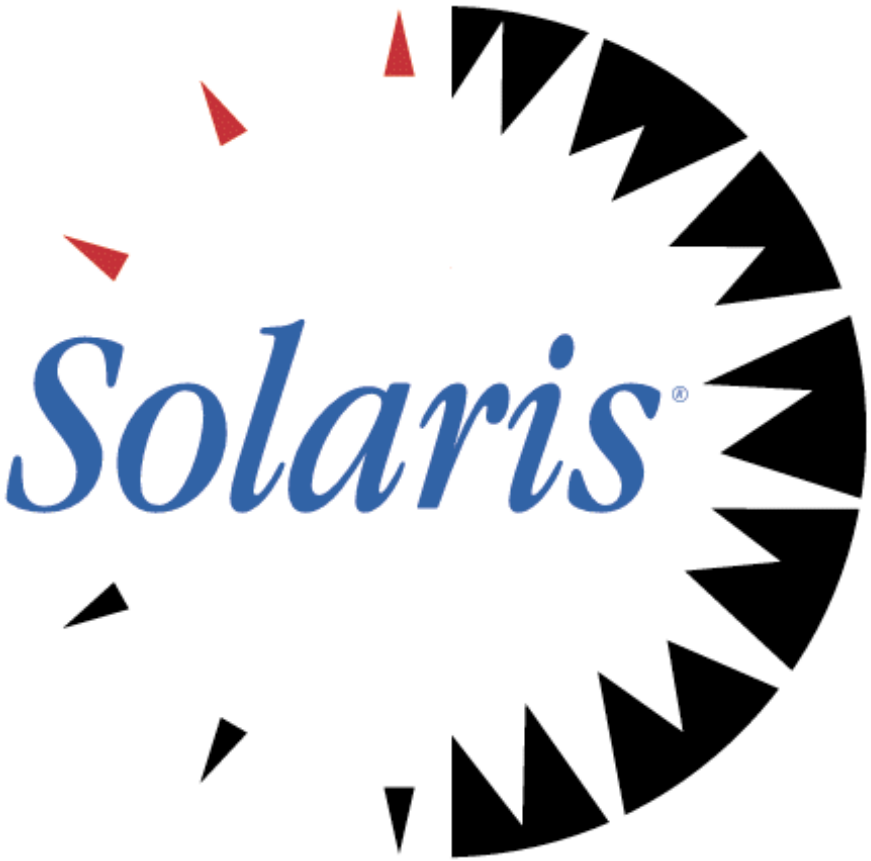

Wesley Chun & Alex Lei
Solaris Networking
Technologies

96

Feb 26 1996

Välkommen till
ようこそ
Bienvenue à
Welcome to
歡 迎
Benvenuti a
환 영
Willkommen zu
欢 迎



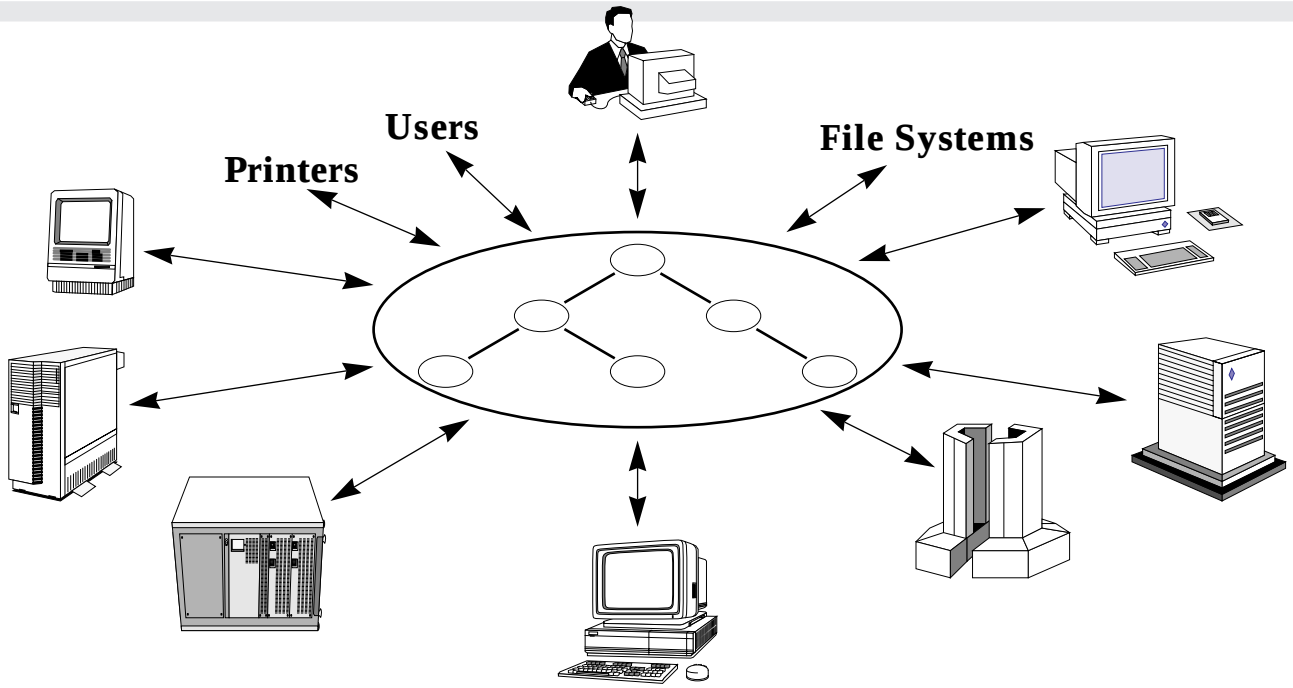

Feb 26 1996

# *Network Information Service Plus (NIS+)*

# *Agenda*

- **General**
  - **What is NIS+? (and what it isn't)**
  - **NIS vs. NIS+**
- **The Internals**
  - **NIS+ Objects**
  - **NIS+ Domains**
  - **Commands and API**
  - **Principals and Credentials**
  - **Scripts, Files, etc.**
- **The Future**
  - **Transitioning from NIS to NIS+**
  - **What's New in Solaris 2.5**

# *What is NIS+?*




- **Network information service for Solaris 2**
- **Replaces NIS/YP as default name service**
- **Provides speed, availability, scalability, and security**

# Notes

## What is NIS+?

- Component of SunSoft's Solaris 2 distributed system management suite, ONC+
- Designed to provide a secure, robust naming/directory service for today's enterprise client/server environments.
- The next generation of NIS/YP which is currently installed on over 3.1 million systems world-wide.

## ONC NIS

- Originally called "yellow pages" or "YP", ONC NIS was designed to address the administration requirements of the first generation of heterogeneous client/server networks.
- Replaced the UNIX `/etc` and corresponding configuration files for other operating environments with a central database.
- The database was replicated for lookup performance and reliability. Each network supported one NIS master server and a number of slaves. Database updates could only be made from the master server to maintain central control.

# *What NIS+ is not:*

- **A distributed relational database (call ORCL, SYBS, or IFMX)**
- **An extension of enhancement of NIS/YP**
- **A product only sold and supported by Sun**
- **A simple or unstructured product**
- **Impossible to learn and use**

# *NIS+ vs. NIS*

**The major advantages of NIS+ over NIS are:**

- **Security -- authentication and authorization of users**
- **Speed -- efficient, fast updates**
- **Scalability -- hierarchical domains**
- **Availability -- multiple servers provide fault tolerance**

# Notes

## NIS+ vs. NIS

- NIS+ was designed to replace NIS. Backward compatible so that existing NIS clients can receive services from NIS+ servers without knowing the difference. This provides a smooth transition path for any organization currently running NIS.
- Both have a basic function of providing network resource management.
- Security -- authenticate (verify) & authorize (permissions) clients, aka “principals.”
- Speed -- efficient, fast updates mean no more all0night (yp)pushes.
- Scalability -- hierarchical structure help manage and simplify network resources
- Availability -- Fault tolerance means more service for clients

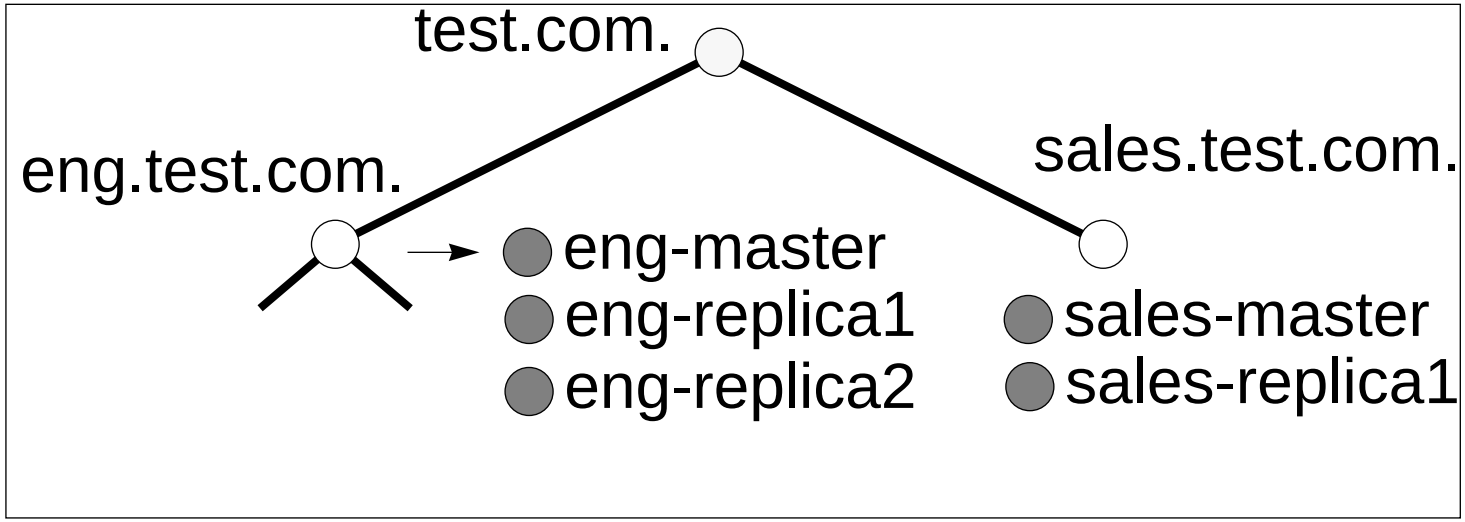
## NIS+ Features

- Streamline administration for range of networks - from very small to very large enterprise networks
- Reliable and consistent updating of NIS+ information

# *User Interfaces*

- **User and System Administration Commands**
  - **For accessing and managing network information resources**
- **Application Programmers' Interface (API)**
  - **For network application development**
- **SunSoft Solstice Enterprise Management Tools**
  - **GUI interface to aid in NIS+ administration (aka *admintool*)**
- **System administration shell scripts**
  - **For easy client and server installation and initialization**

# *Server Replication for Availability*



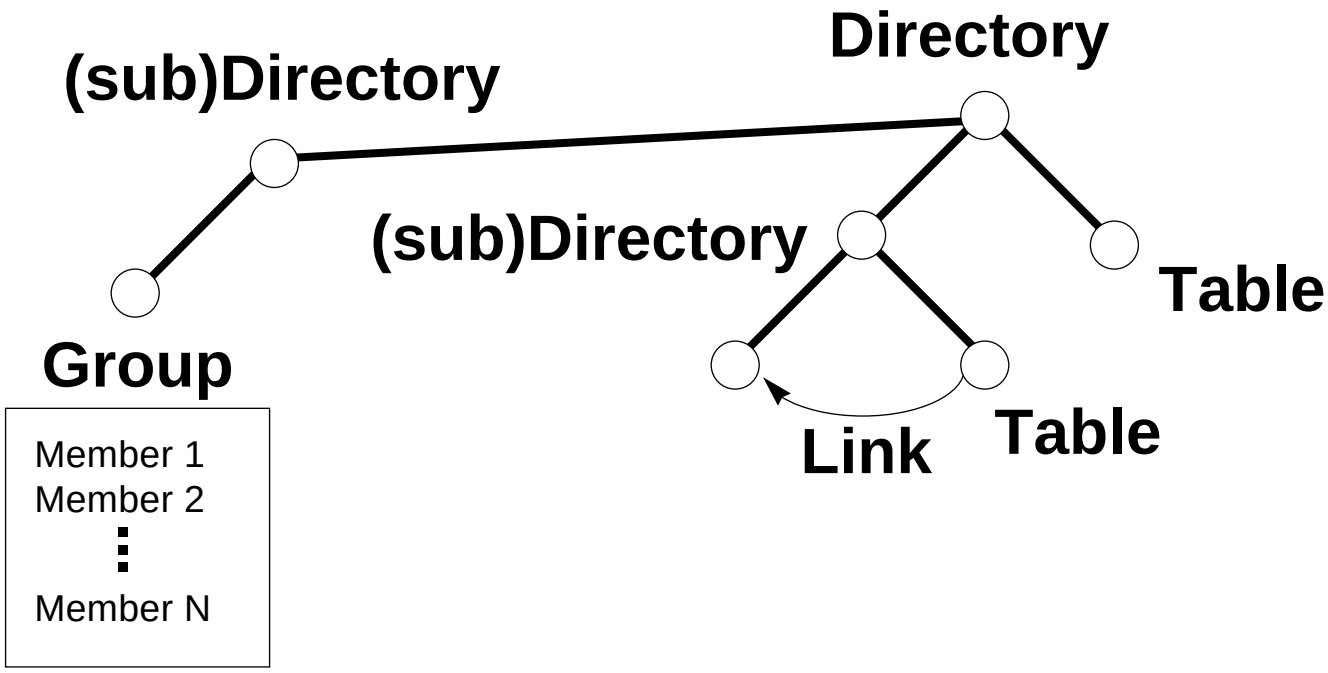

- **Multiple replicated servers for each domain**
  - **For improved reliability and performance**
- **One master server and multiple replica servers**
  - **Updates are applied to master and propagated to replicas**

# *The Internals*

# *NIS+ Objects*

- ***Directory* object**
  **Contain names, addresses, and authentication for domain servers**
- ***Table* object**
  **Primary NIS+ database storage media**
- ***Group* object**
  **Contains list of hosts and users who are members of a group**
- ***Entry* object**
  **Contains name of owner of row and a set of rights for row**
- ***Link* object**
  **Contains name of another object**

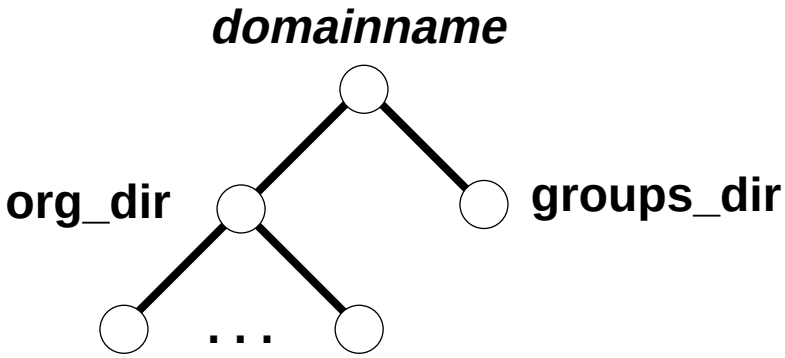
# *Sample NIS+ Hierarchy*



## NIS+ Objects

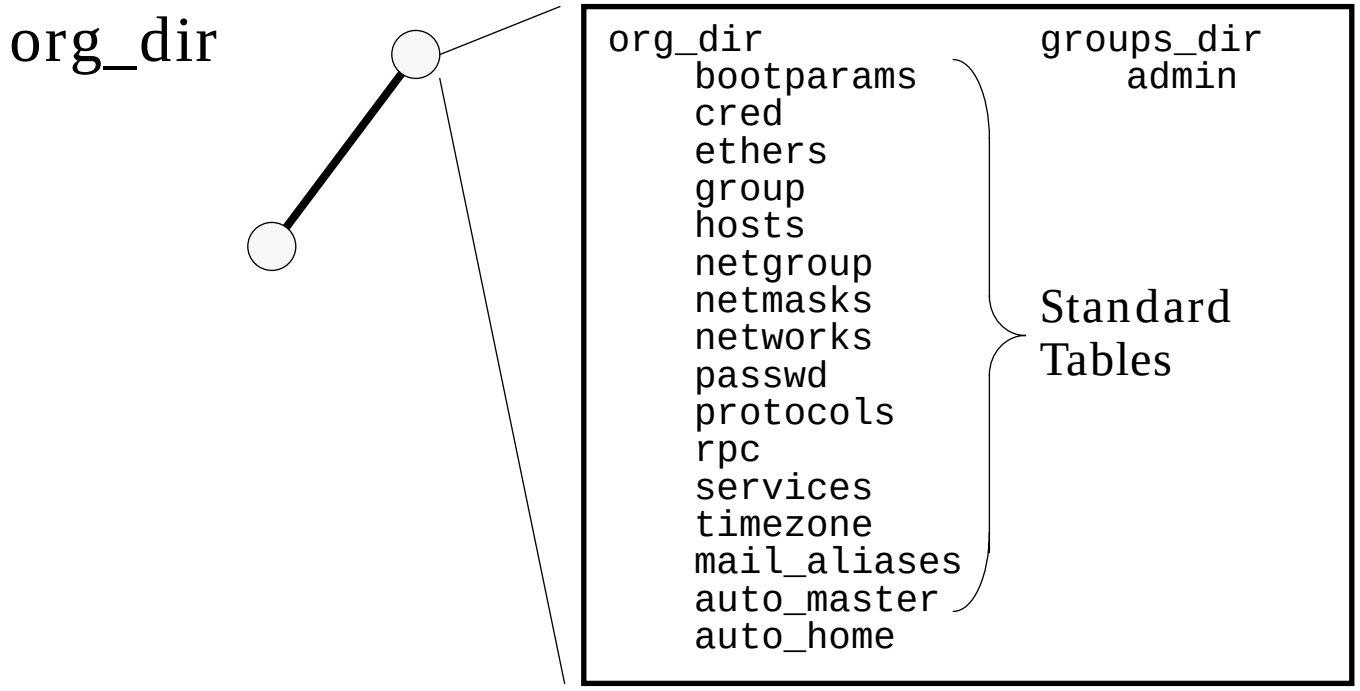
- NIS maps -> NIS+ tables because the model used is that of a columnar table. The table object specifies the number of columns in the table and identifies which columns can be searched with a NIS+ query.
- When a table lookup is performed, an entry object is returned. This entry data consists of the name of the owner for this row, a group owner, a time to live value for the row, and a set of access rights for this row.

# *NIS+ Domains*




- **NIS+ domains consist of three directory objects**
  - **Domain object**
  - **`org_dir.`*`domainname`*`.`**
  - **`groups_dir.`*`domainname`*`.`**
- **For example, the `test.com` domain consists of**
  - **`test.com., org_dir.test.com., groups_dir.test.com.`**

# Domain Directories



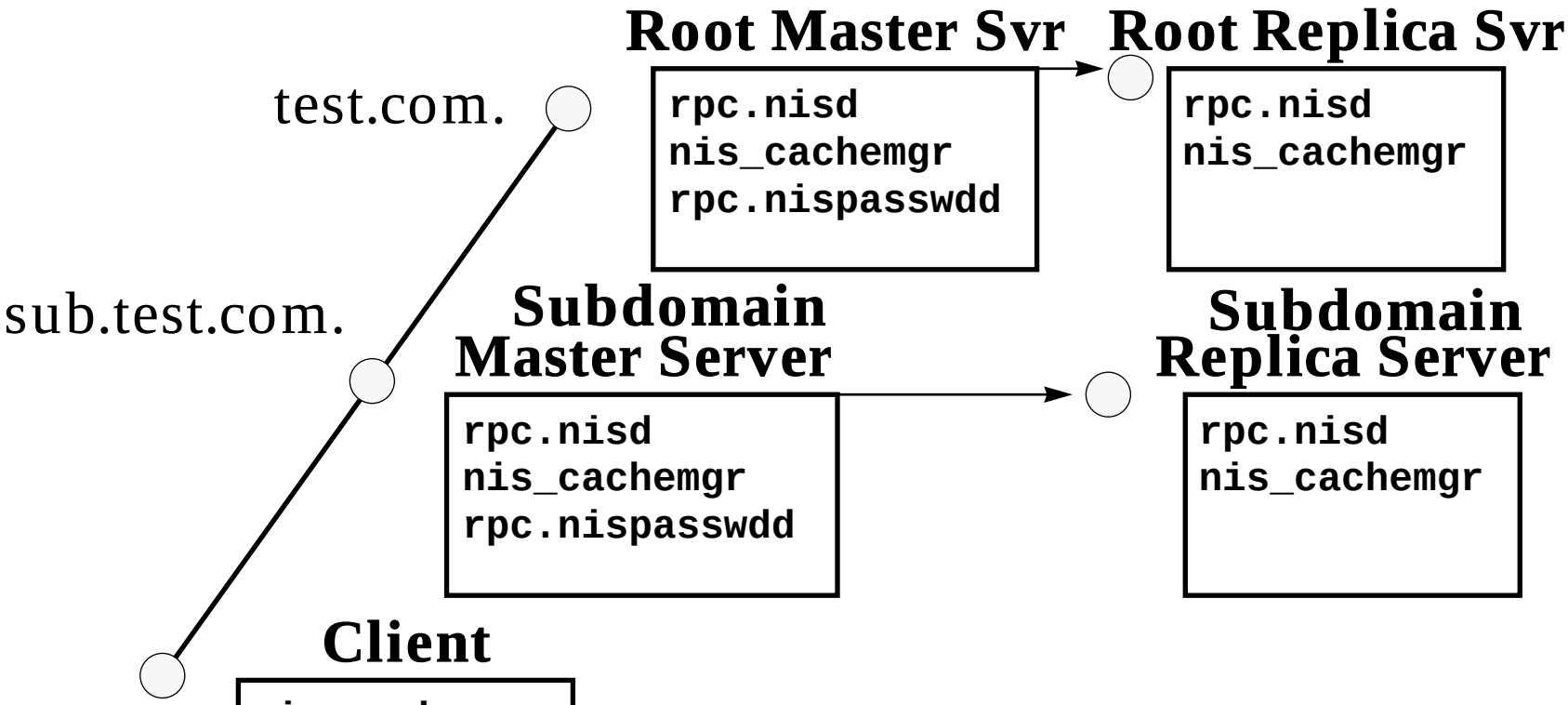

- **`org_dir`**

  **Contains standard (i.e. /etc) tables (see above)**

- **`groups_dir`**

  **Contains all groups; must at least have an administration group**

# Typical NIS+ Processes




- NIS+ daemon, `rpc.nisd`, runs on all servers
- cache manager, `nis_cachemgr`, runs on all clients
- NIS+ password daemon runs on all master servers

# *Frequent NIS+ Client Commands*

| | |
|---|---|
| `niscat(1)` | **Display NIS+ tables** |
| `nismatch(1)` | **Search NIS+ table: simple text match** |
| `nisgrep(1)` | **Search NIS+ table: pattern match** |
| `nisls(1)` | **List NIS+ directory contents** |
| `nisdefaults(1)` | **Display NIS+ Default Values** |

# *Frequent NIS+ Administration Cmds*

| | |
|---|---|
| `nisserver(1m)` | **Setup NIS+ servers** |
| `nisclient(1m)` | **Setup NIS+ clients** |
| `nispopulate(1m)` | **Create/update NIS+ database** |
| `nisping(1m)` | **Ping replicas or update database** |
| `nistbladm(1m)` | **Administer NIS+ tables** |
| `nisstat(1m)` | **Display NIS+ server statistics** |

# *NIS+ API*

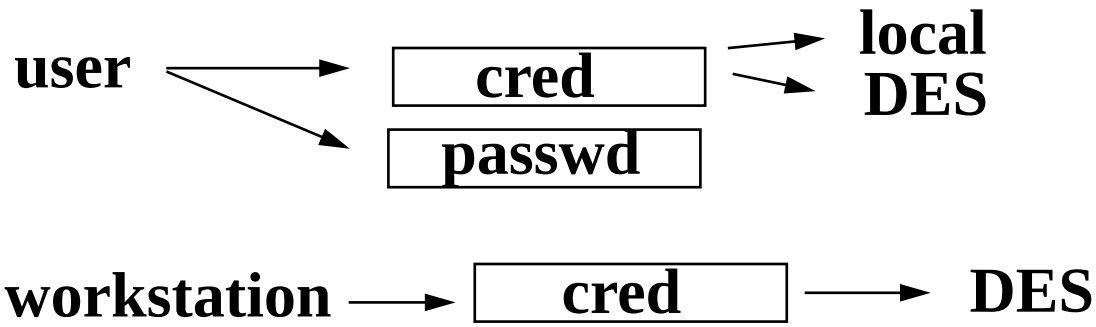
| | |
|---|---|
| `nis_objects(3n)` | **Object/structure formats** |
| `nis_tables(3n)` | **Table functions** |
| `nis_names(3n)` | **Namespace functions** |
| `nis_subr(3n)` | **Miscellaneous subroutines** |
| `nis_server(3n)` | **Server-related functions** |
| `nis_groups(3n)` | **Group manipulation functions** |

# *NIS+ Principals and Credentials*

- **A NIS+ *Principal* is any name service client**
- **There are two types of principals:**
  - **User: *john@doe* Principal name= "*john.test.com.*"**
  - **Workstation: *root@doe* Principal Name= "*doe.test.com.*"**
- **Every principal must have *Credentials***
  - **Main components to authentication**
  - **NIS+ uses Secure-RPC**
  - **Stored in `cred.org_dir` table**
- **There are two types of credentials:**
  - **local standard AUTH_SYS RPC credentials (user principals only)**
  - **DES AUTH_DES Secure-RPC credentials (both principal types)**

# NIS+ Principals and Credentials (cont.)

**Principals and their Credentials:**




- **A *client request* is first authenticated**
  - **Must prove that you are the principal you claim to be**
  - **Uses Secure-RPC credentials with Diffie-Hellman encryption**
- **A *client request* is then checked for authorization**
  - **Client has already been shown to be authenticated**
  - **Must prove now that client has permission to perform operation**

# *NIS+ Authorization Model*

- **Four classes of principals:**

  ***Owner*** **of a Object,** ***Group*** **or set of specified principals,** ***World*** **or set of authenticated principals and** ***Nobody*****: or all clients (includes World)**

- **Four access rights:**

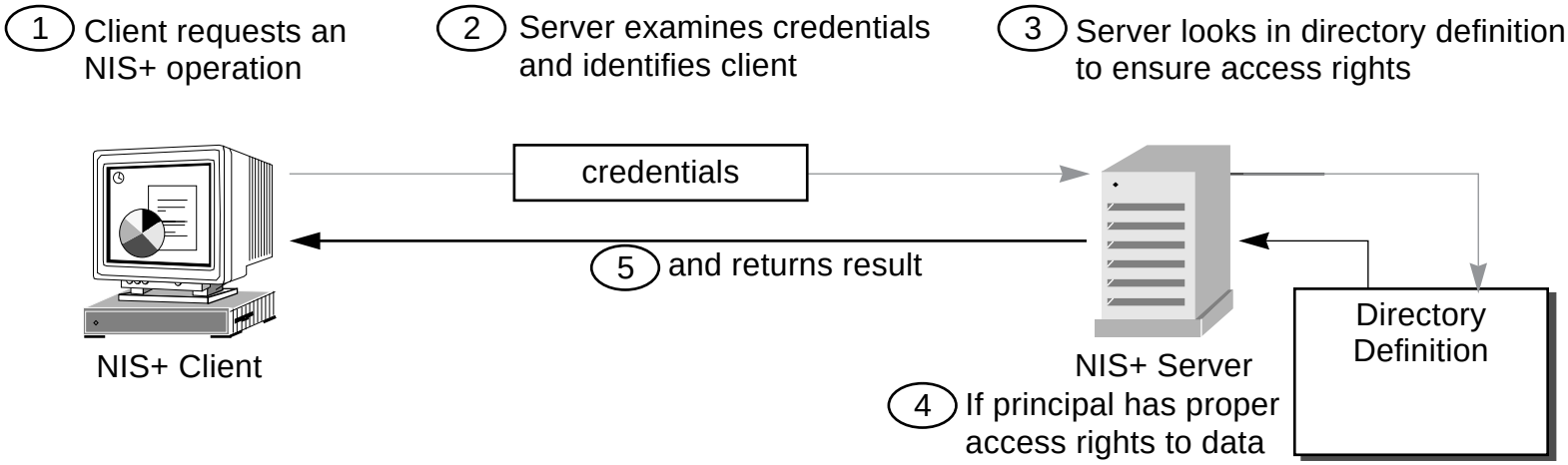
  ***Read*** **contents of objects,** ***Modify*****/change attribute,** ***Create*****, and** ***Destroy***

  **Access rights can be specific on per-directory, -table, -entry, -column**

**Table**

| Row 0 | Entry Data | Column 1 | Column 2 | ••• | Column N |
|---|---|---|---|---|---|
| Row 1 | Entry | | | | |
| ⋮ | | | | | |
| Row M | | | | | |

# Authentication and Authorization




- **Client identity verification to prevent unauthorized access**
  - **Secure administration from across the network**
- **Access controls to protect administrative information**
  - **Flexible setting of administrative policies**

# *NIS+ Files*

- **Transaction log**
  - `/var/nis/trans.log`
- **Cold start file**
  - `/var/nis/NIS_COLD_START`
  - Contains parent directory object
- **Directory object cache file**
  - `/var/nis/NIS_SHARED_DIRCACHE`
  - Caches most recently accessed directories
  - Used by nis_cachemgr(1m)
- **NIS+ database files**
  - `/var/nis/data` directory

# *Using the NIS+ Setup Scripts*

- **Setting up NIS+ servers:**

| | |
|---|---|
| `# nisserver -r` | **setup NIS+ root master server** |
| `# nisserver -M` | **setup NIS+ subdomain master server** |
| `# nisserver -r` | **setup NIS+ replica server** |

- **Setting up NIS+ database:**

| | |
|---|---|
| `# nispopulate -Y` | **populate NIS+ tables from NIS/YP maps** |
| `# nispopulate -F` | **populate NIS+ tables from /etc/files** |

- **Setting up NIS+ clients:**

| | |
|---|---|
| `# nisclient -i` | **initialize NIS+ client machine** |
| `# nisclient -u` | **initialize NIS+ client users** |
| `# nisclient -c` | **create NIS+ principal credentials** |

# *The Future*

# *Transitioning from NIS to NIS+*

- **YP-compatibility mode**
  - **NIS+ servers can serve YP client requests**
- **NSkit 1.2**
  - **YP server available for Solaris 2**
- **NIS+ setup scripts**
  - **Can create NIS+ database from YP map set**

# *What's new in Solaris 2.5*

- **NIS+ Password Update Daemon, `rpc.nispasswdd(1m)`**
  - **Handles password updates without credentials**
  - **Enforces and manages functional password aging**
- **NIS+ portable dictionary and database**
  - **No more `/var/nis/`*`hostname`* and `/var/nis/`*`hostname`*`.log`**
  - **Now `/var/nis/data` and /`var/nis/trans.log`**
- **Name Service Cache Daemon, `nscd(1m)`**
  - **Caches most common name service requests, i.e. hosts, passwd, groups**
- **X/Open Federated Naming (XFN)**
  - **Solaris implementation of XFN spec: Federated Naming Service (FNS)**
  - **Provides policies and name composition for multiple name services**
  - **Creates application coherence with single API for all name service access**