dumpntds

Background

The normal workflow for dumping passwords from **ntds.dit** files is to use the **esedbexport** application from the **libesedb** project. The files generated by esedbextract are then fed into the ntdsxtract project. ntdsxtract uses the files to parse out various different items of information, in this case we would want password hashes that could be fed into john the ripper.

On large domains, the ntds.dit file can be extremely large (10 GB+), from which extracting all of the columns to a CSV file can take a long time, considering the **datatable** table contains over 1000 columns.

The aim of dumpntds is to extract the minimal amount of data required (45 columns) to perform the task in hand, thus speeding up the process.

dumpntds uses the ManagedEsent library to access the data stored in the ntds.dit file. The ManagedEsent library wraps the underlying Windows API calls and therefore needs to be run using .Net, rather than Mono.

Usage

Extract the ntds.dit file from the host and run using the following:

```
dumpntds -n path\to\ntds.dit\file
```

Once the process has been completed it will have generated two output files in the application directory:

- datatable.csv
- linktable.csv

dsusers

The extracted files can then be used with the **dsusers.py** script from the ntdsxtrct project:

```
python ./dsusers.py datatable.csv linktable.csv . --passwordhashes --syshive SYSTEM --pwdformat john --lmoutfile lm.txt --ntoutfil
```

dshashes.py

I have also included an updated version of the dshashes python script, which was broken due to changes in the underlying ntds library. The dshashes script can be used as follows:

```
python ./dshashes.py datatable.csv linktable.csv . --passwordhashes SYSTEM
```