広域分散環境における ssh を用いたシングルサインオン機構の実現

安藤 雅享 東京大学
田浦 健次朗 東京大学
近山 隆 東京大学

1 はじめに

近年のネットワークインフラの発達に伴い、クラスタやグリッドなどの複数の計算機を用いて大きな計算力を得る試みは一般的なものとなりつつあり、複数サブネットに属す地理的に分散した多数の計算機を利用する機会も増えている。しかしながらそれらは複数の管理者によって管理され、安全性などの問題から通信やログインなどに制限がある場合も多い。これらの制限を迂回する方法が用意されている場合も多いが、利用の手間を大幅に増大させている。既存の並列分散計算のツールは管理方針の変更を要求するものが多いために計算資源としての利用率を損ねている。

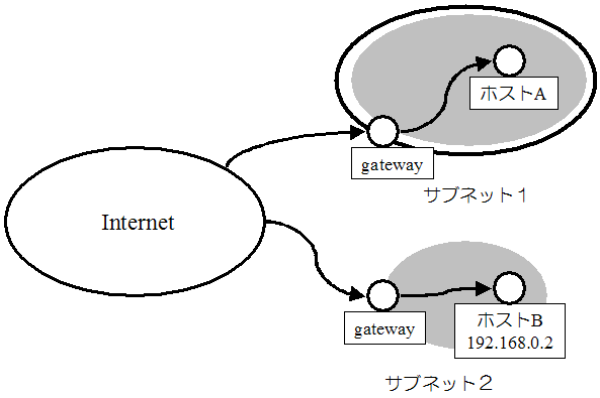

図 1: 複数サブネットを含むネットワーク

例として図 1 のように複数のサブネットに分かれている場合に目的のマシンにジョブを投入することを考える。例えばホストBはファイアウォールの内側に存在し、まずゲイトウェイとなるホストにログインしなければ外部からアクセスすることができない。また通信手段も ssh のみに制限されている場合なども考えられる。ホストCのようなプライベートアドレスしか持たないホストの場合も同様にまず目標の属するサブネット内に入らなければならず、アクセスに多段のログインを必要とする。

また、既存のツールではひとつひとつの計算機に事前にインストールや設定を行う必要があり、扱う計算機の数が増加するにつれこの手間は膨大なものとなってしまう。

本研究では、アクセスに多段のログインを必要とするホストを含んだ複数のホストにジョブを並列投入すること、及びそのための事前の準備を削減することを目的とする。

2 関連研究

グリッド環境を構築するための標準的なツールとなっている Globus [1, 2] は通信のためにいくつかのポートを必要とし、またプライベートアドレスなどには対応していない。

Resource Manager beyond Firewall (RMF) [3] はファイアウォール内の資源の利用を目的として Globus の Resource Manager を改良したものであるが、これもファイアウォール内側への通信用ポートと外側への通信は制限されないことを必要とする。

Virtual Private Network (VPN) はインターネット上に仮想的なプライベートネットワークを構築するものだが、このために管理者権限が必要となる。

3 ssh を用いたシングルサインオン機構

認証機構として広く用いられている ssh がすでに許可されている場合を考える。ssh で用いられる認証方法のうち公開鍵認証を使用し、ローカルで生成した公開鍵の登録は完了しているものとする。

認証鍵を保持する認証エージェントである ssh-agent を利用することによって、多段でないログインについてはユーザーはアクセスの度にパスフレーズを入力する必要がなくなる。

ファイアウォールなどの通信の制限を迂回するためには多段のログインが必要となるが、この際にはエージェント転送と呼ばれる機能を利用する。この機能は認証エージェントへの接続をリモートホストに転送するというもので、例えば図 2 のように認証エージェントへの接続がリモートホストBに転送され、ホストBからさらに先のホストCへログインしようとする場合には、転送された接続を通してローカルの認証エージェントが認証を行う。ホストCへログインするとホストCにもエージェントへの接続が転送される。

エージェント転送によって多段のログインにおけるシングルサインオン機構を実現することができるが、ssh とそのエージェントの実装が違うとエージェントを転送することができない場合がある。例えば OpenSSH 3.5p1

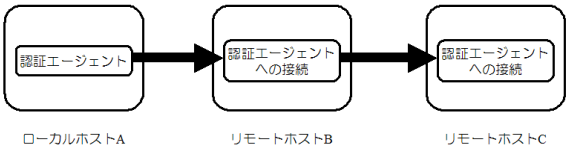

図 2: エージェント転送

[4] と SSH Secure Shell 3.2.2 [5] の間ではエージェント転送はできない。

エージェント転送ができない場合には ssh ポートフォワーディングを利用する。例えば図 3 のようにホストB（エージェント転送不可）の先のホストC（エージェント転送可能）にエージェントを転送したい場合、まずホストBにログインする。このときローカルホストAの未使用なポートXをホストCの ssh ポートに転送する。その上でこのポートフォワーディングを通じてローカルからホストCにログインすることによりホストCにエージェントが転送できる。エージェント転送できないホストが多く存在する場合でも、図 4 のようにポートフォワーディングを繰り返すことによって転送可能となる。

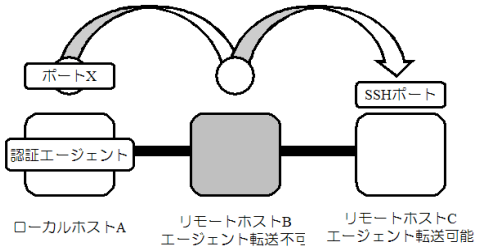

図 3: ポートフォワーディング

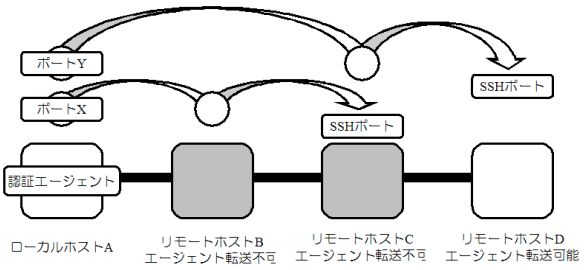

図 4: 多段ポートフォワーディング

ポートフォワーディングを用いる場合、ローカルホストの予めフォワードしたポートに対して ssh を行うことになるが、ホスト鍵ファイル中の localhost の鍵が参照されるため、予め localhost の鍵が登録されている場合や複数のホストに対してこの方法を用いる場合などはホスト鍵が違うことによって警告を受けてしまう。これを回避する方法としては、localhost に対する ssh にはホスト鍵認証を行わない設定をするというものがあるが、この方法ではセキュリティホールとなってしまう。そこでアクセスしたいホストに合わせてホスト鍵のエイリアスを設定する必要がある。

エージェント転送を用いず、ssh ポートフォワーディングを用いる方法のみでも多段のログインが必要なホストへのシングルサインオン機構を実現できるが、リモートホストの数が増えると無数のポートを使用しなければならない。このため、できるだけエージェント転送を用いるのが好ましいと考えられる。

4 並列なジョブの投入

この方式を用いて複数のホストにジョブを並列投入する方法の単純なものとしては、一つ一つのホストへのジョブ投入をバックグラウンドで行うという方法が考えられるが、この方法ではローカルホストがノードの数よりも多くのコネクションを同時に持たなければならないのでスケーラビリティに乏しい。

スケーラビリティを得るため通信は階層化し、各ノードでより先のノードへのジョブの投入と自身のノードへのジョブの投入及び出力を管理するプログラムを立ち上げる。ローカルホストには全ホストリストが存在し、どのように階層化しどのような経路で通信するかはローカルホストで決定される。各ノードで子ノードの通信に必要なホストリストを生成して送信する。このとき、管理プログラム自身をも送信することで事前の準備の手間を省く。このため管理プログラムはできるだけ環境に依存しないものである必要がある。

5 まとめと問題点

以上の方法のメリットは主に二点存在する。

- 多段のログインが必要な環境へのジョブの並列投入が可能。これにより ssh が許されているホストは利用可能となる。
- 事前の準備が最低限でよい。予めローカルで生成した公開鍵が登録してあれば、あとはホストリストなどの情報をローカルで設定するだけでよい。

問題点としては、通信が途切れた場合やクラッシュが発生した場合などの対処機構が存在しないことや通信経路の変化に動的に対応できないことなどの他、ひとつの認証エージェントに頼っているためスケーラビリティに乏しいという根本的な問題点が存在する。

また、今回は ssh が許されている場合のみを考えたが、他の通信手段を用いる場合も考えることが望ましい。

参考文献

[1] The Globus Project. http://www.globus.org/.
[2] I. Foster, C. Kesselman. “Globus: A Metacomputing Infrastructure Toolkit.” International Journal of Supercomputer Applications, 11(2):115-128, 1997.
[3] 田中 良夫, 平野 基孝, 佐藤 三久, 中田 秀基, 関口 智嗣 “Globus を用いたグローバルコンピューティング環境の構築とその評価” インターネットコンファレンス’99 論文集, pp.97-106, 1999.
[4] OpenSSH. http://www.openssh.com/.
[5] SSH Communication Security. http://www.ssh.com/.