3 November 1995

Accessing the bindery files directly

Alastair Grant, Cambridge University

1. Introduction

This document describes a command for accessing the NetWare 3.x bindery files directly, bypassing the NetWare network API calls.

It can be used for fast bindery access, bulk user management, bypassing security restrictions, investigating problems etc.

It is quite possible to destroy the bindery completely, or to reveal information which could be used by hackers to obtain passwords. Users are assumed to have a basic grasp of good procedures for security and backup.

2. Command syntax

The basic format of the command is

`bindery` *[options] bindery-spec action action* `...`

2.1 Specifying a bindery

A bindery specification takes the form

path`/.`*extension*

E.g. `SYS:SYSTEM/.SYS`. The path defaults to the current directory. The extension defaults to `.OLD`.

Alternatively an ‘active’ bindery can be specified:

`SERVER` *server*

The bindery will be closed if necessary.

2.2 Actions on the bindery

`INFO`	print info about the bindery
`SCHEMA`	checks the bindery against the schema in `BINDERY.SCH`
`DUMP` *obj*	dump all information for the specified object(s)
`OBJ`	list all object records
`PROP`	list all property records
`VAL`	list all value records
`VALDATA`	list all value records, with data
`EXPORT`	export the bindery to a text file; see below
`IMPORT`	import the bindery from a text file
`ETC`	export user password information, suitable for input to the password-cracking program described below

The following actions apply only if a bindery has been specified by the `SERVER` parameter:

`CLOSE`	close the bindery, i.e. make it available for direct access; users attempting to access the bindery via NetWare API calls will receive an error
`OPEN`	open the bindery, which causes the server to reload it and may take some time for large binderies
`COPY` *directory*	copy the bindery files into a directory elsewhere

3. Export/import

The bindery can be exported to and imported from a text file. This can be used for various purposes:

- problem diagnosis and repair
- creation of large binderies given a set of user information
- compaction of binderies
- merging binderies or moving users between binderies while preserving their passwords

To see the format of the export file, try exporting a small bindery.

4. Password cracking

Passwords are not stored in clear in the bindery. What is stored is a 16-byte value computed via a one-way function from the user's object id and the password. Given the object id and password it is possible to generate a candidate password which can be compared against that in the bindery.

The `ETC` option of the **bindery** command produces a file containing the required information, in a format superficially similar to `/etc/passwd` on Unix:

userid:*pw-hash*:*object-id*:*pw-len*:*name*::

e.g.

```
ttidy:32d8998e098a05830f809b809ea02137:D0000001:8:Terry Tidy
```

This can then be input into bindery cracking programs. Separating the functions in this way allows various forms of parallelism:

- the password file can be split into smaller chunks
- the same password file can be worked on by several cracking programs each with different dictionaries or algorithms
- cracking programs can be run on faster machines

A cracking program **bincrack** is provided which takes such a file as input. It has command syntax:

```
bincrack [/verify] [/numsub] pw-file dict-file
```

`/verify` lists the passwords that are being tried. `/numsub` tries substituting numbers for letters, e.g. "`1D10T`". This takes a lot longer as all possible combinations are tried. *pw-file* is an exported bindery password file. *dict-file* is a simple word list.

Versions are available for MS-DOS and for Solaris 1 and Solaris 2 SPARC systems.

Suitable wordlists can be found at

```
ftp://ftp.ox.ac.uk/pub/wordlists/
```