実施報告

Proxy サーバの効果と運用上の設定

萩原 拓郎

Proxy サーバによってネットワークトラフィックを減少させ，WWW アクセスを高速化することが出来ることを説明し，WWW アクセスの高速化およびセキュリティ向上のために必要な WWW サーバの設定を説明する．また WWW サーバの設定検討のために必要な WWW サーバの動作状態を調査するための手法についても言及する．Proxy サーバとしては Squid ver2.1 を用いることとする．

キーワード： Proxy サーバ,セキュリティ,サーバ設定

1 はじめに

1．1 Squid について

Squid は米国 National Laboratory for Applied Network Research の Duane Wessels がボランティアグループをまとめて開発したオープンソースソフトウェアで，無料で利用できる Proxy サーバソフトである．[1]の URL からソースファイルをダウンロード可能であり，コンパイル，インストールして利用する．2000/03/02 現在の安定版の最新バージョンは 2.3 であった．

1．2 Proxy 利用によって得られる効果

HTTP Proxy とは WWW ブラウザからのデータ要求を受け付けてブラウザの代理として WWW サーバからデータを取得してブラウザへ渡す，HTTP の代理サーバである．

複数の WWW ブラウザが外部ネットワークの同一の URL を取得する場合，通常は同じデータを複数回外部ネットワークから受信することになる．

ところが WWW ブラウザと外部 WWW サーバとの間に共用のキャッシュが存在すると，最初の URL の取得の場合だけ外部ネットワークからデータを受信するが，残りの受信では WWW ブラウザはキャッシュからデータを受信するため外部ネットワークの通信を減らすことができる．この時のキャッシュにあたるのが Proxy である．

これにより WWW ブラウズの高速化および Proxy-WWW サーバ間のトラフィックを削減できる．

2 セキュリティ設定

2．1 外部ネットワークからの Proxy サーバの利用の禁止

Proxy サーバおよびネットワーク資源は通常外部ネットワークへ公開する必要がないので外部ネットワークからの Proxy サーバの利用を禁止しておく．禁止しないでおくと WWW サーバの不正利用のための踏台にされる危険がある．

Proxy サーバは WWW ブラウザの代理として WWW サーバへデータ要求を行うため，通常 WWW サーバでは(WWW ブラウザからではなく)Proxy サーバでユーザがブラウザを利用しているようにみえる．

WWW サーバへ不正な接続を行うために外部のコンピュータから自分の Proxy サーバを利用された場合，自分の Proxy サーバが犯人として疑われる可能性がある．この場合実際に接続を行った WWW ブラウザが外部ネットワーク上に存在していた場合は犯人の特定，再発防止が難しい．

(必要な squid.conf の設定項目例)

```
acl in src 192.168.1.0/255.255.255.0
acl all src 0.0.0.0/0.0.0.0
http_access allow in
http_access deny  all
```

2．2 クライアント情報の WWW サーバへの送信

Proxy サーバは WWW クライアントからの URL 要求が届いた際にキャッシュ内に当該データが存在しなければ WWW サーバへデータ要求を行う．この際に URL 要求を行った WWW クライアントのドメイン名または IP アドレスを WWW サーバにも送信するようにする．

このようにすることで Proxy サーバを経由して WWW サーバへ不正アクセスや障害があった場合などに，実際に URL 要求をしたクライアントマシンを特定したり問題解決のための手がかりとして利用できるようにする．

Squid は標準でこのようになっているので設定を行う必要はないが，WWW サーバ側では設定変更が必要になることがある．広く利用されている WWW サーバソフト Apache の場合，ログの記録形式を変更しないと Proxy サーバから送られてきた本当のブラウザのアドレス情報を捨ててしまうことになる．Apache 1.3.0 では以下のように修正を行う．

HAGIWARA Takuro
武蔵工業大学環境情報学部情報メディアセンター技術員

(必要な httpd.conf の修正項目)

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
                    ↓
LogFormat "%h %l %u %t \"%r\" %>s
           %b \"%{X-Forwarded-For}i\"" common
```

3 キャッシュヒット率の向上

3．1 キャッシュ容量拡大への問題点

Proxy を利用することにより WWW ブラウズを高速化し，外部ネットワークとのトラフィックを削減できるのは Proxy のキャッシュがヒットした場合だけである．Proxy の恩恵をより高めるためにはキャッシュのヒット率を向上させることが必要である．

キャッシュのヒット率を向上するためには Proxy のキャッシュ容量の拡大が必要になる．Proxy サーバへ容量の大きなハードディスクを接続してキャッシュとして利用することになるが，いくつか問題がある．

(1)メモリ要求量の増大

キャッシュ内の全データに関して，どの URL のデータであり，ファイル名は何であるか，などの情報はデータ要求に対する応答時間を短くするために仮想メモリなどのハードディスクではなく実メモリ上になければいけない．

キャッシュの増大にしたがってこのためのメモリ要求量は増えて，例えば Solaris2.5.1(SPARC)+Squid2.1 の組合せでキャッシュ 8Gbytes を利用した場合，実メモリ 100MBytes 程度が必要になり，キャッシュ容量を 1GByte 減らす毎に実メモリはおよそ 10MByte 少なくて済む．

このことからキャッシュを増やすためには多くの実メモリを Proxy サーバ上に必要になることがわかる．

(2)データ要求の集中

より大きなキャッシュディスクを持つ Proxy サーバにはより多くの WWW ブラウザからの多くのデータ要求が集中するようになる．一定時間内で単一の Proxy サーバが処理できるデータ要求数は，ハードディスクの読み書き速度，サーバのバス速度，ネットワーク帯域などにより限りがある．

Proxy サーバのように大量のファイルを持ちファイル要求に対する応答時間を短くしたいシステムのためには UNIX のファイルシステムは特別向いているわけではない．そこで単一の Proxy サーバの性能を高めるために専用のファイルシステムを持つ Proxy 専用のハードウェアを用いてもよい[1]．(こうした専用の Proxy サーバマシンであっても下記で述べるような ICP および HTTP を使った連携動作を行うことは可能である．)

(1)および(2)の問題を解決するためには，サーバ本体をより高い性能をもつものへ変更するという以外に，複数の Proxy サーバを連携させるという方法がある．

3．2 隣接サーバとの連携動作

複数の Proxy サーバが設置されている場合これらを連携可能である．複数の Proxy サーバを連携させると，WWW ブラウザからのデータ要求に対して，連携している Proxy サーバのキャッシュも使用してデータを渡せるようになる．

連携した Proxy のシステムでは，要求されたデータが Proxy サーバのキャッシュになかった場合，ネットワークを経由して連携している他の Proxy サーバ(隣接サーバ)へ当該データがキャッシュにないかどうか ICP[5][6]というプロトコルを使用して問い合わせる．

隣接サーバのキャッシュに当該データが存在した場合は WWW サーバではなく隣接サーバからデータを受け取って WWW ブラウザへとデータを渡す．

隣接サーバとの連携設定により，隣接サーバ同士は一つの大きなキャッシュを持つ一つの Proxy サーバのように振舞うことが出来る．これによりキャッシュのヒット率の向上が期待でき，外部との低速通信路のトラフィックを削減できる．

隣接サーバ同士を結ぶネットワークが高速であったり，外部ネットワークとの間のネットワークが低速であったりする場合特に有用である．

3．3 隣接サーバとの連携時に注意すべき設定

隣接サーバを持つ Proxy サーバの設定には注意すべき設定項目がある．

あるサーバのキャッシュにあるデータは，その隣接サーバによってあたかも自ら取得したデータであるかのように WWW ブラウザへ渡されるということである．これは次の例を考えることでこの問題を理解できる．

ある URL C はネットワーク A からはアクセスが許可されているがネットワーク B からはアクセスが禁止されているとする．ネットワーク A およびネットワーク B に Proxy サーバ(a)，(b)があり，それぞれのネットワークに存在するブラウザからのみデータ要求を受け付けるとする．

この段階では URL C へのアクセス制御は正常に適用される．Proxy 経由であるとなしに関わらずネットワーク A からはアクセス可能であり，ネットワーク B からはそうではない．

ところが一旦(a)と(b)が連携すると閲覧できる場合もあれば閲覧できない場合もあるという不安定なことになる．(a)が URL C のデータを持っていない場合，ネットワーク B からは URL C のデータを取得できない．(a)が URL C のデータを持っている場合はネットワーク B からでも(b)

経由で(a)からデータを取得できてしまう.

結果として WWW サーバ側で行っているアクセス制限が正しく機能しなくなってしまい問題である.

このようなことは，情報メディアセンタでも実際に発生した．横浜キャンパス内のみ接続を許可する URL が存在し，横浜キャンパス内へ設置した Proxy サーバを世田谷キャンパスの Proxy サーバと連携動作するように設定を変更した.(ここでは横浜キャンパスがネットワーク A，世田谷キャンパスがネットワーク B と考えられる．)

この問題を解決するために情報メディアセンタでは横浜キャンパスの Proxy サーバを変更した.

(1) 横浜キャンパス内の WWW データを連携サーバへは送らない
(2) 学内の WWW サーバへは連携サーバへ問い合わせず，自分でデータを取得する

(必要な squid.conf の設定項目例)

```
hierarchy_stoplist cgi-bin ?
hierarchy_stoplist domain.name
acl neighbor  srcdomain neighbor.domain.name
acl insrv dst 192.168.1.0/255.255.255.0
icp_access  allow in  !insrv
icp_access  allow neighbor !insrv
```

4 Proxy サーバソフトの詳細な動作状態の観察

Proxy サーバの動作を知ることは Proxy サーバを管理，設定するために必要なことである．上述したようにキャッシュ容量を大きくしすぎると実メモリの不足を招き，ユーザのアクセスの傾向によっては他のサーバとの連携にはほとんど利点がないかもしれない.

Proxy サーバの運用のためにはこうしたことを具体的に動作中のサーバやログのチェックにより改善していくことが必要である.

ここではサーバの動作状態を得るための幾つかの方法を紹介する.

4．1 CGI プログラム cachemgr.cgi

squid をコンパイルした際に src ディレクトリに同時に作成されるプログラムに cachemgr.cgi というものがある．このプログラムを任意の WWW サーバ(Squid と別のマシンで構わない)で CGI プログラムとして動作させることで以下のような Squid の内部情報をブラウザから確認することができる.

(1) データサイズや要求回数に対するキャッシュのヒット率(Proxy サーバがどの程度の割合で WWW ブラウズの高速化に役立っているか)
(2) 外部ネットワークとのネットワークトラフィック削減にどの程度役立っているか
(3) サーバのディスクキャッシュ使用量
(4) キャッシュ内のデータの寿命(どのくらいの期間データがキャッシュ内に保存されているのか)

その他にも非常にたくさんの情報を得ることができる.

(cachemgr.cgi の設定方法，アクセス手順)

Proxy サーバに作成された cachemgr.cgi を WWW サーバの CGI ディレクトリにコピーして通常の CGI プログラムと同様に実行できるようにする.

cachemgr.cgi はプログラムバイナリであるので，Proxy サーバと WWW サーバが異なるアーキテクチャのマシンである場合には WWW サーバでコンパイルし直す必要があることに注意.

Proxy サーバの設定を行い cachemgr 用ユーザ名およびパスワードを設定する．(cache_mgr に設定するメールアドレスのユーザ名部分がユーザ名として使用される)

(必要な squid.conf の設定項目例)

```
cache_mgr hostmaster@domain.name
cachemgr_passwd disable shutdown
cachemgr_passwd goodpasswd all
```

次にブラウザから WWW サーバ上の cachemgr.cgi へアクセスする．Proxy サーバのサーバアドレス，HTTP ポート番号，上記で設定したユーザ名とパスワードを入力するとメニューが表示される.

4．2 top

top は現在実行中の全プロセス情報を継続的に画面に表示するソフトウェアであり，こちらもフリーウェアである．様々なところからダウンロードできるが例えば Solaris 用であればバイナリパッケージを SunSite[3]からダウンロードできる.

top を用いることで Squid のプロセスが OS によって割り当てられているメモリ量や，ハードディスクアクセスの割合などを確認できる．UNIX では一般にプロセスサイズは大きくすることは可能であるが，小さくすることはできないので top によって表示される Squid の使用メモリ量はこれまでの最大サイズであるが，必要メモリ量の見積もりに十分役に立っている.

4．3 ログを扱うためのスクリプト

一定期間の利用状況を知るためにログファイルを処理することも必要である．こうした目的のために Squid のログを扱うためのスクリプトを作成した[4]．以下に Squid のログから統計情報を出力するための perl スクリプト squidlog.pl の出力を挙げる.

```
-squidlog.pl の出力-----------------------------
From:2000/1/19 0:1:16
To   :2000/1/26 0:1:1
アクセス総数 (回数／bytes)      :873811/3975288669

外部ネットワークからのアクセス数統計
有効アクセス数(Proxy)              :303377
ICP リクエスト数                   :284936
ICP リクエストヒット／ミス      :22930/262006/8.0%
隣接サーバオブジェクト要求         :18441
隣接オブジェクト要求ヒット／ミス :18441/0/100.0%
有効アクセス数(Proxy 以外)         :
無効アクセス数                     :1343
有効アクセス(Proxy 以外)比率(回数):  0.0%

内部 URL へのアクセス数            :44117
内部 URL データ転送量(bytes)       :351671440

内部ネットワークからのアクセス統計
キャッシュヒット (回数／bytes)  :213174/778742132
キャッシュミス (回数／bytes)    :310550/2756663484
キャッシュヒット率 (回数／bytes): 40.6%/ 22.0%
隣接ヒット (回数/bytes)            :/
隣接ヒット率 (回数／bytes)         :  0.0%/  0.0%
有効アクセス数合計 (回数/bytes) :523724/3535405616
無効アクセス数合計 (回数／bytes):1250/1719724
有効アクセス比率(回数／bytes)    : 99.8%/100.0%

NONE/400 : 1208
TCP_DENIED/403 : 1386
TCP_HIT/000 : 10
:

内部ネットワークからのアクセス数統計
133.78.109.x        :
133.78.110.x        :
133.78.111.x        :17580
133.78.112.x        :49760
:

2000 0 19 0 10 :  11.1/3/24/  0.0//11
2000 0 19 0 20 :  38.6/17/27/  0.0//8
2000 0 19 0 30 :  64.5/20/11/  0.0//6
2000 0 19 0 40 :  26.9/7/19/  0.0//2
:
-----------------------------------------------
```

スクリプト出力の最初のパートでは，外部ネットワーク，内部ネットワーク(キャンパス内)毎にアクセス数，キャッシュヒット数，キャッシュヒット率(回数，バイト)，無効なアクセス(許可されていない外部ネットワークからのアクセスの試み)などを列挙している．

"ICP リクエスト"，"隣接サーバオブジェクト要求"の項目では連携している他の Proxy サーバからの要求について，"隣接ヒット"の項目では自分が連携している他の Proxy へデータ要求をしたことについての統計情報を出力している．

2番目のパートではSquidがWWWサーバから受け取ったHTTP リプライ毎に回数を列挙している．

3 番目のパートでは内部ネットワークのサブネット毎にアクセス回数を列挙している．

4 番目のパートでは期間中の日時毎のキャッシュヒット率，キャッシュヒット数，アクセス数を列挙している．

Squid はキャッシュのために巨大なハードディスク容量を使用するが，大学のような組織では毎日のアクセス数も比較的多くなるためログファイルもかなり大きくなる．このファイルは様々なトラブルの際に調査資料になるため保存しておきたいが，ハードディスクを圧迫しないように工夫することが必要になる．このために一定期間毎にログを退避(ローテート)する sh スクリプト(newlog)を作成した．

このスクリプトは実行されると Squid プロセスにログをローテートするようにシグナルを送信し，ローテートされた古いログを gzip を使って圧縮する．これにより大量のログを比較的小さなディスク容量で保存することができる．

```
-(newlog)-------------------------------------
#!/bin/sh
LOGDIR=/array3/squid/logs
SQUID=/array3/squid
GZIP=/usr/local/bin/gzip

cd $LOGDIR
rm *.log.2.gz
mv access.log.1.gz access.log.2.gz
mv access.log.0.gz access.log.1.gz
mv cache.log.1.gz cache.log.2.gz
mv cache.log.0.gz cache.log.1.gz
mv store.log.1.gz store.log.2.gz
mv store.log.0.gz store.log.1.gz
$SQUID/bin/squid -k rotate
/usr/bin/sleep 200
$GZIP *.log.0
```

5 おわりに

Proxy サーバソフト Squid2.1 を使用してネットワークトラフィックを削減するために必要な設定について説明した.

運用していく上でセキュリティだけでなく動作状態の取得も必要であることを説明した.

参考文献

[1] http://www.squid-cache.org/
http://www.meisei-u.ac.jp/mirror/squid/ (国内ミラーサイト)

[2] Rousskov,A. Wessels,D. Chisholm,G. Newman,D: “Web アクセスをスピードアップ プロキシ・キャッシュ 7 製品をテスト”日経コミュニケーションズ, 第 310 号, pp.150-156, 2000

[3] http://sunsite.sut.ac.jp/

[4] http://www.yc.musashi-tech.ac.jp/~takuro/unix/squidlog.pl

[5] RFC2186
ftp://ftp.iij.ad.jp/pub/rfc/rfc2186.txt

[6] RFC2187
ftp://ftp.iij.ad.jp/pub/rfc/rfc2186.txt