社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－DSM－40 (1)
2006／3／29

ブラックリストを用いたPAM遅延モジュールによるSSHへの攻撃抑制

鈴木 聡† 湯浅 富久子†

インターネットからSSHへのパスワード攻撃が頻発している。OS非依存の対策は有効なものが少なくパスワード認証の禁止が一般的である。当機構では共同利用機関の必要性から幅広いユーザ層を抱えておりユーザ機器にカーネルの更新やパスワード認証禁止の徹底は難しい。ローカルもしくはDNS上のブラックリストを参照して遅延を発生させブルートフォース攻撃を妨害するPAMを作成したのでその有効性について述べる。

Prevension of SSH password cracking by the "pam_bldelay" based on blacklists

Soh Y. Suzuki† and Fukuko Yuasa†

Today, SSH is widely used as a remote access method. Instead of the commonness, SSH is continually attacked by the Brute Force password cracking. There are several security measures to the cracking, but OS-independent antidotes such as the password prohibition are not suitable for our inter-university research institute corporation. As OS-independent solution, we built the PAM module "pam_bldelay", that makes a certain delay according to the blacklist on the local system or the DNS record. We report the effectivity of the module.

1. 動　機

現在ではリモートアクセスの手段としてSSHが広く使われている。昨年は外部からアクセスできるSSHサーバに対して辞書攻撃を用いてアカウントとパスワードの総当たりを行なって侵入しようとする攻撃が非常に多かった。

攻撃されるマシンの正規利用者数が少ない場合は

- パスワード認証を禁止する
- アクセス元のIPアドレス範囲を限定する
- ポート番号を22から変更する
- ポート22に対しての頻繁な接続を拒否する

などの対策が有効である。当機構では計算科学センターがログインサーバを運用しているが、これを利用するには共同利用者としての登録が必要なため研究者のグループが独自に運用しているログインサーバが多数存在している。これらの研究グループ管理のサーバの場合、管理者がいずれの対策も採用したがらない事例が散見される。研究グループ管理のサーバの場合、管理者の立場は利用者より弱く利便性を損うことが許されなかったり、管理者本人が利便性を優先しているからである。

以下に各対策が損う利便性について述べる。

パスワード認証を禁止し、公開鍵認証を強制する場合は事前に公開鍵を作成して可搬端末とログインサーバ側の両者に設定をしておかなくてはならず、所外に出る前にこれを忘れるとログインできなくなる。管理者自身も緊急時呼出の場合に動きがとれないことを危惧し、公開鍵生成の手間と可搬端末への設定を利用者に徹底できず、一律禁止に踏みきることは難しい。

アクセス元IPアドレスの制限については出張が多い利用者はどのアドレスが現地で支給されるか不明なため、攻撃があってからでないとフィルタできない。しかし現在攻撃の主流は動的割当ての空間からであるため、事前の封鎖が難しい。

ポート番号の変更についても出張先でポート22、80、443以外は外にアクセスを許さない等のフィルタが掛けられていることがあるのでこれも忌避されがちである。

頻繁な接続を拒否する方法については非常に有効かつ一般利用者に対して損う利便性もない。ただし、すべてのOSで使用できるわけではないこととkernel-2.2系やRedHat9以前のディストリビューションをアプリケーションサーバとして提供している場合にはこの方法も適用できない。

最後の一つ以外はOS非依存であるが、不測の事態が発生した場合の身動きがとれないことが導入を躊躇わせる原因である。本稿では

- OS・カーネルを選ばず
- 利用者に対してパスワード認証を禁止せずに

SSHのアカウントに対する攻撃を抑制する方法とし

† 高エネルギー加速器研究機構
High Energy Accelerator Research Organization

グループ名	呼び出す場面 (例)
account	期限切れ等で禁止されてないか
authentication	本人確認
password	パスワード更新
session	セッション開始・終了処理

表 1　PAM の手続きグループ

て、Pluggable Authentication Modules (PAM) 用の遅延モジュールを作成し認証試行頻度を下げる方法を提案する。

2. PAM

PAM とはユーザーアカウントの有無、パスワードや権限情報等を検査する関数をモジュール化したものである。

古典的な UNIX では getpwent 等を通じて`/etc/passwd`や`/etc/shadow`等のデータを生々しくアクセスしていた。NIS、LDAP、RADIUS などの認証サーバ経由の集中管理の方式が一つ増えるたびにこれらの関数が増えるのはアプリケーション側の対応が非常に面倒である。そこで API を規定しておき認証機構がこの API に沿った関数を集めてダイナミックリンクモジュールとして提供しておけばその認証機構の差異を意識することなくアプリケーションからの使用が可能となる。

アプリケーションはこれらのモジュール中の関数を直接呼び出すことはせず、PAM ライブラリの関数を呼び出し、その正否を認証結果として取り扱う。

PAM では認証の手続きを 4 つのグループ (表 1) に分類している。PAM ライブラリ中にはアプリケーション用に各グループに対応する関数が用意されている。アプリケーションはそれぞれの場面に応じて PAM ライブラリ関数を呼出す。PAM ライブラリ関数は設定ファイルに応じてモジュールを読み込み、逐次処理を行なう。アプリケーションは各モジュールの逐次処理について知ることはない。

PAM による認証手続きでは複数のモジュールを順次適用していくことが可能である。順次適用にあたっては個々のモジュールの認証結果によって分岐できる (図 1)。分岐およびモジュールの指定は PAM の設定ファイルに記述でき、アプリケーションが意識する必要はない。設定中の分岐指定は required,requisite,sufficient の 3 つである。required もしくは requisite 指定のモジュールは認証に成功した場合は後続のモジュールに判断が引き継がれる。失敗した場合、結果は必ず拒否となる。失敗時に requisite は後続のモジュールを呼び出さずに即座に拒否で終了する。required の場合は後続のモジュールを呼び出し、失敗したことは最後のモジュールを呼び出すまでアプリケーションに伝わらない。なお、required 指定モジュールが失敗した後に

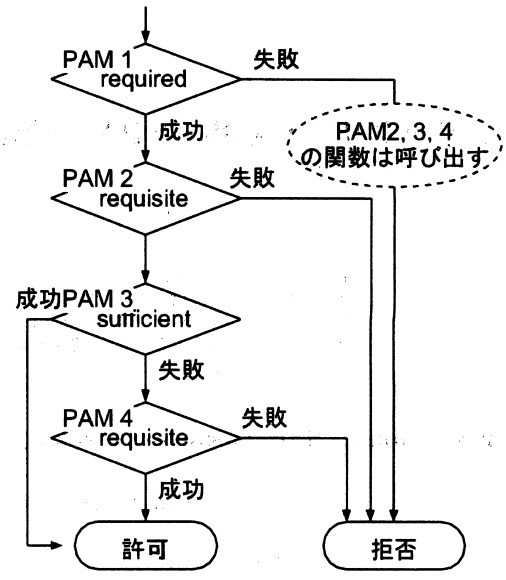

図 1　PAM 認証モジュールの逐次適用

```
# service group      control_flag module_path
sshd      auth       sufficient   pam_skey.so
sshd      auth       sufficient   pam_opie.so
sshd      auth       required     pam_unix.so
sshd      account    required     pam_unix.so
sshd      password   required     pam_permit.so
sshd      session    required     pam_permit.so
```

表 2　PAM 設定ファイルの例 (FreeBSD-4)

sufficient 指定モジュールが成功しても拒否の判断が覆ることはない。一方 sufficient 指定のモジュールが認証に成功すると即座に許可で終了し、後続のモジュールは呼び出されない。事前に required なモジュールが失敗している場合、sufficient が成功しても後続のモジュールは呼び出される。sufficient 指定のモジュールで失敗した場合は後続のモジュールに判断が引き継がれる。

また、これら規則はアプリケーション別、グループ別に個別に指定される。表 2 のような PAM 設定ファイルの場合、`pam_opie` や `pam_skey` はパスワード認証時には使用されるが、アカウントの禁止判定等の時点では使用されない。PAM モジュールを自作する場合、適用したいグループの関数を用意するだけで十分である。

今回開発するモジュールは単に適当な遅延を発生した後、常に認証に失敗する。このモジュールを sufficient 指定で組込むことによって認証の頻度を制限することができる。認証に失敗しても sufficient 指定なので後続のモジュールによって認証が継続され完全に

拒否されることはない。手で入力して失敗している程度の頻度であれば問題にならないような遅延とする。

この遅延時間を認証の失敗回数に応じて増やし、辞書攻撃のように頻繁なアクセスによってパスワードを破る攻撃を妨害する。

3. SSH の PAM 呼出

PAM 機構はアプリケーションが明示的に対応していないと使用できないので、あらゆるプログラムで即座に使用できるわけではない。RedHat Linux や FreeBSD ではベンダが PAM を有効にした OpenSSH[2] を提供しているが、自分でビルドする場合は明示的に PAM を有効にする指定が必要である。また、OpenSSH は 1.2 以降 PAM に対応しているが、PAM と併用した場合の脆弱性が報告されているものがあるので、注意が必要である。(openssh.com のアナウンス[3])。

SSH は PAM や通常の UNIX パスワード認証以外に公開鍵による認証機構を自前で備えており、これらが

(1) 公開鍵認証
(2) SSH 本体の持つパスワード認証
(3) PAM による認証

の順に適用される (図 2)。(2) の SSH 本体がもつパスワード認証機構は`/etc/passwd` もしくは `/etc/shadow` 中のパスワードとの合致だけを検査し、すぐに認証の可否を返すので遅延を挟むことができない。(2) は設定ファイル `sshd_config` 中に `PasswordAuthentication no` を記述することで禁止でき、パスワード認証を強制的に PAM 経由にすることができる。

(1) の公開鍵認証によるログインが成功した場合は PAM 認証の関数は呼ばれないため、今回作成したモジュールによる遅延は発生せず、従前の使い勝手が保障される。

4. pam_bldelay

SSH の PAM 呼出の振舞いを利用し、パスワードに対するブルートフォース攻撃に対してのみ遅延を発生させるような PAM モジュール "**pam_bldelay**" を作成した。

状況によらず一定の遅延を発生させる pam_delay[4] やランダムな遅延を発生させる Linux-PAM 中の pam_delay が既に存在しているが、これらを利用すると正規のユーザーに対しても遅延が発生するので極端に長い遅延を強制することができない。

本モジュールはブラックリストに認証試行回数を記録し、試行回数に応じた遅延を発生させる。ブラックリストは接続を試みたアクセス元の IP 毎に記録され、認証に成功した際に回数を 0 にリセットされるのでブルートフォース攻撃を掛けたホストに対してのみ遅延が発生する。

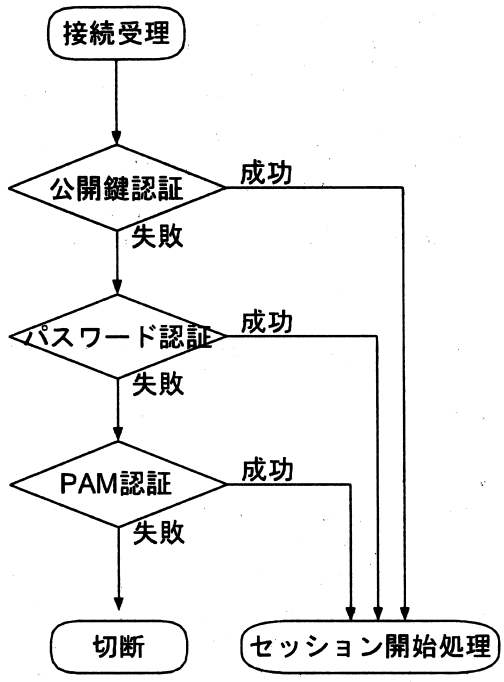

図 2 SSH での PAM 呼出は一番最後になる

また、攻撃を中断しただけでは認証試行回数は 0 にリセットされないため、期間をおいて再開した場合でも遅延は増大する一方である。

モジュールの導入に必要な手順は

- コンパイルとライブラリパスへのコピー
- sshd_config の修正 3 個所および再起動
  - `UseDNS no`
  - `UsePAM yes`
  - `PasswordAuthentication no`
- pam.conf の修正

である。

pam.conf 等の設定ファイル中の auth グループの先頭に pam_bldelay を指定するだけである (図 3)。sshd の再起動等は sshd_config の変更を反映させるためにのみ必要で、PAM 設定ファイルや pam_bldelay の更新・導入に際しては不要である。

作成にあたっては Linux-PAM 中に含まれる pam_deny を雛形とした。ローカルのブラックリスト機能だけに限定すれば 300 行程度の小さなモジュールである。

4.1 ブラックリスト機能

pam_bldelay のブラックリストはローカルのものと DNS 上のものを参照することができるが、先にローカルシステム上のものについて述べる。

(FreeBSD-4:/etc/pam.conf)

```
sshd auth sufficient pam_bldelay.so
sshd auth sufficient pam_skey.so
sshd auth required   pam_unix.so  try_first_pass
```

(Linux および FreeBSD-5 以降:/etc/pam.d/sshd)

```
auth sufficient /lib/security/pam_bldelay.so
auth required   /lib/security/pam_stack.so \
                              service=system-auth
auth required   /lib/security/pam_nologin.so
```

図 3 設定ファイルへの追加

PAM の認証モジュールにはログイン要求時にアプリケーションが

- ログインしようとしているユーザー名
- リモートホスト名
- リモートユーザー名

の情報を渡す。SSH が PAM を呼び出す場合、リモートユーザー名は常に空であり、ユーザー名とリモートホスト名のみが記録されている。リモートホスト名については `sshd_config` 中の `UseDNS` を `No` にすることによって IP アドレスの形で取得することができる。

ローカルシステム中にブラックリストとしてこのリモートホスト名を持つファイルを作成し、その中に認証試行の回数を記録する。この PAM モジュールは呼ばれる毎にファイルの内容を更新し、認証試行回数分だけの遅延を発生させる。このカウンタは呼出の時点で増加するので、攻撃者が認証の反応が遅いのをみて途中で接続を切断しても遅延は必ず増える。

次に認証に成功した場合にこのブラックリストをリセットすることについて考える。SSH が使用する PAM 関数は

- `pam_authenticate`
- `pam_setcred`
- `pam_open_session`
- `pam_close_session`

であり、authenticate と setcred の 2 つは auth グループに属する。authenticate で一連の検査をした結果が成功であればさらに続けて auth に羅列されているモジュール内の `setcred` に対応した関数が呼び出される。authenticate に成功した場合にのみ呼び出されるので該当モジュール以外の認証の成否を知ることができる。

これを利用して `pam_setcred` 中でカウンタをリセットすることにすれば、当該 PAM モジュール以外によって認証が成功した際に遅延が 0 に戻され、正常な利用をつづけることができる。`pam_open_session,pam_close_session` でも同様の動作が実現できるが、グループが auth ではなく session であるため設定ファイルの変更が増える。今回は

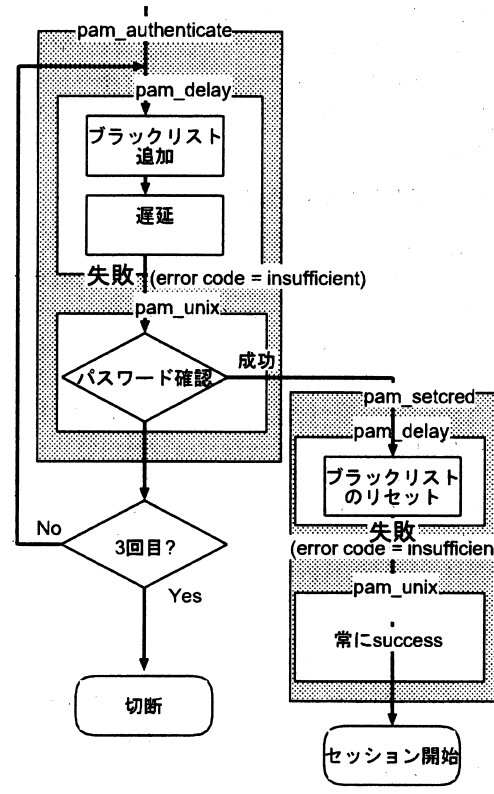

図 4 pam_authenticate と pam_setcred の呼出の流れ

auth グループに追加するだけで済む方がよいと考え、setcred 中でリセットを行なうことにした。

4.2 DNS ブラックリスト

現在見受けられる攻撃では 1 攻撃元からの試行回数によらず、遅延が 30 秒を越えると同じホストからのアクセスが止むことから、複数のサーバで情報を共有することは有効だと考えられる。そこで、前述のカウンタの値を DNS に展開して情報を共有することにした。

DNS 上のデータを扱うにあたり、データの取得には gethostbyname()、データの更新には Dynamic DNS update を使用している。

カウンタ値の取得は接続元の IP アドレスにサフィックスドメインを付加して A レコードを検索することによって行なう。たとえば 12.34.56.78 からの接続に対しては 12.34.56.78.local.domain.sshbl. の A レコードを検索し、127.0.0.N が返された場合、N をブラックリストのカウンタの初期値とする (図 5)。サフィックスドメイン名の local.domain.sshbl. は PAM モジュールに対する引数で設定する。おとりホストなどに対する攻撃から DNS 上のデータを更新して一般のホストに対する攻撃の閾値を上げることができる。

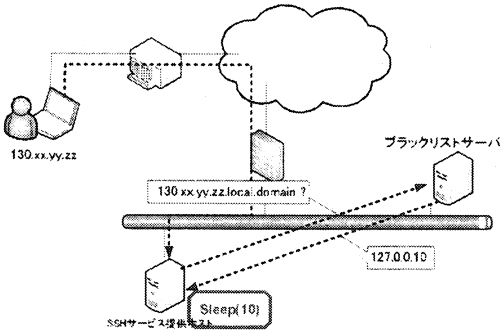

図 5 未知のアクセス元について初期値を DNS から参照する

このサフィックスドメインを管理する DNS サーバで Dynamic DNS update を受付けるようにしておくことにより、DNS 上のカウンタも自動的に増加させることができる。ただし、DDNS の update のパケット自体は容易に偽造できる内容であり、通常の update と同様に UDP を使用した場合はさらに送信元 IP も偽ることが可能であるため、bind9 が実装しているように鍵署名を使用しない限り安全性は担保されない。送信元 IP アドレスの偽造を避けるため現段階では TCP による update 送信として実装しているが、DNS ブラックリストサーバに直接接続する必要がある。

5. 適用結果

実際に適用した場合のログを以下に抜粋する。

```
04:47:40 sshd(pam_bldelay) rhost=211.167.***.*** user=a
04:47:40 sshd(pam_bldelay) sleeping 1
04:47:43 sshd(pam_bldelay) rhost=211.167.***.*** user=b
04:47:43 sshd(pam_bldelay) sleeping 2
04:47:48 sshd(pam_bldelay) rhost=211.167.***.*** user=c
04:47:48 sshd(pam_bldelay) sleeping 3
04:47:54 sshd(pam_bldelay) rhost=211.167.***.*** user=d
04:47:54 sshd(pam_bldelay) sleeping 4
04:48:00 sshd(pam_bldelay) rhost=211.167.***.*** user=e
04:48:00 sshd(pam_bldelay) sleeping 5
04:48:08 sshd(pam_bldelay) rhost=211.167.***.*** user=f
04:48:08 sshd(pam_bldelay) sleeping 6
04:48:16 sshd(pam_bldelay) rhost=211.167.***.*** user=g
04:48:16 sshd(pam_bldelay) sleeping 7
04:48:26 sshd(pam_bldelay) rhost=211.167.***.*** user=h
04:48:26 sshd(pam_bldelay) sleeping 8
```

この攻撃者はユーザー名を変えた試行を行なってきているが、遅延が 10 秒を越えた時点でアクセスが跡絶えた。遅延に対する降伏値は攻撃パターン毎に異なり、別の攻撃では 30 秒まで試行してきた。

```
17:58:18 sshd(pam_bldelay) rhost=200.193.**.** user=root
17:58:18 sshd(pam_bldelay) sleeping 1
17:58:20 sshd(pam_bldelay) rhost=200.193.**.** user=root
17:58:20 sshd(pam_bldelay) sleeping 2
17:58:29 sshd(pam_bldelay) rhost=200.193.**.** user=root
17:58:29 sshd(pam_bldelay) sleeping 3
17:58:33 sshd(pam_bldelay) rhost=200.193.**.** user=root
17:58:33 sshd(pam_bldelay) sleeping 4
17:58:37 sshd(pam_bldelay) rhost=200.193.**.** user=admin
17:58:37 sshd(pam_bldelay) sleeping 5
17:58:43 sshd(pam_bldelay) rhost=200.193.**.** user=admin
17:58:43 sshd(pam_bldelay) sleeping 6
17:58:45 sshd(pam_bldelay) rhost=200.193.**.** user=root
17:58:45 sshd(pam_bldelay) sleeping 7
(中略)
18:00:27 sshd(pam_bldelay) rhost=200.193.**.** user=root
18:00:27 sshd(pam_bldelay) sleeping 28
18:00:36 sshd(pam_bldelay) rhost=200.193.**.** user=root
18:00:36 sshd(pam_bldelay) sleeping 29
```

この攻撃では前回とことなり、ほとんど root にたいして集中的に攻撃を行なってきていた。これらの結果を見る限りでは固定長の数秒程度の遅延を挟んだ程度では攻撃を諦めさせるに至らないと考えられる。

6. まとめ

SSH へのブルートフォース攻撃に対する対策として遅延を発生させる PAM モジュール "pam_bldelay" を開発した。PAM モジュールを用いる方法は OS および daemon の入替えを必要としないので導入に対する負荷が少なく、SSH の内蔵する公開鍵認証に対して影響がないので利便性に対する弊害も少ない。また、ブラックリストを用いることによって既存の pam_delay とは異なり、攻撃の深刻度に応じた遅延を挟んでサービスを提供することが可能になり、攻撃を効果的に抑制できている。

今後は DNSSEC に対応した DDNS update を実装し、安全な DNS ブラックリスト運用を可能にする予定である。

参考文献

1) http://www.kernel.org/pub/linux/libs/pam/
2) http://www.openssh.com/
3) http://www.openssh.com/txt/sshpam.adv
4) http://www-uxsup.csx.cam.ac.uk/~pjb1008/project/pam_delay/