```
System  . . . . . . : DEFCON23
Subsystem . . . . . : QTRACK4
Display . . . . . . : QPADEV0001
```

Hack the legacy!

IBM i (aka AS/400) revealed.

Bart Kulach

```
MA   a             MV                                              08/007
```

Agenda

- Let’s get introduced
- Why should we care about legacy?
- Evil Java?
- Privilege escalation – let’s jump!
- Password security and hash grabbing
- Summary + Q&A

Let’s get introduced

- I’m googleable.

https://nl.linkedin.com/in/bartkulach

Disclaimer:

Any views or opinions presented here are solely those of the author and do not necessarily represent those of his employer(s).

KEEP
CALM
and
PWRDWNSYS
*IMMED

Why should we care about legacy?

Why should we care about legacy?

- It’s legacy… but hard to get rid of.
- It’s processing most interesting data.
- It’s usually less secure than front-ends.
- It’s often more vulnerable than you think.
- It’s still quite accessible to potential intruders.
- It’s existing everywhere - in all economic sectors.
- It’s already been exploited (“Hacking iSeries” by S.Carmel)!

Evil Java?

Evil Java?

- IBM Toolbox for Java/JTOpen
- Allows for remote system API calls and usage of built-in system commands (“Limited capability” not effective here)
- Gives the flexibility of coding “outside” the AS/400 box (no need for extra authorities on the system)
- Is generally poorly written (decompile and check yourself!)
- Handling of authorisations by Java VM on server side is inconsistent (object authority vs. data authority), allowing for greater visibility

Demo time:
Evil Java – visibility example

Privilege escalation – let’s jump!

Privilege escalation – let's jump Part 1 – remote profile switching

- Do you use group profiles? Like one common group profile?
- Are your admins also members of the group?
- Are your object **<u>and</u>** data authorities hardened?
- Do you monitor profile handle swapping?
- Let's jump remotely:
  - check the list of profiles you have access to
  - grab a profile handle
  - switch to the profile
  - repeat until you're happy with your access level ☺

Demo time:
Remote profile switching

Privilege escalation – let's jump Part 2 – nested command use

- Exit points/programs generally allow to protect the system quite easily from usage of specific SQL queries or system commands
- Most commercial protection software that use exit programs have their weaknesses/vulnerabilities.
- They can be however often be circumvented by using nested commands (commands running commands)
- Especially if you cross the environments (CL–PASE–DB2)…
- And if we add JDBC to that… like

  ```
  CALL QSYS.QCMDEXC('QSH CMD(''DB2 "select * from
  library.file" | Rfile -w /QSYS.LIB/QSYSPRT.FILE'')',
  0000000077.00000 ☺
  ```

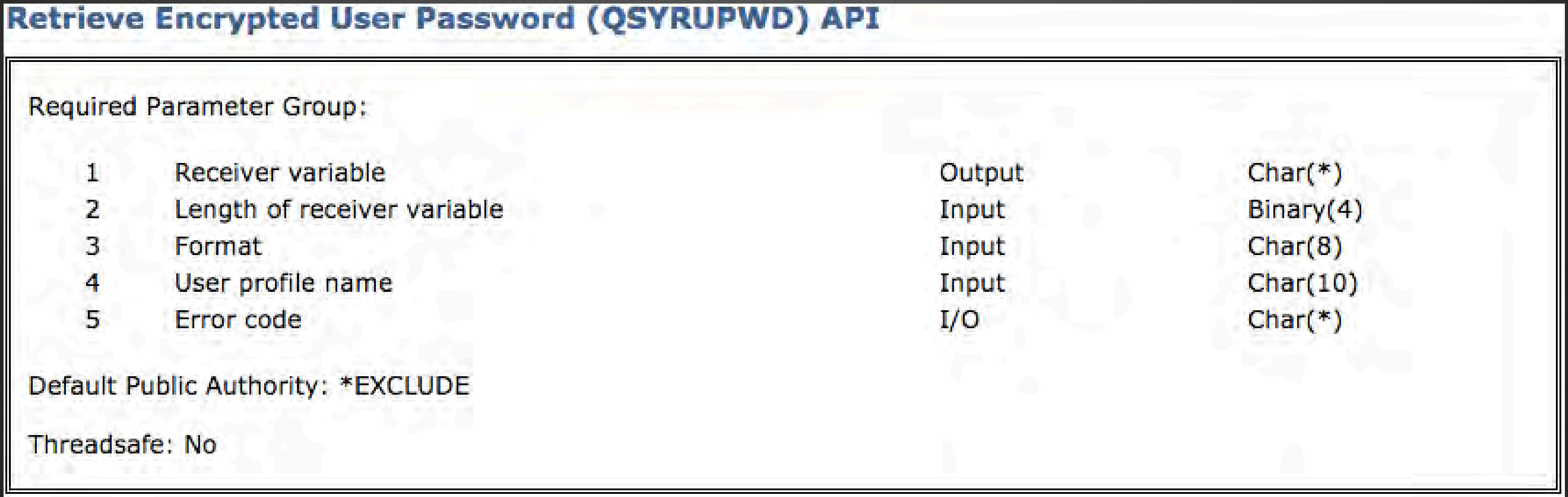
Demo time:
Nested command use

Password security and hash grabbing

Password security and hash grabbing

- IBM offers you a nice API (QSYRUPWD) to grab the hashes.
- QSYRUPWD allows for getting an extract of all hashes for a particular user.
- The output format is proprietary and was never published until today ☺
- Is your QPWDLVL system value 0, 1 or 2*? If so, you can enjoy the LM hashes ☺
  **for QPWDLVL=2, QPWDMAXLEN must be <=14*
- You have to be *SECADM (and ideally *ALLOBJ) though (so go back and escalate your privileges first).

Password security and hash grabbing – cont'd.



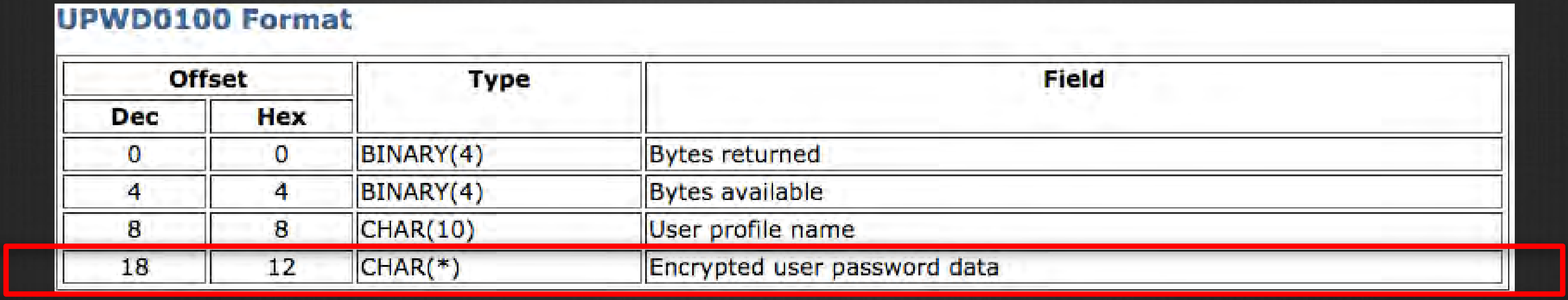
Retrieve Encrypted User Password (QSYRUPWD) API

Required Parameter Group:

1	Receiver variable	Output	Char(*)
2	Length of receiver variable	Input	Binary(4)
3	Format	Input	Char(8)
4	User profile name	Input	Char(10)
5	Error code	I/O	Char(*)

Default Public Authority: *EXCLUDE

Threadsafe: No



UPWD0100 Format

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Bytes returned
4	4	BINARY(4)	Bytes available
8	8	CHAR(10)	User profile name
18	12	CHAR(*)	Encrypted user password data

Password security and hash grabbing – cont’d.

- QSYRUPWD Encrypted password data hex string

Offset (Dec)	Length (Chars)	Field	QPWDLVL
0	16	DES 56-bit encrypted password substitute (RFC2877)	0, 1, 2*
16	16	DES 56-bit encrypted password substitute (RFC2877)	0, 1, 2*
32	32	LM hash	0, 1, 2*
64	*4*	*No data*	-
68	40	HMAC-SHA1 encrypted password token (RFC4777)?	0**, 1**, 2, 3
108	40	HMAC-SHA1 encrypted password token (RFC4777)?	0**, 1**, 2, 3
148	*6*	*No data*	-
154	384	Unknown (hash?) data	0, 1, 2, 3

**depending on password rules; **from V5R1 onwards*

Demo time:
Password grabbing

Summary + Q&A

- Java is the evil for AS/400.
- Be sceptic about IBM Security books.
- Visit www.hackthelegacy.org

@bartholozz

www.hackthelegacy.org