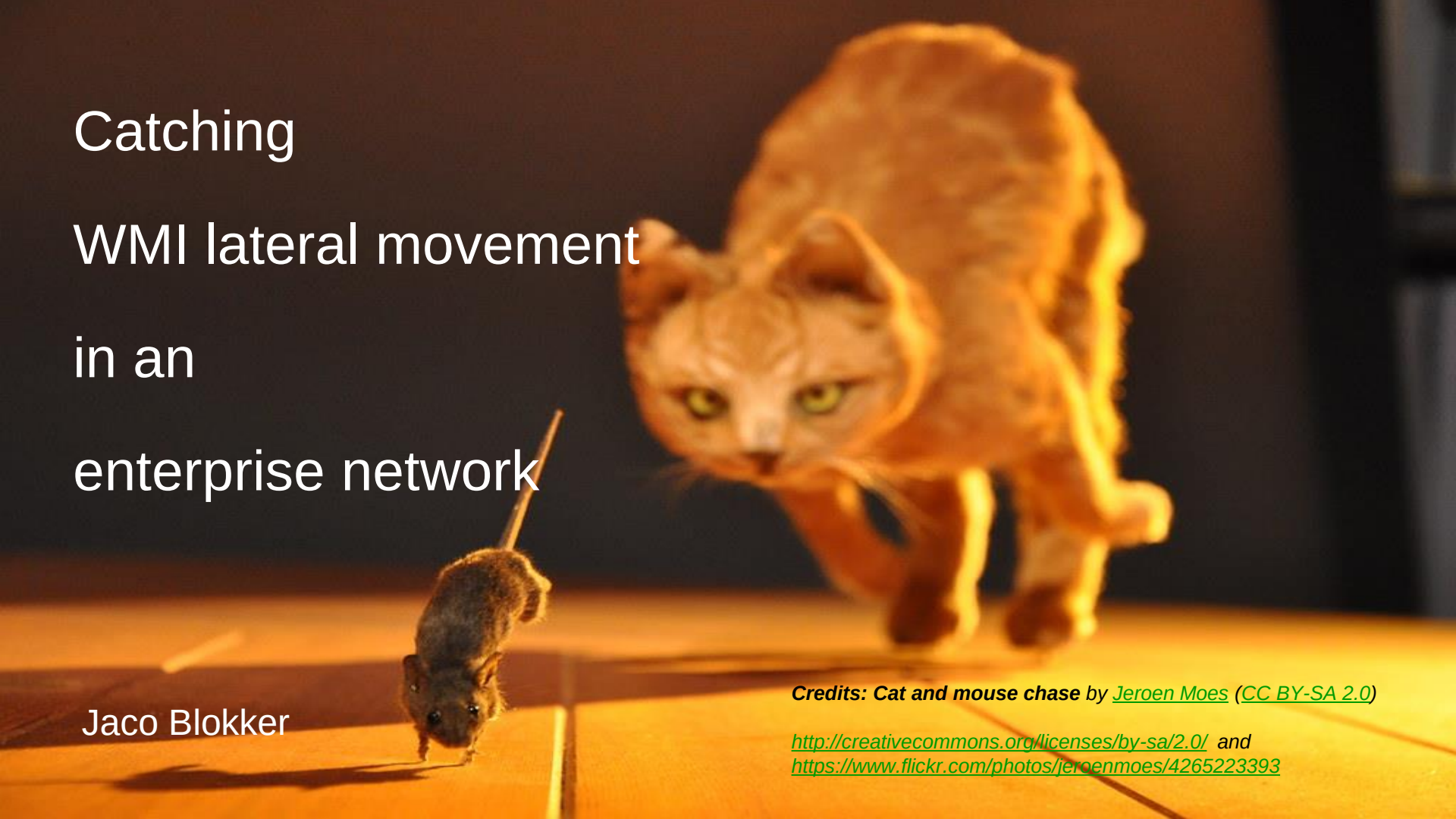


Catching WMI lateral movement in an enterprise network

Jaco Blokker

About Jaco Blokker

15y infosec, 6y Senior member Blue team KPN

Event analysis

Develop, tune (network) detection, Snort

Bottom-line-up-front

Network detection IDS ruleset:

github.com/KPN-CISO/Network-Detection

Why sharing it ?

Agenda

- Method, approach
- Findings
- What worked and what not

…Starting point for future enhancements

About WMI

Dmtf standards

- WBEM Web-Based Enterprise Management
- CIM Common Information Model

Microsoft’s implementation:
Windows Management Instrumentation

WMI characteristics

Core OS component

- Read, manipulate, execute

Access defaults

- local administrator (allowed)
- hostbased firewall (blocked)

Transports

- WS-man (Winrm)
- RPC/DCOM [*this research*]

Different perspectives

System administrator
Attacker
Defender

How to detect + distinguish legit / non-legit?

Just theory?

MITRE | ATT&CK™

APT32	APT32 used WMI to deploy their tools on remote machines and to gather information about the Outlook process. [61]
Astaroth	Astaroth uses WMIC to execute payloads. [35]
BlackEnergy	A BlackEnergy 2 plug-in uses WMI to gather victim host details. [15]
Cobalt Strike	Cobalt Strike can use WMI to deliver a payload to a remote host. [5]
Deep Panda	The Deep Panda group is known to utilize WMI for lateral movement. [49]
DustySky	The DustySky dropper uses Windows Management Instrumentation to extract information about the operating system and whether an anti-virus is active. [25]
Emotet	Emotet has used WMI to execute powershell.exe. [45]
Empire	Empire can use WMI to deliver a payload to a remote host. [10]
EvilBunny	EvilBunny has used WMI to gather information about the system. [46]
FELIXROOT	FELIXROOT uses WMI to query the Windows Registry. [14]
FIN8	FIN8's malicious spearphishing payloads use WMI to launch malware and spawn cmd.exe execution. FIN8 has also used WMIC during and post compromise cleanup activities. [56] [57]
FlawedAmmyy	FlawedAmmyy leverages WMI to enumerate anti-virus on the victim. [43]
GravityRAT	GravityRAT collects various information via WMI requests, including CPU information in the Win32_Processor entry (Processor ID, Name, Manufacturer and the clock speed). [20]
HALFBAKED	HALFBAKED can use WMI queries to gather system information. [28]

kpn

Define monitoring objectives

- Is it doable?
- Non-legit usage vs vulnerability detection
- Detect anomalies on network level

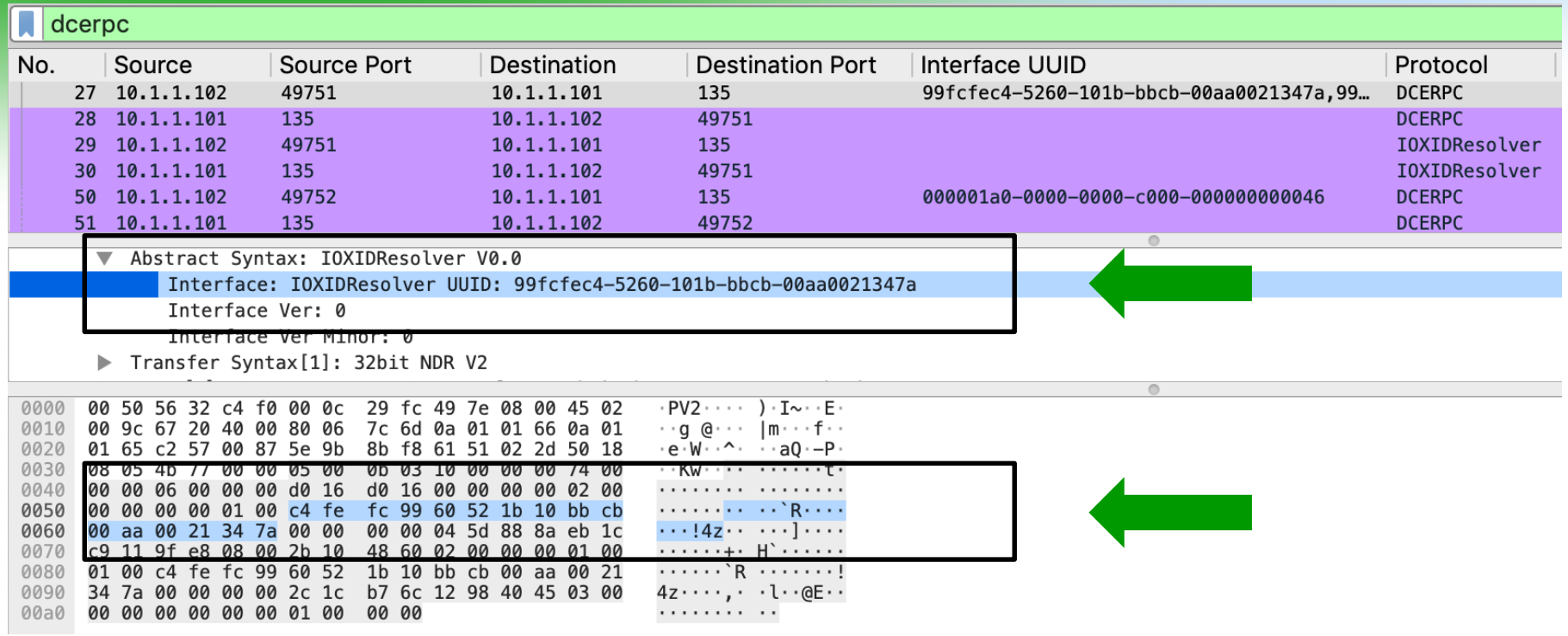
Support defender with:

- Evidence, context (who/what), attempts (success/failure)

Our very first attempt

- What does WMI look like from network perspective?
- PS> Get-Wmiobject Win32_computersystem –Computername WIN-J0GNFCAISH2.testing.local –Credential <lookwhoistalking> Ipconfig.exe
- [Not authorized]. We knew.

On the wire


dcerpc
No. Source Source Port Destination Destination Port Interface UUID Protocol
27 10.1.1.102 49751 10.1.1.101 135 99fcfec4-5260-101b-bbcb-00aa0021347a,99… DCERPC
28 10.1.1.101 135 10.1.1.102 49751 DCERPC
29 10.1.1.102 49751 10.1.1.101 135 IOXIDResolver
30 10.1.1.101 135 10.1.1.102 49751 IOXIDResolver
50 10.1.1.102 49752 10.1.1.101 135 000001a0-0000-0000-c000-000000000046 DCERPC
51 10.1.1.101 135 10.1.1.102 49752 DCERPC
Abstract Syntax: IOXIDResolver V0.0
Interface: IOXIDResolver UUID: 99fcfec4-5260-101b-bbcb-00aa0021347a
Interface Ver: 0
Interface Ver Minor: 0
Transfer Syntax[1]: 32bit NDR V2
0000 00 50 56 32 c4 f0 00 0c 29 fc 49 7e 08 00 45 02 ·PV2····)·I~··E·
0010 00 9c 67 20 40 00 80 06 7c 6d 0a 01 01 66 0a 01 ··g @··· |m···f··
0020 01 65 c2 57 00 87 5e 9b 8b f8 61 51 02 2d 50 18 ·e·W··^· ··aQ·-P·
0030 08 05 4b 77 00 00 05 00 0b 03 10 00 00 00 74 00 ··Kw···· ·····t·
0040 00 00 06 00 00 00 d0 16 d0 16 00 00 00 00 02 00 ···· ···· ····
0050 00 00 00 00 01 00 c4 fe fc 99 60 52 1b 10 bb cb ····· ·· ··`R····
0060 00 aa 00 21 34 7a 00 00 00 00 04 5d 88 8a eb 1c ···!4z·· ···]····
0070 c9 11 9f e8 08 00 2b 10 48 60 02 00 00 00 01 00 ····· +· H`····
0080 01 00 c4 fe fc 99 60 52 1b 10 bb cb 00 aa 00 21 ····· `R ····· !
0090 34 7a 00 00 00 00 2c 1c b7 6c 12 98 40 45 03 00 4z···· ,· ·l··@E··
00a0 00 00 00 00 00 00 01 00 00 00 ···· ···· ··

Initial payload filter

- c4 fe fc 99 60 52 1b 10 bb cb 00 aa 00 21 34 7a
- Turned it into a rule like:
  (“guess this is WMI !”;
  content:”|c4 fe fc 99 60 52 1b 10 bb cb 00 aa 00 21 34 7a|”)
- Risk of false positive?

RPC call

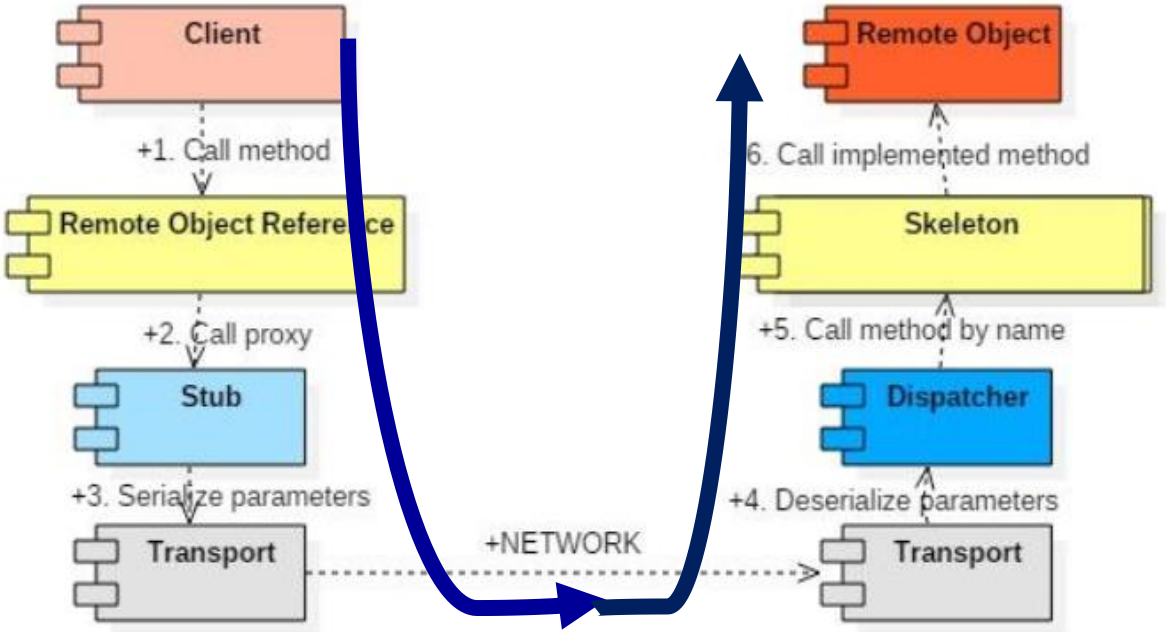


Figure 2. Simple RPC call

From: coned.utclu.ro/~salomie

Map onto RPC preprocessor

dce_iface

dce_opnum

dce_stub_data

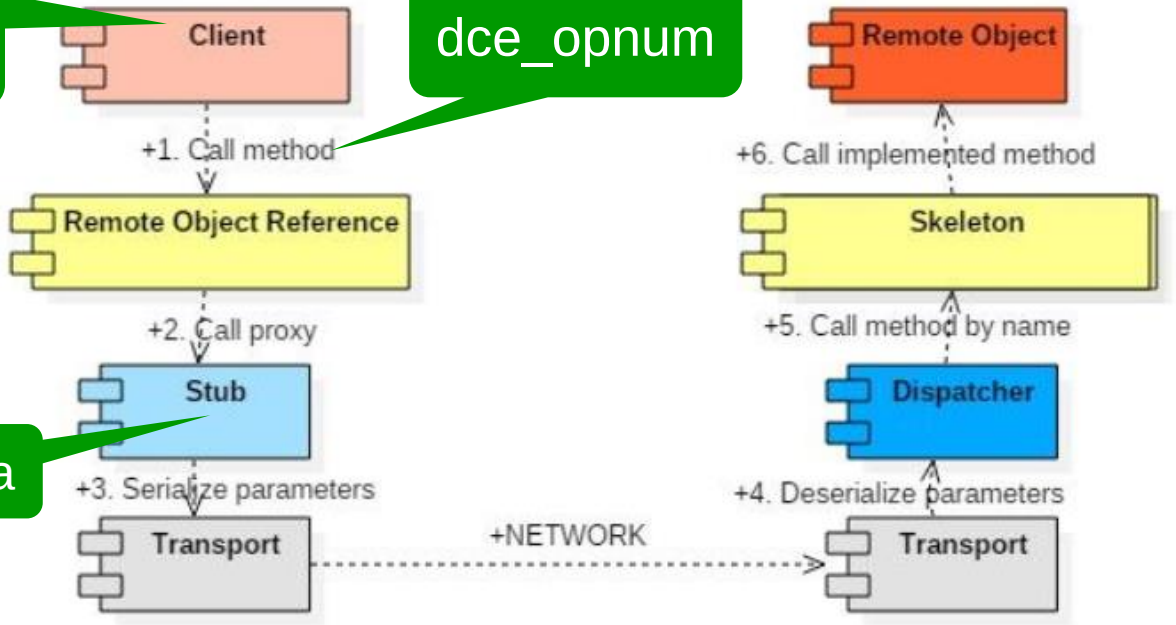


Figure 2. Simple RPC call

Detection pattern changed into

(msg:“guess this is WMI !”;
dce_iface:99fcfec4-5260-101b-bbcb-00aa0021347a; …)

> Did a re-test: triggers again!

Preprocessor abstracts away: used endianess type

Differentiate legit - non-legit

- Up to here: 1st detection pattern defined

- Next, few suggestions to distinguish:
- Based on time? Used credentials ? traffic path?

- What is expected to be legitimate traffic in the enterprise?

Differentiate legit - non-legit

- Engage with system administrators
- Establish a policy if not already there:
  “We shall administer <this and that> using WMI only from < *endpoints > “

  *) Typically steppingstone-like

Pitfall: change management

kpn

Payload pattern and policy combined

Whitelist approach:

alert tcp !$legitimate_sources any -> $protected_targets 135
…
(msg:"guess this is WMI !";
dce_iface:99fcfec4-5260-101b-bbcb-00aa0021347a; ..)

Next: offer our rule for re-test

Revisited by redteam

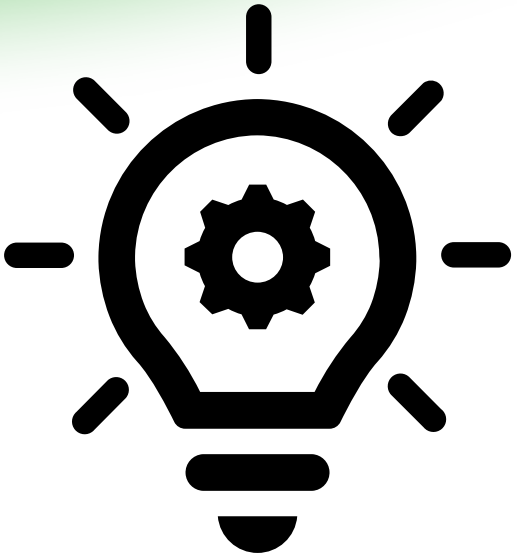
Various clients

- Windows powershell
- Linux Impacket (low-level network protocol library)

- Using unauthorized account

Results

- Windows client triggered the rule as expected
- However: “Linux” based client did not ;-(
- What next ?



[MS-WMI] Protocol Initialization

- “The client MUST call the IWbemLevel1Login::NTLMLogin method.
- The IWbemLevel1Login interface allows a user to connect to the management services interface in a particular namespace.
- The interface MUST be uniquely identified by the UUID {F309AD18-D86A-11d0-A075-00C04FB68820} “

From: https://msdn.microsoft.com/en-us/library/dd208060.aspx

kpn

3.1 Server Details WMI

CLSID_WbemLevel1Login ({8BC3F05E-D86B-11D0-A0

CLSID_WbemBackupRestore ({C49E32C6-BC8B-11D2-

..

The following GUIDs are used for the interfaces:

IID_IWbemLevel1Login ({F309AD18-D86A-11d0-A075

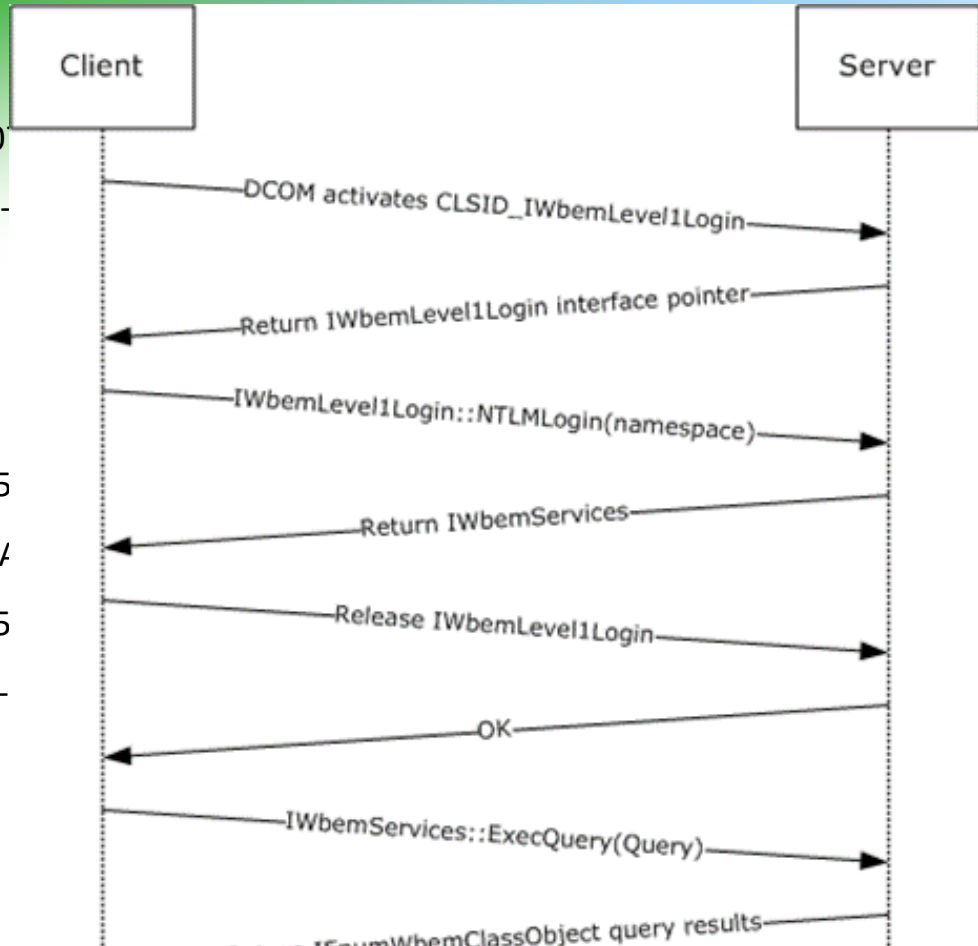
IID_IWbemServices ({9556DC99-828C-11CF-A37E-00A

IID_IWbemBackupRestore ({C49E32C7-BC8B-11d2-85

IID_IWbemBackupRestoreEx ({A359DEC5-E813-4834-

…

From: https://msdn.microsoft.com/en-us/library/dd208060.aspx



Do cross-check

examples/wmiquery.py

Showing the top match Last indexed on Aug 15, 2017

```
185                                    doKerberos=options.k, kdcHost=options.dc_ip)
186
187             iInterface =
        dcom.CoCreateInstanceEx(wmi.CLSID_WbemLevel1Login,wmi.IID_IWbemLevel1Login)
```

https://github.com/CoreSecurity/impacket/

kpn

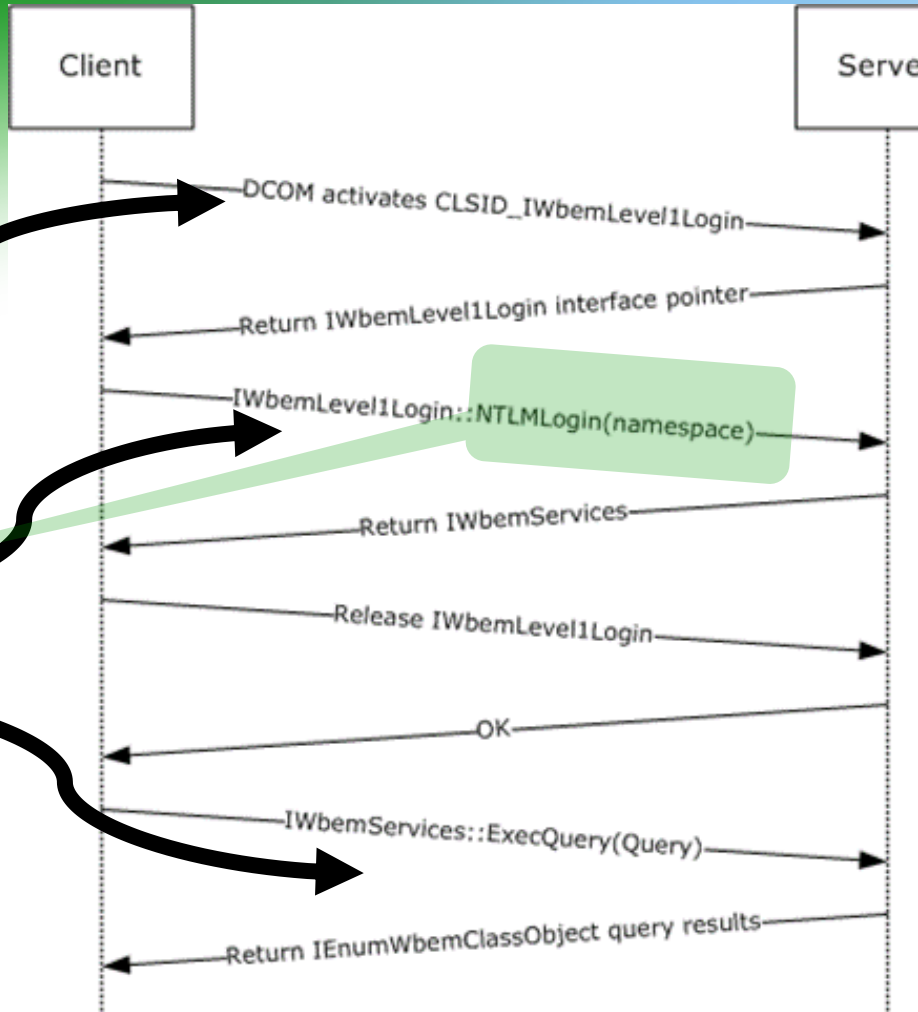
Detection pattern category

Rpc stage:

- Rule – AppID for service winmgmt

Indicator WMI, presumed success

- IID Iwbemlevel../opnum 6
- IID Iwbemservices/.. (Bonus)



Ruleset becomes (condensed)

'RPC' stage

- #100: dce_iface:000001a0-0000-0000-c000-000000000046; dce_opnum: 4; dce_stub_data; content: "|5e f0 c3 8b 6b d8 d0 11 a0 75 00 c0 4f b6 88 20|

'WMI' stage; Golden Rule:

- #110: dce_iface:F309AD18-D86A-11d0-A075-00C04FB68820; dce_opnum:6

- #114: dce_iface:9556dc99-828c-11cf-a37e-00aa003240c7
- #116, #120: likewise for IID's IEnumwbemobject/IWbemclassobject

kpn

Detect bruteforce attempts

- Indicator WMI call (rule #110) and subsequent calls may not happen
- Force multiple failed attempts:
- PS C:>\wmic /node: “10.1.1.101" process call create "cmd.exe /c ipconfig.exe"

Server replies with ‘fault PDU’

```
▷ Transmission Control Protocol, Src Port: 135, Dst Port: 60169, Seq: 269, Ack: 989, Len: 32
◢ Distributed Computing Environment / Remote Procedure Call (DCE/RPC) Fault, Fragment: Single, FragLen: 32, Call: 2, [Req: #8]
    Version: 5
    Version (minor): 0
    Packet type: Fault (3)
  ▷ Packet Flags: 0x03
  ▷ Data Representation: 10000000 (Order: Little-endian, Char: ASCII, Float: IEEE)
    Frag Length: 32
    Auth Length: 0
    Call ID: 2
    Alloc hint: 32
    Context ID: 0
    Cancel count: 0
    Status: nca_s_fault_access_denied (0x00000005)
  ◢ [Expert Info (Note/Response): Fault: nca_s_fault_access_denied]
        [Fault: nca_s_fault_access_denied]
        [Severity level: Note]
        [Group: Response]
    Opnum: 4
    [Request in frame: 8]
    [Time from request: 0.873886000 seconds]
```

```
0000  d4 81 d7 b9 1b 81 00 08  e3 ff fd 18 08 00 45 00   ........ ......E.
0010  00 48 20 6f 40 00 74 06  a0 66 90 2c 65 fb ac 1f   .H o@.t. .f.,e...
0020  a3 93 00 87 eb 09 fd a1  00 8c 92 c8 87 fe 50 18   ........ ......P.
0030  01 ff 04 4a 00 00 05 00  03 03 10 00 00 00 20 00   ...J.... ...... .
0040  00 00 02 00 00 00 20 00  00 00 00 00 00 00 05 00   ...... . ........
0050  00 00 00 00 00 00                                 ......
```

kpn

Fault PDU structure

- pubs.opengroup.org/
- onlinepubs/9629399/chap12.htm

The fault PDU

The IDL declaration of the **fault** PDU is as follows:

```
typedef struct {

      /* start 8-octet aligned */

      /* common fields */
        u_int8  rpc_vers = 5;        /* 00:01 RPC version */
        u_int8  rpc_vers_minor;      /* 01:01 minor version */
        u_int8  PTYPE = fault;       /* 02:01 fault PDU */
        u_int8  pfc_flags;           /* 03:01 flags */
        byte    packed_drep[4];      /* 04:04 NDR data rep format label*/
        u_int16 frag_length;         /* 08:02 total length of fragment */
        u_int16 auth_length;         /* 10:02 length of auth_value */
        u_int32  call_id;            /* 12:04 call identifier */

      /* end common fields */

      /* needed for request, response, fault */

        u_int32  alloc_hint;      /* 16:04 allocation hint */
        p_context_id_t p_cont_id; /* 20:02 pres context, i.e. data rep */

      /* needed for response or fault */

        u_int8  cancel_count      /* 22:01 received cancel count */
        u_int8  reserved;         /* 23:01 reserved, m.b.z. */

      /* fault code  */

        u_int32  status           /* 24:04 run-time fault code or zero */

      /* always pad to next 8-octet boundary */

        u_int8  reserved2[4];     /* 28:04 reserved padding, m.b.z. */

      /* stub data here, 8-octet aligned
               .
               .
               .                          */

      /* optional authentication verifier */
      /* following fields present iff auth_length != 0 */

       auth_verifier_co_t   auth_verifier; /* xx:yy */

} rpcconn_fault_hdr_t;
```

RPC access denied

(msg:" RPC PDU - fault_access_denied response 0x00000005"; flow:to_client, ... ;

content:"|05 00 03|"; offset:0; depth:3;

byte_test:4,=,0x00000005,24,dce;

metadata:service dcerpc; …)

Test the rule set

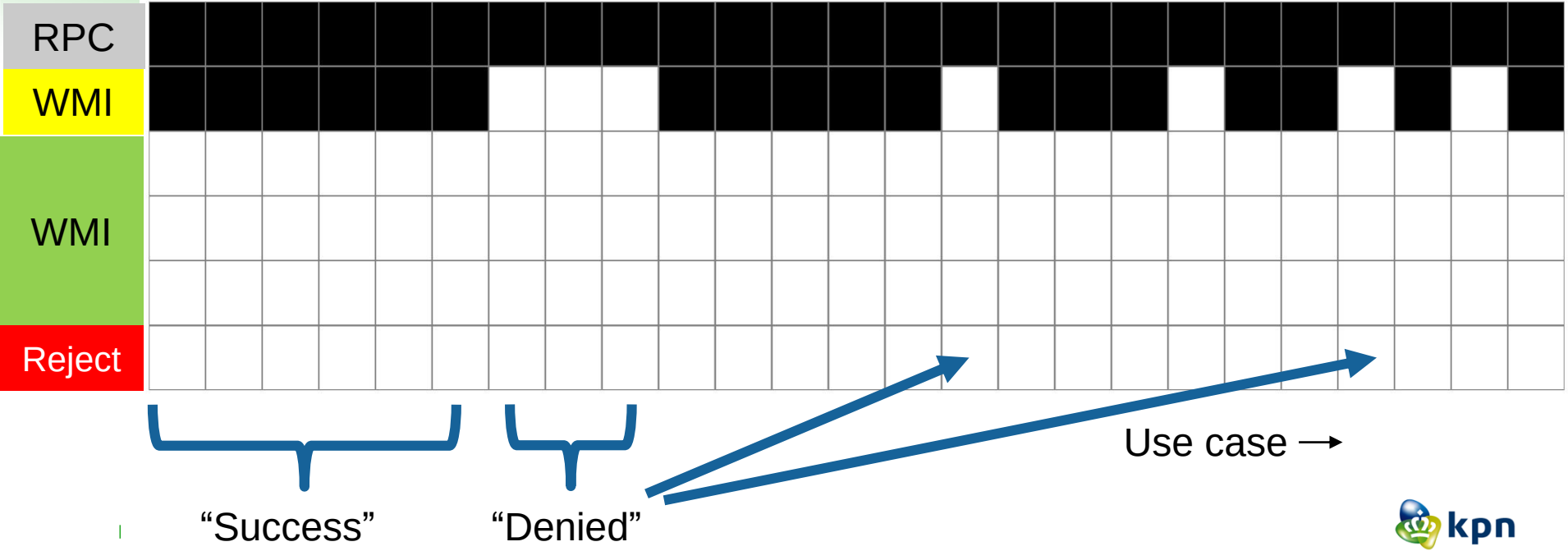
- Engage with system administrators
- They did the heavy lifting!

- Be aware: keep policy implementation up-to-date

Test blueprint

- Targets { Windows 2012R2/2016 }
- Clients { Windows cli:wmic, ps:get-wmiobject, imp:wmiquery, imp:wmiexec }
- Fully privileged/authorized + unprivileged account
- Result from client perspective: success, failure

Visualize results as heatmap



Workaround for reject rule

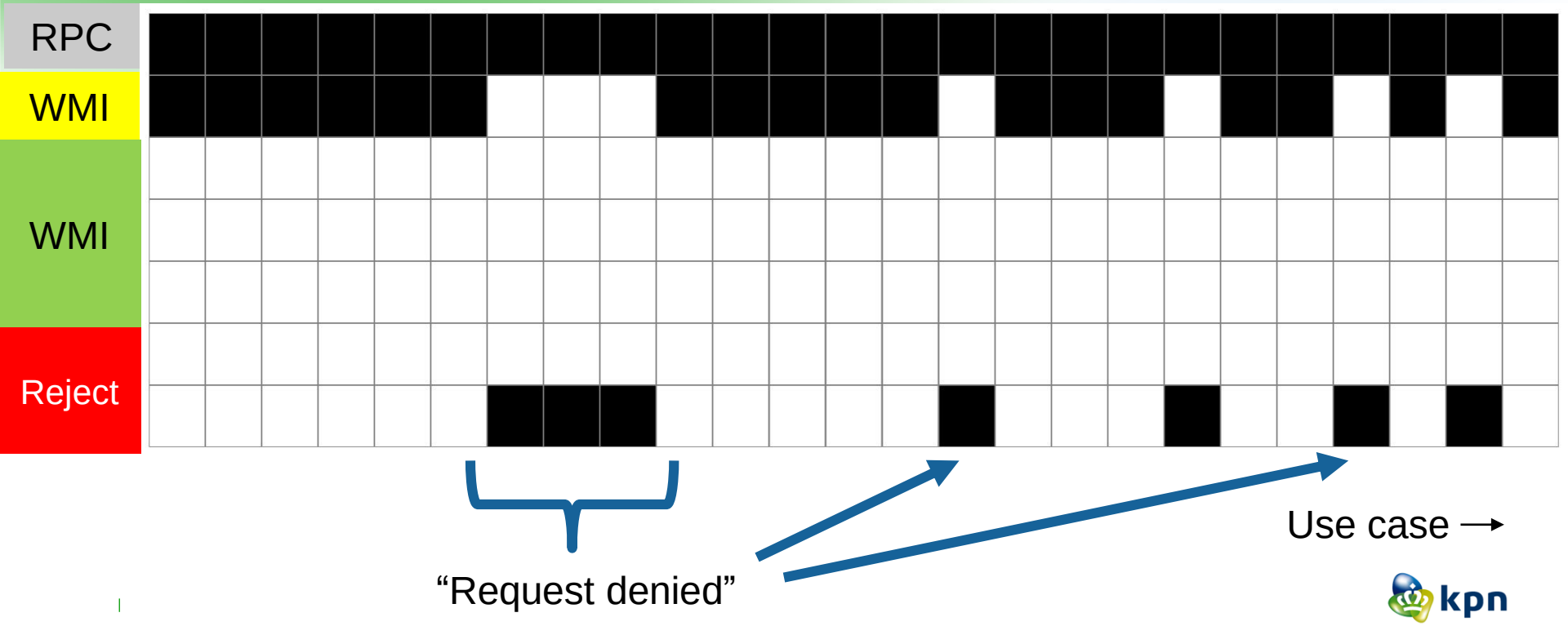
(msg:"RPC generic reject"; content:"|05 00 03|"; offset:0; depth:3;

byte_test:4,=,0x00000005,24,little; …

Note:

- “access denied” common as part of server-client negotation
- better: use as correlation, apply with threshold
- maybe better: indicator higher in protocol stack

Re-test compare both versions reject rule



Up to here

- Improved rule set
- Testing involved both system administrators & redteam
- Rules fire when expected to fire, and when not

Are we done and ready to deploy?

#1 en #2 major concern for a security analyst?

How do we know and find out?

- False positives
- False negatives

Assess the ruleset quality

- From ~~attacker~~ defender perspective
- What means are left for ~~an attacker to evade detection~~ defender to assess the ruleset pro-actively?

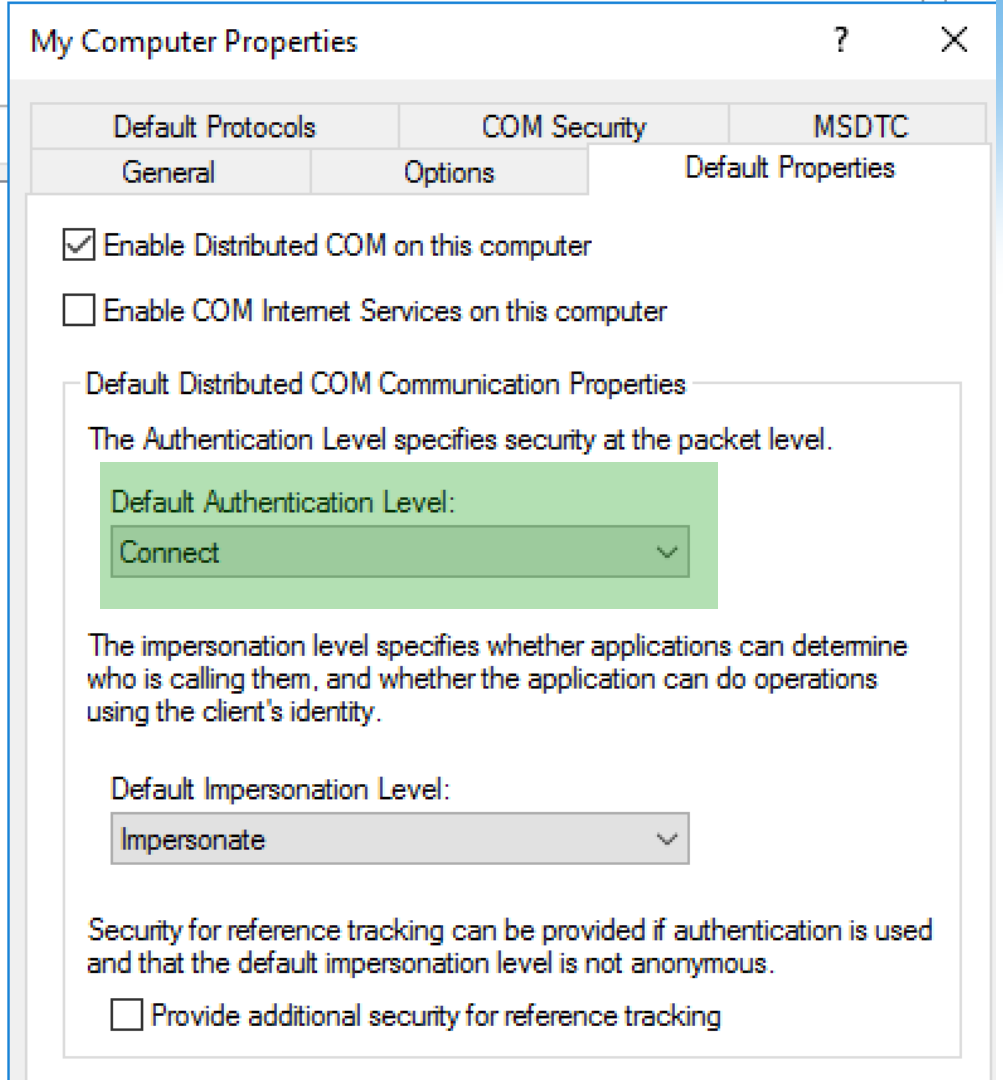
WMI security

- Namespace (S/DACL, securable objects)
- Transport level
  - DCOM/RPC
    - Impersonation
    - Authentication level options
      - Server and client require Mutual agreement
      - None / connect / call / pkt / pktintegrty / pkt_privacy
      - “Privacy”: encrypts argument values

[I] https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rpce/425a7c53-c33a-4868-8e5b-2a850d40dc73

Options and defaults

- Dcomcnfg.exe
  - UI to registry
  - Machine wide
  - Process wide
- Default level: “Connect”



Setup pristine lab evironment

Client / Server

- W 2016 domain controller + member + standalone
- W 10 standalone
- Linux client (impacket)

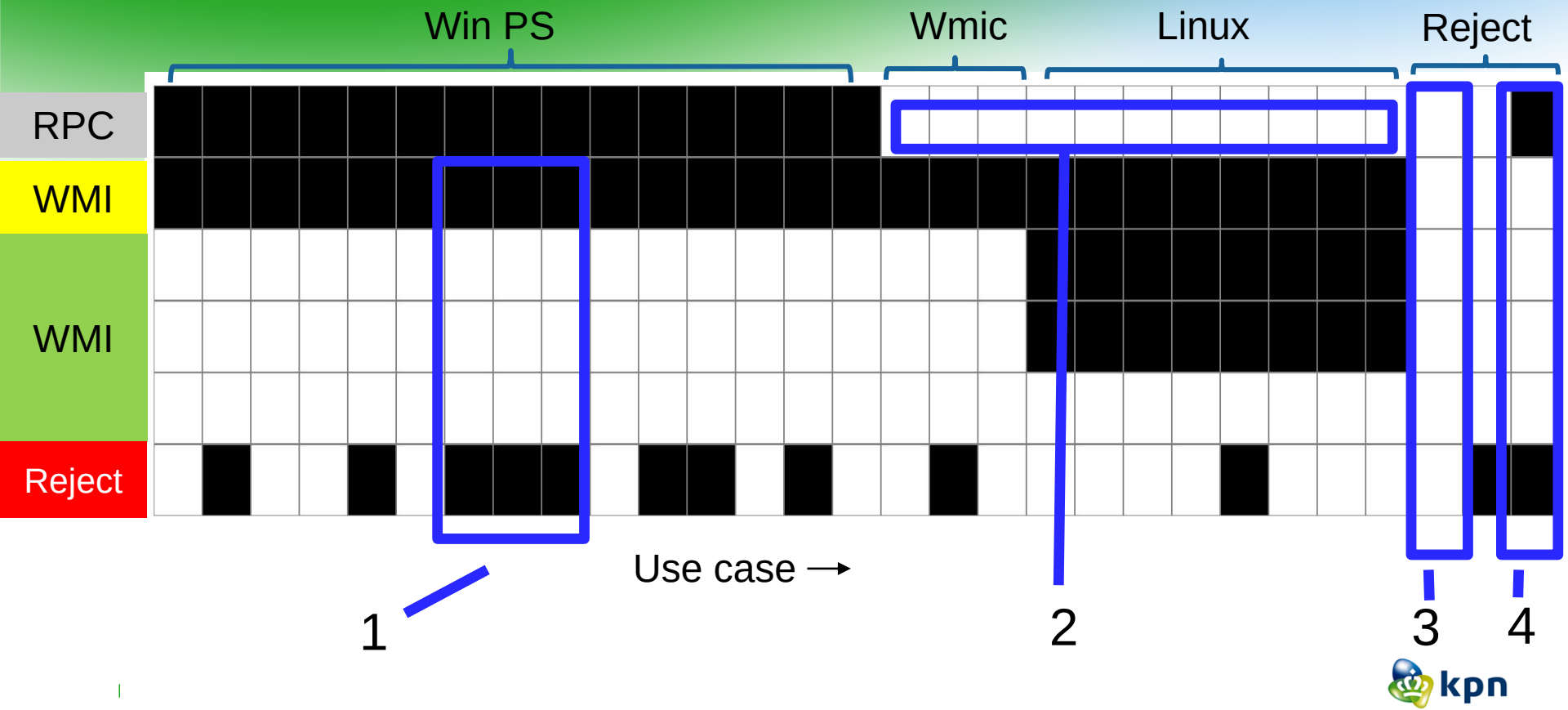
Encryption

- [d] default
- [ec] client only
- [ecs] client + server

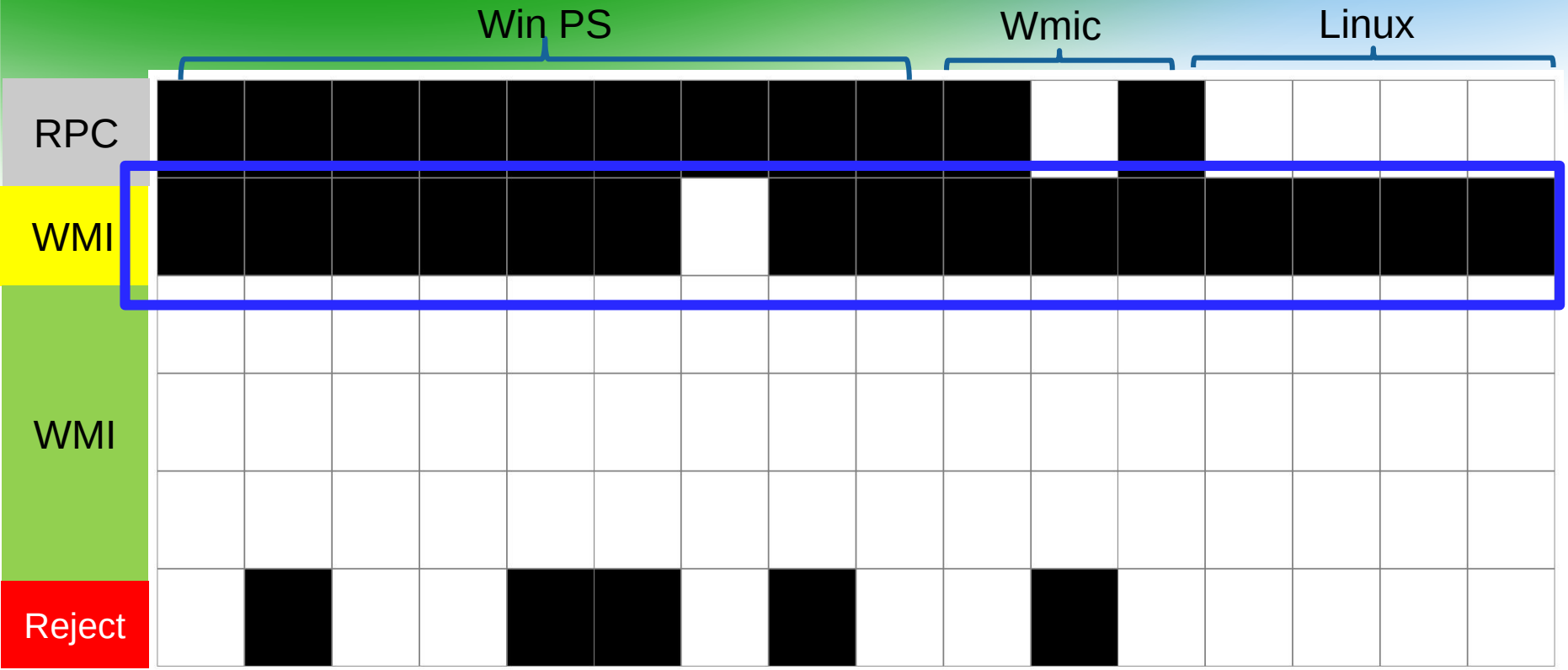
Client methods

1. Get-CimInstance -ClassName Win32_OperatingSystem -CimSession $csd
2. Get-WmiObject win32_computersystem -ComputerName _ -Credential adm
3. wmic /node:<> /user:"administrator" cpu get name
4. wmiquery.py <>/administrator@<> -file wql.file
5. pth-wmic -U Administrator%<> //<> "select Name from Win32_UserAccount"
6. like @1, with invalid password
7. like @ 5, with invalid password
8. like @2, valid credentials, however not authorized

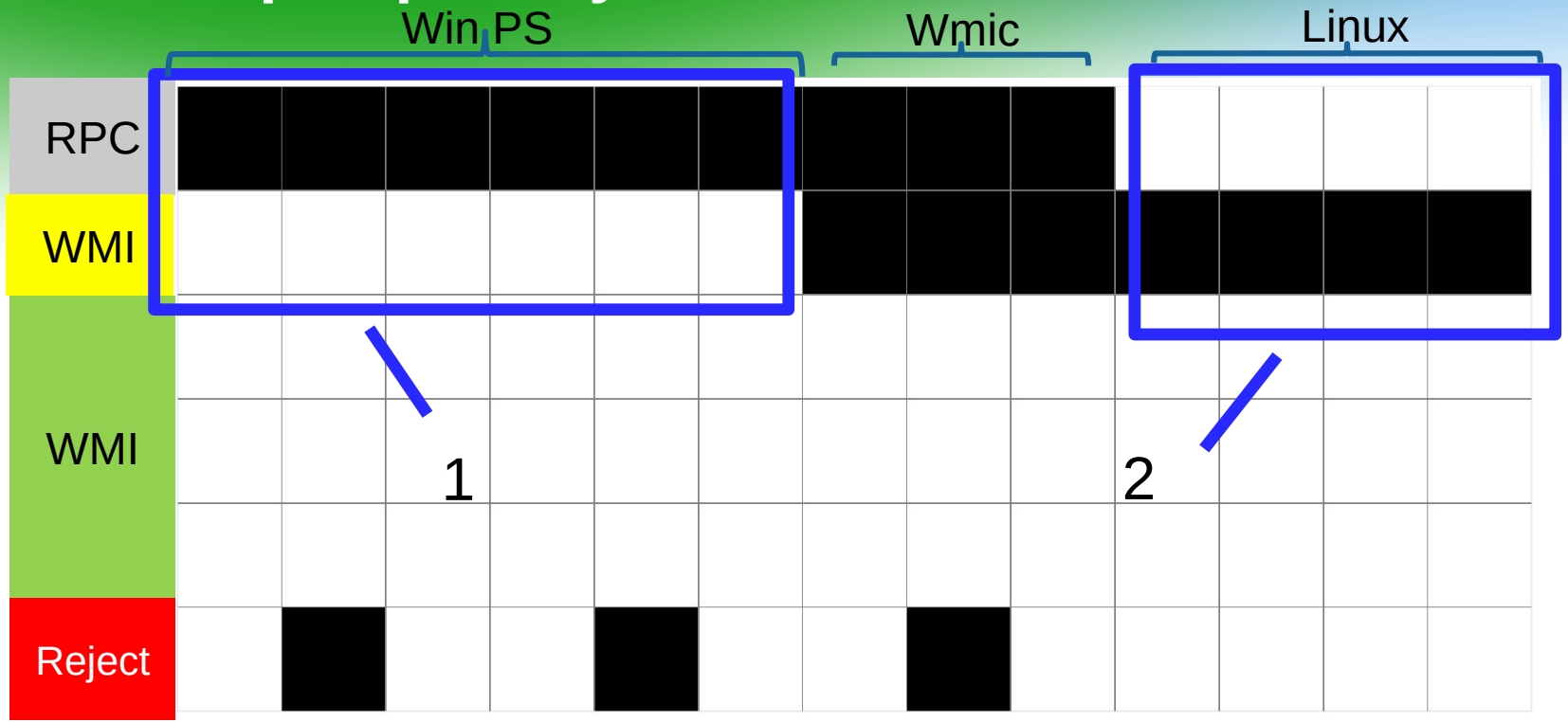
Default



Force clientside encryption (“ec”)



Force pkt_privacy on both sides



Extend the ruleset

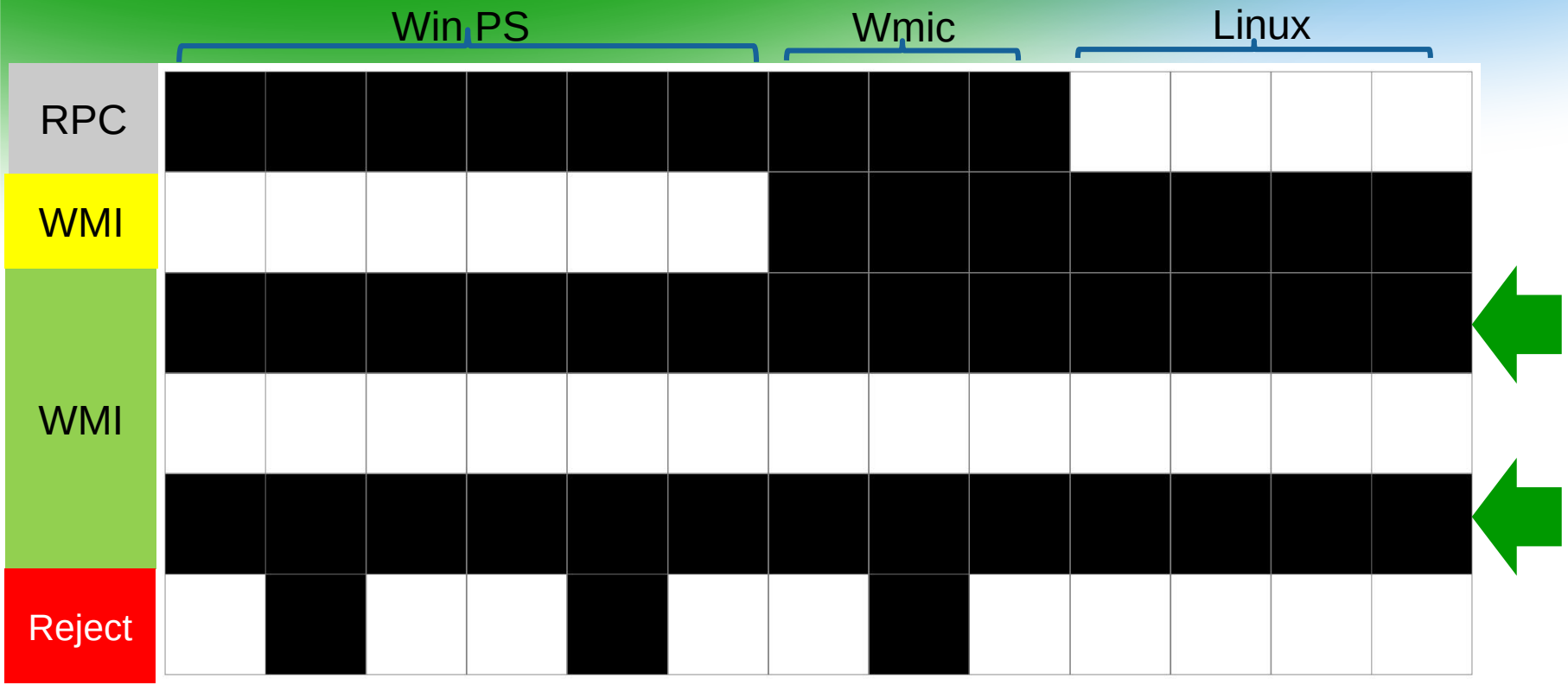
- iwbemlevel1login: f309ad18-d86a-11d0-a075-00c04fb68820

Rule #112 -> "|18 ad 09 f3 6a d8 d0 11 a0 75 00 c0 4f b6 88 20|”

- IWebmServices: 9556dc99-828c-11cf-a37e-00aa003240c7

Rule #115 -> "|99 dc 56 95 8c 82 cf 11 a3 7e 00 aa 00 32 40 c7|”

Re-run with extended rule set



Takeaways

- Network level detection is doable
- Github.com/KPN-CISO/Network-Detection
- Cross-team collaboration is key
- Based on testing so far, happy with FP
- FN a concern

- Future research evasion techniques, improve detection, resolve open ends

Final thoughts…

“ One sunny day
A few lacking rules became a big takeaway
To overcome the annoyance and frustration
With the support of both admins and reds
We fulfilled the promise nothing is beyond our reach
Now it’s time to call on you to have a look and make it better
Administrator, defender or attacker, the role doesn’t matter
Suggesting to combine it with a Belgian beer
Let me say it loud and clear
I feel confident we can work it out together ! “