9grid, cross domain auth, trouble ahead...

Steve Simon, steve@quintile.net

confidentiality, integrity, availability, authentication, authorisation/access control, auditability, nonrepudiation.

1. The grid

Plan 9 has long had the necessary features for grid computing but its promise has yet to be fulfilled. In recent years things have started to change but there still seems to be some confusion about what a grid should provide and how to implement it, particularly with respect to cross domain authentication. In this paper I shall attempt to contrast the different grid authentication models and expand on the other requirement of a grid computing environment.

2. What I want...

I want my isolated plan9 system to be able to join a larger grid of plan9 nodes, to access resources on those machines and to be able to offer services on my system to other grid users. This processes should be simple and quick, I should be able to control how much of my systems resources I offer so don't lose a significant level of service, and I don't want to be hacked or impersonated by other grid users. That is:

- Encrypted connections for all data
- Mutual authentication of all links
- A user account on any grid node infers an account on the whole grid
- Any node may chose to allow only an arbitrary subset of grid users access
- Resources offered to grid users should be configurable
- Adding an local authentication domain to the grid should be easy
- Allowing local users to access grid services should be trivial
- Allowing remote grid users access to local resources is a separate issue, it requires a human level Trust Relationship
- The adminstrator who gives a user an account remains responsible for that user and their actions on the grid
- Provide a reasonable degree of protection against malicious users.

I make a distinction between adding a user to an existing grid node and adding a new authentication domain (or authdom) to the grid - which infers adding all (or an arbitrary subset of) the grid's users.

3. Current status

At present the 9grid is a collection of disparate machines each with its local authentication server, recently two projects have attempted to combine these into a more global resource.

3.1. Centralised authentication

Here users may attach to any grid node authenticating against a single central authentication server, currently *sources* (outside.plan9.bell-labs.com). A modified factotum(1) is all that is required to allow users to participate in this grid.

Though all communications between the auth servers are encrypted the control of the central authentication database becomes significant. The administrative load of ensuring the identity of each grid user,

distributing passwords securely, dealing with abuse of the grid are all likely to become major issues.

Unfortunately since this work was done authentication at the *sources* server has become the exception rather than the rule, though authenticated accounts are still available users generally update their plan9 systems anonymously.

3.2. Distributed authentication

A mesh structure allows users who have an account in one 9grid authdom and to be given access to all peers authdoms, and this infers the user must have a home authdom where their account is controlled.

This has the advantage of pushing the requirements for user control (managing the Trust Relationship) to the manager of the authdom who introduces the new user. For example users in 9grid.trustworthy.com could be given local disk access and be put in the fossil group trustworthy, while the users authenticateing at 9grid.suspicious.com may be put in a group which only has read access to the /386 and /rc directories.

4. UID specification

Remote users may specify the authdom in which their account is to be validated when connecting to a remote host using a simple Email-like syntax - user@authdom, If the user is authenticated using the proposed distributed authentication mechanism the users UID would take the same form which prevents UID clashes between local and grid users.

Finer grained control over specific users in a remote authdom could still be exercised by editing /lib/ndb/auth, possibly limiting the percentage of the saturated CPU they may use [KL88], disk quotas, or network bandwidth limitations.

Allowing remote users to access local resources would require a secure channel between the local and remote authservs. The obvious way to implement this is using a shared secret between the two auth servers. Unfortunately, for a grid of **n** authdoms **2n** secret keys are required. These keys would have to be delivered securely between each pair of cooperating nodes and would have to be kept private.

5. Public Key Encryption

The classical solution to the problem of key distribution is to use public key encryption. If PKE were used then the grid of **n** authdoms requires only **2n** public key pairs, and only one key per host needs to be kept secret.

6. Example

The authdom plan9.cork.ie wishes to join the Irish plan9 grid. The system manager of plan9.cork.ie has already been given the ordinary user account called gridmaster.cork.ie on the existing grid node 9grid.dublin.ie, and this user is a member of the group gridmasters.

The following operations would be performed as the user bootes on console of plan9.cork.ie's auth server.

```
# create a temporary work area secure from prying eyes
cpu% ramfs -p
cpu% cd /tmp

# create our local grid control files and set their permissions
cpu% touch /adm/grid/keys /lib/ndb/grid
cpu% chmod 664 /adm/grid/keys /lib/ndb/grid
cpu% chgrp gridmasters /adm/grid/keys /lib/ndb/grid

# Generate a key pair and save them in bootes's factotum
cpu% auth/secstore -u bootes -g factotum -s auth.plan9.cork.ie
cpu% auth/rasgen -t 'authdom=plan9.cork.ie' > keypair
cpu% cat keypair >> factotum
cpu% auth/secstore -u bootes -p factotum -s auth.plan9.cork.ie

# Connect to our initial grid node
cpu% import -u 'user=gridmaster.cork.ie' 9grid.dublin.ie / /n/9grid.ie

# Publish our public key and auth server's name
cpu% auth/ras2pub keypair >> /n/9grid.ie/adm/grid/keys
cpu% echo 'auth=9grid.dublin.ie authdom=plan9.cork.ie' >> /n/9grid.ie/lib/ndb/grid

# Fetch the other 9grid authdom's public keys and auth server's names
cpu% cp /n/9grid.dublin.ie/adm/grid/keys /adm/grid/keys
cpu% cat /n/9grid.dublin.ie/lib/ndb/grid > /lib/ndb/grid

cpu% cd
cpu% unmount /tmp
```

The grid nodes 9grid.dublin.ie and plan9.cork.ie are now twinned and their users may connect to either machine - assuming the default authorisation files do not restrict this (see below). The files /lib/ndb/grid and /adm/grid/keys now need to be distributed across all grid nodes. This could be accomplished by a cron(1) job running a simple script to copy the files using rx(1) across the grid nodes.

7. Authorisation

So far we have covered authentication of grid users - the verification of the users identity. Authorisation - the control of the resources and operations that this user is allowed can access on this machine is a seperate issue.

At the simplest level authorisation could be controlled through entries in /lib/ndb/auth, though extra fields would probably have to be added, to allow remote users to be further constrained.

These constraints might include users limits on the cpu and network bandwidth by uid, the use of listen and dial, and the group membership they are assigned by fossil.

For example:

```
authdom=9grid.ie
  uid=friend cpu=80% group=users,sys expire=1/9/2007
  uid=suspicious cpu=10% net=10% dial=0 listen=denied group=noworld ns=/lib/namespace.limited
  uid=!hacker
  uid=* cpu=20% net=20% dial=17000-17090 listen=denied group=none ns=/lib/namespace.friends

authdom=*
  uid=* cpu=20% net=20% dial=17000-17090 listen=denied ns/lib/namespace.friends
```

If the majority of local users are not to be permitted access to the grid, for example in an undergraduate teaching environment, then the local environment could be split into two authdoms, only the one of which holds trusted (staff and postgrad) accounts being registered for membership of the grid.

Trust Relationships

The verification of identity and intent of new users requesting accounts on 9grid nodes is a fundamental problem, and must be solved locally at each authdom. Authdom administrators would be expected to revoke 9grid privileges for abusive users, and administrator who repeatedly allow abusive users access the 9grid could simply have their authdom’s key deleted, thus disconnecting them from the grid.

Should the 9grid become very widespread then it will almost certainty become a prime target of hackers and it must have the ability to deal with unfriendly users as they appear.

References

[KL88] J. Kay and P. Lauder. "A Fair Share Scheduler," Communications of the ACM, January 1988