正提案部については後述する．

不要なポリシを削除する場合

（1） SELinux がログを出力
（2） ログ変換部が，Linux Auditing System の一部である audit dispatcher daemon（以降，audispd と略す）からログを受信
（3） ログ変換部がログをポリシの形式に変換し，管理用ポリシに保存
（4） (2)，(3) を一定時間繰り返し，ログを収集した後，比較部を起動
（5） 比較部が，ログから作成したポリシとポリシのソースファイルを比較し，差分のポリシを作成した後，ポリシ修正提案部を起動
（6） ポリシ修正提案部が管理者に差分のポリシの内容を通知し，ポリシの修正を提案
（7） 管理者がポリシの修正を許可した場合，ポリシ修正提案部がポリシを修正し，システムに反映
（8） ポリシ修正提案部が修正したポリシを管理用ポリシに保存

誤って削除したポリシを復元する場合

（1） SELinux が拒否ログを出力
（2） ログ変換部が audispd から拒否ログを受信し，ポリシに変換後，比較部を起動
（3） 比較部が，拒否ログから作成したポリシと管理用ポリシを比較し，以前に削除したポリシと一致するかを調査
（4） 一致した場合，比較部がポリシ修正提案部を起動し，ポリシ修正提案部が管理者にポリシの復元を提案
（5） 管理者がポリシの修正を許可した場合，ポリシ修正提案部がポリシを修正し，システムに反映
（6） ポリシ修正提案部が修正したポリシを管理用ポリシに保存

3.2.2 ログ変換部

ログ変換部は audispd からログを受信し，ログをポリシに変換するデーモンである．本構成部の処理の流れを述べる．

（1） audispd からログを受信
（2） ログをポリシの形式に変換し，管理用ポリシに保存
（3） ログが拒否ログだった場合，比較部を起動

本構成部を実現するための課題と対処を以下に示す．

(課題 1) ログを出力させる方法

不要なポリシを発見するために，許可ログ，またはアクセスを拒否した際に出力されるログ（以降，拒否ログと略す）を出力させる必要がある．そこでログを出力させる方法として auditallow 文を利用した．

(課題 2) ログを出力する量を制限する方法

全ての許可ログを出力させた場合，ログの出力によるオーバヘッドが大きく，audit の設定によっては，計算機が正常に動作しなくなる問題がある．また，カーネル空間からユーザ空間にログを転送する際，一定時間に出力するログの量が，ログを一時的に保管する領域である backlog を超えるとユーザ空間にログを転送する前に，ログが消失してしまう問題がある．そこで，全てのモジュールに auditallow 文を適用せずに，調査

Vol.2011-CSEC-53 No.13
Vol.2011-IOT-13 No.13
2011/5/13

したいモジュールのみに適用し，ログの出力を制限する．

(課題 3) ポリシを復元する契機

誤って削除したポリシの復元は，システムが知り得た早い段階で行ったほうがよい．そこで，拒否ログが出力されたとき，以前削除したポリシかどうか判定を行い，復元を行う．このタイミングであれば必要なアクセスを 1 回拒否するだけで，影響は小さいといえる．また，一定時間経過したときにまとめてポリシを復元することもできるようにする．

3.2.3 比 較 部

比較部は，ログから作成したポリシと，ポリシのソースファイルを比較し，差分のポリシを作成する．比較部は 2 種類の比較を行う．1 つ目は，不要なポリシを削除するための比較である．2 つ目は誤って削除したポリシを復元するための比較である．本構成部の処理の流れを述べる．

(1) 不要なポリシを削除するための比較
  (a) ログ変換部が一定時間ログを収集した後，比較部を起動
  (b) 管理用ポリシにある許可ログから作成されたポリシとポリシのソースファイルを比較し，差分を作成
  (c) 不要なポリシが存在した場合，管理者に提示する情報を管理用ポリシに保存した後，ポリシ修正提案部を起動

(2) 誤って削除したポリシを復元するための比較
  (a) 拒否ログが出力されたとき，ログ変換部が比較部を起動
  (b) 拒否ログから変換したポリシと管理用ポリシを比較し，以前削除したポリシと一致するものがあるか探索
  (c) 以前削除したポリシと一致した場合，管理者に提示する情報を管理用ポリシに保存した後，ポリシ修正提案部を起動

本構成部を実現するための課題と対処を以下に示す．

(課題 4) モジュール単位でポリシを比較し，修正する方法

モジュールを構成するソースファイル（.te，.if，.fc）は m4 マクロ[8] を利用している．3 つのソースファイルのうち，.te ファイルと.if ファイルはこのマクロを全て展開しなければポリシの比較に必要な情報を得ることができない．

そこで，次の手法で対処した．モジュールのソースファイルからロード可能なモジュールに変換する際，＜モジュール名＞.tmp というファイルを経由する．このファイルは，.te ファイルと.if ファイルのすべての m4 マクロを展開し，1 つのファイルにしたものである．このファイルを利用して，モジュール単位でポリシの比較と修正を行う．

表 1 管理者が行える操作

操作	内容
java modify_policy -a ＜モジュール名＞	モジュールを調査対象に追加
java modify_policy -d ＜モジュール名＞	モジュールを調査前の状態に復元
java modify_policy -e ＜モジュール名＞	モジュールの調査を終了し，修正
java modify_policy -f	調査済みのモジュールのリストを表示
java modify_policy -l	調査中のモジュールのリストを表示
java modify_policy -m	修正済みのモジュールのリストを表示

3.2.4 ポリシ修正提案部

ポリシ修正提案部は，ポリシの修正の提案や，ポリシの修正を行う．また，管理者がコマンドを入力することで，ポリシの状態の確認，ポリシの修正ができる．本構成部の処理の流れを述べる．

(1) 管理者にポリシの修正を提案
(2) 管理者が修正を許可した場合，ポリシのソースファイルを修正し，システムに反映
(3) ポリシを修正した内容を管理用ポリシに記述

表 1 に，管理者が行うことができる操作を示す．また，本構成部を実現するための課題と対処を以下に示す．

(課題 5) ポリシの削除，復元を提案する際に提示する情報

本手法では，ポリシの削除，復元を管理者に提案する際に，管理者がポリシの変更をするかどうかを判断するために有益な情報を提示する必要がある．ポリシの削除を提案する際に提示する情報とポリシの復元を提案する際に提示する情報を以下に示す．

(1) ポリシの削除を提案する際に提示する情報

module=⟨ モジュール名 ⟩ subj=“関連するサブジェクト” obj=“関連するオブジェクト” allow subj_t obj_t:tclass{av};

(2) ポリシの復元を提案する際に提示する情報

subj=“サブジェクトのフルパス” obj=“オブジェクトのフルパス” syscall=システムコール名 allow subj_t obj_t:tclass{av};

3.3 期待される効果

本手法を適用することで，期待される効果を述べる．

Vol.2011-CSEC-53 No.13
Vol.2011-IOT-13 No.13
2011/5/13

(1)　最小特権に近づけること
不要なポリシを削除することにより，必要以上のアクセス権限を与えずにすむ．これにより，最小特権に近づけることができる．
(2)　管理者の負担の軽減
SELinux を利用しているシステムの管理者は，どのポリシが必要かを判断しなければならない．しかし，判断するための情報が少ないという問題がある．また，ポリシの構造が複雑なため，不要なポリシがわかりにくいという問題がある．このため，不要なポリシの削除を提案することで，管理者の負担を軽減できる．
(3)　メモリ使用量の削減
不要なポリシを削除することにより，カーネルにロードするポリシが減るため，メモリ使用量を減らすことができる．

4. 評　　価

4.1 目的と評価環境

本評価の目的は，提案手法を適用した場合のポリシの削減量とオーバヘッドを定量的に示すことである．また，ポリシの削減をする際，システム管理者にかかる負担を従来の方法と比較して評価する．カーネルは Linux 2.6.34.6-54.fc13.i686.PAE（Fedora 13），CPU は Intel(R) Pentium(R) 4 CPU 2.80GHz，メモリは 512MB，ポリシのバージョンは selinux-policy-targeted-3.7.19-62.fc13 という環境で評価を行った．

4.2 ポリシの削減量

ポリシのサイズ，ラベルの数，モジュールの数，および allow 文の数を評価する．ポリシのサイズはメモリ使用量の削減と対応しており，ラベルの数，モジュールの数，および allow 文の数は最小特権に近づけることに対応している．

Reference Policy のモジュールには，base モジュールとその他のモジュールがある．base モジュールは，システムに必須のモジュールであり，容易に修正を行うものではないため調査対象から除外する．本評価では，base モジュールを除いたその他のモジュール全てを調査対象とする．本評価は，モジュールを調査対象に追加した後，計算機を再起動し，2 日間ログを収集した後，評価を行った．計算機は，HTTP，FTP，ファイル共有，メールサーバ，および DNS が動作しているサーバである．

表 2 に評価結果を示す．allow 文の数から，本環境ではポリシの約 8 割は実際には利用されていないことがわかる．また，モジュールの数から，モジュールの 9 割以上が利用されていないことがわかる．これは，利用されていないデーモンやアプリケーションのモジュールが多く含まれていたからだと考えられる．

表 2　ポリシの削減量

	デフォルト	提案手法適用後	減少量 (%)
ポリシのサイズ (B)	5,852,591	1,074,081	81.8
ラベルの数	3,083	1,417	54.0
モジュールの数	223	19	91.5
allow 文の数	271,296	49,289	81.8

表 3　ポリシの削減量と内訳

	削減量	モジュールの削除による減少	モジュール内のポリシの削除による減少
ポリシのサイズ (B)	4,778,510	4,455,414	323,096
ラベルの数	1,666	1,666	0
allow 文の数	222,007	195,167	26,840

表 3 にポリシの削減量と内訳を示す. 表 3 から，以下のことがわかる．ポリシのサイズはモジュールの削除や，利用しているモジュール内の不要なポリシの削減により減少した．ラベルの数は，ラベルを定義しているモジュールの削除により，減少した．allow 文は，モジュールの削除や，利用しているモジュール内の不要なポリシの削減により減少した．

以上のことから，提案手法は利用しているモジュールに含まれるポリシと，利用していないモジュールに含まれるポリシをどちらも削減でき，ポリシを最小特権に近づけることができたといえる．また，ポリシのメモリ使用量も大幅に削減できた．

4.3 オーバヘッド

下記の 4 つの条件でデーモンの 1 つである abrtd の再起動を 10 回行い，再起動の平均時間を計算することにより，提案手法のオーバヘッドを測定した．
(条件 1)　提案手法適用時，abrt に許可ログを出力させる
(条件 2)　提案手法適用時，abrt に許可ログを出力させない
(条件 3)　提案手法非適用時，abrt に許可ログを出力させる
(条件 4)　提案手法非適用時，abrt に許可ログを出力させない
(条件 1) は，abrtd に対応するモジュールである abrt を調査対象に加えた状態を指す．(条件 2) は，調査対象のモジュールがない状態を指す．(条件 3) は，提案手法を適用せずに abrt に許可ログの出力設定をした状態を指す．(条件 4) は，デフォルトの状態を指す．

Vol.2011-CSEC-53 No.13
Vol.2011-IOT-13 No.13
2011/5/13

表 4 提案手法のオーバヘッド

	提案手法適用時	提案手法非適用時
許可ログの出力あり	(条件 1) 0.477s	(条件 3) 0.344s
許可ログの出力なし	(条件 2) 0.229s	(条件 4) 0.227s

表 4 に，測定結果を示す．(条件 1) と (条件 4) の差である 0.250 秒 (110%) が，提案手法のオーバヘッドである．これは，許可ログの出力によるシステムコール回数の増加，ログ変換部までのログの送受信時間，およびコンテキストスイッチの発生が影響していると考えられる．

(条件 1) と (条件 3) の差である 0.133 秒 (39%) が，ログ変換部が audispd からログを受信する際に発生するオーバヘッドである．ログの送受信によるオーバヘッドや，コンテキストスイッチの発生が影響しているものと考えられる．

(条件 2) と (条件 4) の差である 0.002 秒 (1%) が，許可ログの出力設定をしていない際のオーバヘッドである．ほとんど差がない理由は，ログを受信していないとき，ログ変換部はログの受信待ちの状態であるため，性能に影響を与えないためであると考えられる．

以上のことから，オーバヘッドは許可ログを収集している時だけに発生する一時的なものであり，実用に耐えうる程度であることがわかる．

4.4 システム管理者への負担

従来の方法と提案手法の，不要なポリシを発見し，削除する手順を以下に示し，システム管理者の負担を比較する．従来の方法では，システム管理者がモジュールのソースファイルを参照して，不要なポリシを発見する必要がある．ポリシは m4 マクロによって記述されていて，理解が難しいため，ポリシの開発者と同程度の知識が必要となる．また，ポリシのソースファイルはモジュール単位で記述されている．このため，モジュールの数に比例して，参照するポリシのソースファイルが増えるため，全てのモジュールの調査に，多くの時間を費やす必要がある．それに加えて，削除したポリシが本当に不要だったという判断が難しいという問題もある．以下に提案手法の利用手順について述べる．

(1) modify_policy コマンドを使い，1 つ以上のモジュールを調査対象に追加する．
(2) システムが自動で不要なポリシを発見し，ウィンドウを表示する．
(3) システム管理者が，3.2.4 項の情報をもとに，必要なポリシかどうかを判断し，Yes, または No のボタンをクリックする．
(4) Yes をクリックした場合は，システムが自動でポリシを修正し，適用する．No をクリックした場合は何も行わない．

提案手法では，システムが自動で調査対象のモジュールの不要なポリシを発見して，システム管理者に提案する．システム管理者は提案された情報をもとに判断をするだけであるため，負担が少ない．以上のことから，提案手法はシステム管理者の負担を軽減できるといえる．

5. おわりに

SELinux のアクセス制御で利用しているポリシには，ポリシの設定の難しさ，最小特権の実現の難しさ，およびメモリ使用量の問題があることを述べた．これらの問題を解決するために，SELinux が出力するログに着目して，利用されていないポリシを削減する方法を提案し，設計と評価結果について述べた．評価では，配布されているポリシの約 8 割は，評価環境では利用されていないことを示し，検出したポリシを削減することで，最小特権に近づけることと，メモリ使用量の削減を実現した．また，不要なポリシの検出，提案，および削除を自動化することで，システム管理者の負担を軽減できることを示した．

残された課題として，ポリシの削減量とオーバヘッドの詳細な評価が挙げられる．

参考文献

1) NSA: Security-Enhanced Linux. http://www.nsa.gov/research/selinux/
2) Project, S. P.E.: SELinux Policy Editor. http://seedit.sourceforge.net/
3) Nakamura, Y., Sameshima, Y. and Yamauchi, T.: SELinux Security Policy Configuration System with Higher Level Language, *Journal of Information Processing*, Vol.18, pp.201–212 (2010).
4) 山口拓人，中村雄一，田端利宏：SELinux の不要なポリシの削減手法の提案，情報処理学会研究報告，2008-CSEC-40, pp.37–42 (2008).
5) 海外浩平：Linux のセキュリティ機能：2．SELinux のアーキテクチャとアクセス制御モデル，情報処理学会会誌，Vol.51, No.10, pp.1257–1267 (2010).
6) Technology, T.: SELinux Reference Policy. http://oss.tresys.com/projects/refpolicy
7) 中村雄一，山内利宏：Linux のセキュリティ機能：3．セキュリティポリシー設定簡易化手法，情報処理学会会誌，Vol.51, No.10, pp.1268–1275 (2010).
8) Project, G.: GNU m4. http://www.gnu.org/software/m4/m4.html