

Objective--Recover Alabaster's Password (Part 5)

Decrypting the key

The steps shown here appear simple, but they are the result of hours of errors and searching. The original function that used public key encryption to encrypt \$Byte_key is here.

```
1 $cert = New-Object -TypeName System.Security.Cryptography.X509Certificates.X509Certificate2
2 $cert.Import($pub_bytes)
3 $encKey = $cert.PublicKey.Key.Encrypt($key_bytes, $true)
4 |
```

As the assignment from the last lesson states, the import function will be used differently because the new PFX file we made requires a password (my password was "password".) We will need to change line 3 to use the Private Key instead of Public Key, and the syntax is slightly different. Lastly, we will need to copy any conversion functions we need from the malware.

This is the code we will use to decrypt the key.

```
1 function H2B {
2     param($HX)
3     $HX = $HX -split '(.)' | ? { $_ }
4     ForEach ($value in $HX){
5         [Convert]::ToInt32($value,16)
6     }
7 }
8 function B2H {
9     param($DEC)
10    $tmp = ''
11    ForEach ($value in $DEC){
12        $a = "{0:x}" -f [Int]$value
13        if ($a.length -eq 1){
14            $tmp += '0' + $a
15        } else {
16            $tmp += $a
17        }
18    }
19    return $tmp
20 }
21
22 $cert = New-Object -TypeName System.Security.Cryptography.X509Certificates.X509Certificate2
23 $path_to_key = Get-ChildItem C:\Users\John\malware\server.pfx
24 $cert.Import($path_to_key, "password", $true)
25 $enc_key_hex = Get-Content C:\Users\John\malware\512byte_values.txt
26 $enc_key_bytes = H2B $enc_key_hex
27 $Key_Bytes = $cert.PrivateKey.Decrypt($enc_key_bytes, $true)
28 $Key_Hex = B2H $Key_Bytes
29
```

If we print the key after the decryption script runs, we have something that looks reasonable.

```
PS C:\Users\John\malware> $key_hex
fbcf121915d99cc20a3d3d5d84f8308
```

It would be good to save the values to files.

```
\malware> $key_hex | Out-File key_hex.txt
\malware> $Key_Bytes | Out-File key_bytes.bin
```

Decrypting Alabaster's Password Database

The malware uses function Enc_Dec-File to encrypt and decrypt files using AES encryption. The other function, enc_dec, just keeps 12 jobs running at a time, and each of those jobs are just calls to Enc_Dec-File. We only have one file to decrypt, so we can skip enc_dec. Note: remember that Enc_Dec-File wants the binary version of the key.

You should be able to use the malware function to decrypt Alabaster's file. The easiest way is to paste the code of the function into a new tab (remove the function line and the closing brace.) Then write lines above the ex-function to give it the values it needs for \$key (binary version of the key), \$file (path to Alabaster's wannacookie file), and false (you do want to decrypt, I assume.)

Once you decrypted Alabaster's file, you will find it is a sqlite3 database. You can learn to read the database using information [here](#). Installation shouldn't be necessary if you use sqlite3 in a Linux VM. Find the name of the database, then the name of the table. Once you know that you can use a SELECT statement to dump the table. Or, you can just see if the file contains any text...

Hand in

- 1) What is Alabaster's password for the vault?