Orange
Cyberdefense

802.1x NAC & BYPASS TECHNIQUES

Hack in Paris 2017
Valérian LEGRAND

ABOUT

- Valérian LEGRAND, Security consultant and Penetration Tester at Orange CyberDefense
  - Breaking things is my job

- Why this research ?
  - 802.1x often disabled for penetration tests
    - Provides a good excuse for bad devs & admins
    - “Great you found X critical vulnerabilities… but we disabled 802.1x for the penetration test so it’s not that bad !”
  - Needed for specific Red Team engagements

- Also, huge thanks for the help to :
  - Andrei Dumitrescu (twitter : @_dracu_)
  - Quentin Biguenet
  - Florent “KASH” Lalegerie
  - But also : Fabien, Nicolas, Pierre, Simon, Slim, etc…

WHAT WE GONNA TALK ABOUT

Wired 802.1X
How the hell does it work ?

A Brief Overview of 802.1X Bypasses

FENRIR

Goddammit, We Want Shells !

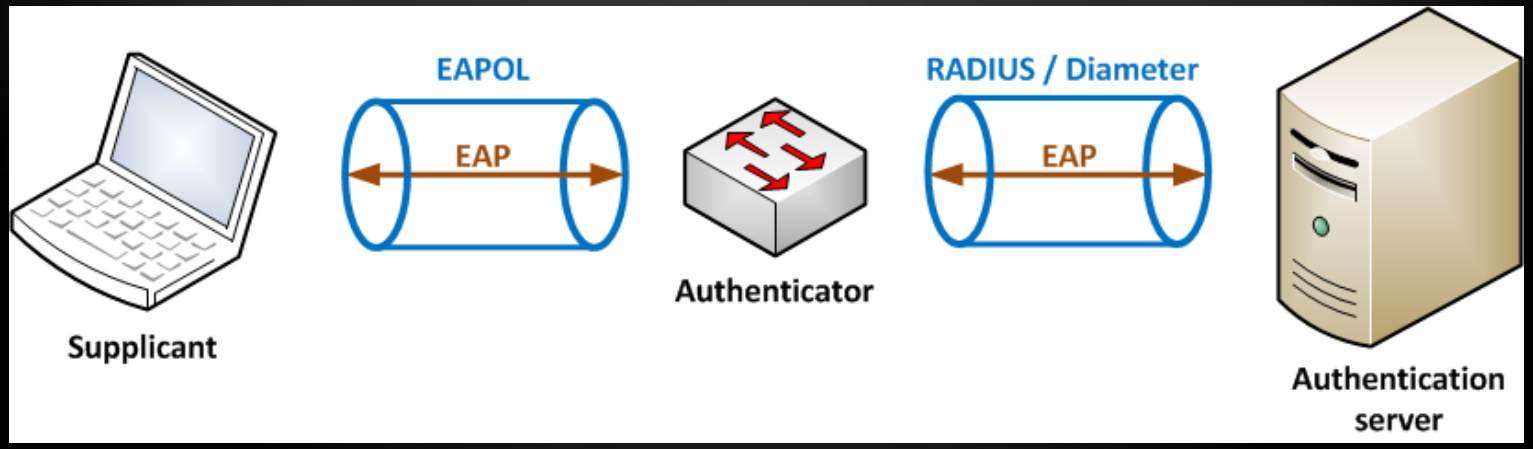
WIRED 802.1X

- IEEE standard originally created in 2001
- **Physical port-based** network access control
  - The new device has to authenticate in order to access the network beyond the switch
- 3 roles involved
  - **SUPPLICANT** : The new device
  - **AUTHENTICATOR** : The switch (or Wireless AP)
  - **AUTHENTICATION SERVER** : The server responsible for checking credentials (Usually a RADIUS server)

THE HAPPY EAP FAMILY

- **EAP** = **E**xtensible **A**uthentication **P**rotocol
  - Defines authentication message formats
  - LOTS of different formats
    - EAP-MD5, EAP-TLS, EAP-TTLS, EAP-PSK, etc…
  - Some are very weak (Seriously, don't use LEAP)

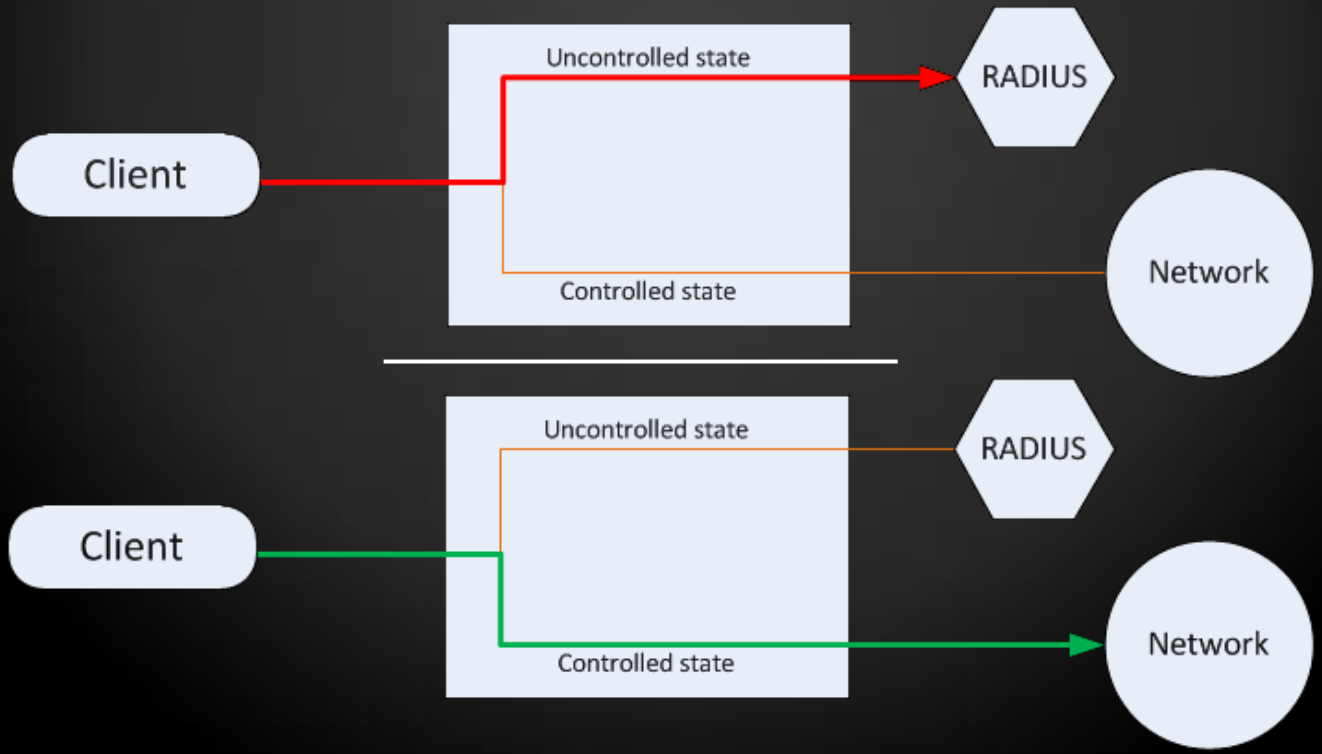
- EAP is NOT a wire protocol
- EAP messages are encapsulated by other protocols
  - **EAPoL** = **EAP o**ver **L**AN
  - **PEAP** = **P**rotected **EAP** (mainly used on Windows systems)
  - …



PORT-BASED ACCESS CONTROL

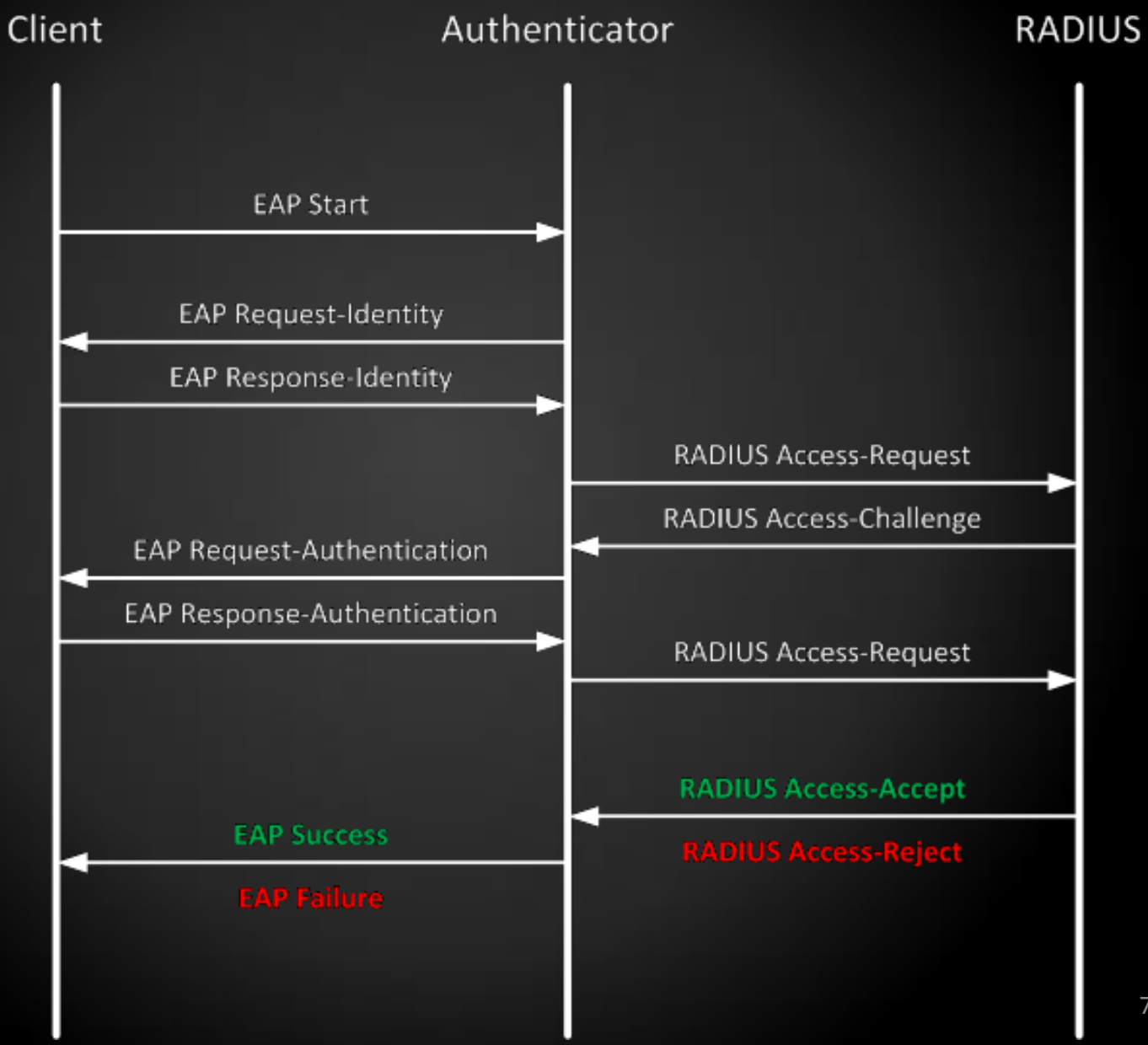
- The **Authenticator** defines **2** logical states per physical port
  - **Uncontrolled** State
  - **Controlled** State

- The **uncontrolled** state allows 802.1x frames only
  - The Authenticator forwards the frames to the Authentication Server
- The **controlled** state acts like a “normal” port
  - The network is fully accessible
  - From this point, **any packet** can go wherever it needs to on the network without authentication !



GIMME ACCESS, YOU SWITCH !

Standard base authentication scheme



A BRIEF OVERVIEW OF 802.1X BYPASSES

BYPASS BY DESIGN

Just a quick note about what is NOT 802.1x protection

- 802.1x only acts as a **gatekeeper**
  - If a device is compromised when already connected to the LAN, 802.1x protection is useless
    - Social engineers don't care about 802.1x (think malicious attachments for example)
  - 802.1x is not a solution to protect a LAN against **BYOD hazards**
    - The compromised device will authenticate against 802.1x as usual
- It is also possible to **retrieve credentials/certificates** on legitimate devices
  - Mimikatz (Benjamin Delpy)
- Note : bruteforce is not possible
  - Temporization rules on authentication server

SOME DEVICES JUST WANT TO SEE THE NETWORK BURN

BYPASS - The easy way :

- Some devices do not support 802.1x
  - You wish they do, but they don't…
  - Usually : **old** devices, **low-grade** equipment (printers), or very **specific** systems (security cameras)
- These devices can be unplugged and their Ethernet port hijacked in order to access the network without the need to authenticate

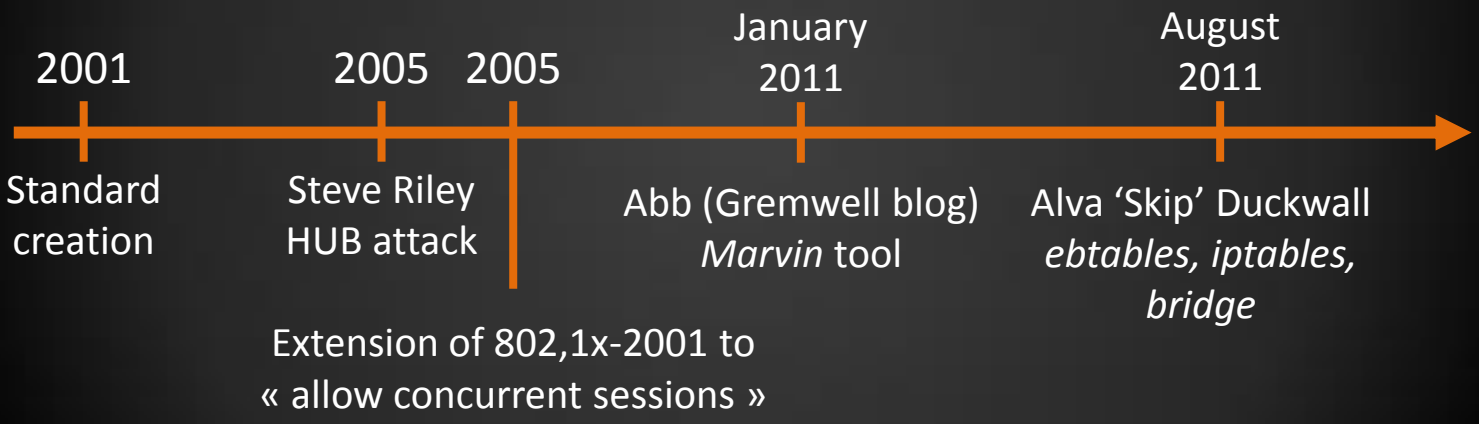
- Solution : **MAC Authentication Bypass (MAB)**
  - (Seriously, who puts "bypass" in the name of a security feature ???)

- MAB uses the device's MAC address to validate its identity
  - The authenticator first tries to authenticate the new device by sending EAP Request-Identity messages
  - After 3 unsuccessful attempts, the authenticator falls back to MAB and sends the device's MAC address to the authentication server

THE REAL BYPASS : TRAFFIC INJECTION

BYPASS - The hard way :

- 802.1x provides Network Access Control
  - It provides authentication over who can access the network
  - It does NOT provide traffic encryption (many people believe it does)
  - It does NOT provide per-packet authentication



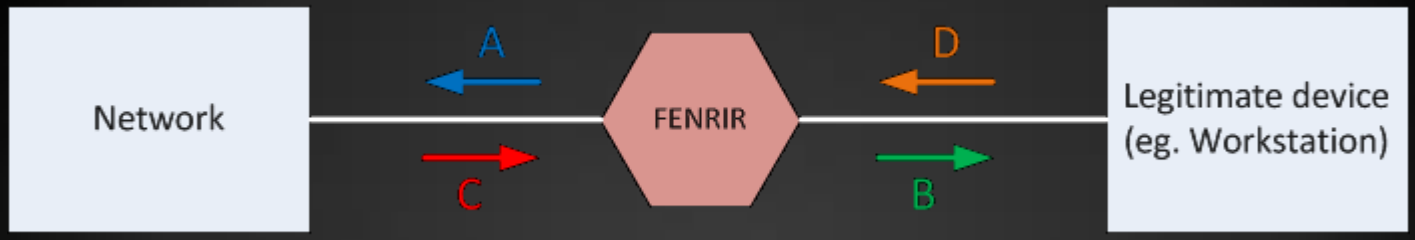
- Traffic Injection : spoof a legitimate and authenticated supplicant's MAC and IP address to fake legitimate packets
  - Still works today in a vast majority of cases !
  - Especially works in traditional Windows environments

FENRIR

FENRIR & TRAFFIC INJECTION

- Traffic Injection is the most reliable technique to **physically** attack a 802.1x network
- This led to the development of **FENRIR**
  - Traffic tapping and injection
  - Stealth
  - Auto-configuration
  - Collision issue avoidance
  - Modularity & extensibility
  - Full control over the traffic
  - Reverse connection capabilities
  - Not developed in Java !
- The goal was to obtain a tool “out-of-the-box” that could be useful during a penetration test (including Red Team)
- Requirements :
  - A laptop with 2 physical interfaces (external netcards work great !)
  - Python & Scapy

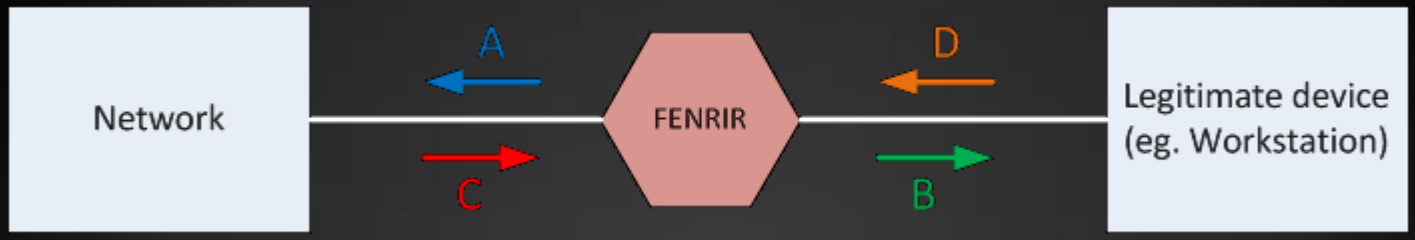
HOW IT WORKS



- We need that :
  - Frames at “A” : appear to be coming from the legitimate host
  - Frames at “B” : appear to be coming from the network
  - Frames at “C” : appear to be addressed to the legitimate host
  - Frames at “D” : appear to be addressed to the network

- FENRIR captures frames on both physical interfaces and rewrites headers to make the FENRIR host disappear
  - Frames from/to the legitimate host are forwarded
    - (You can also do whatever you want to do on them too here)
  - Frames from/to FENRIR are rewritten

HOW IT WORKS



- Step 1 :
  - FENRIR acts as a wire and let the legitimate device authenticate itself to the switch

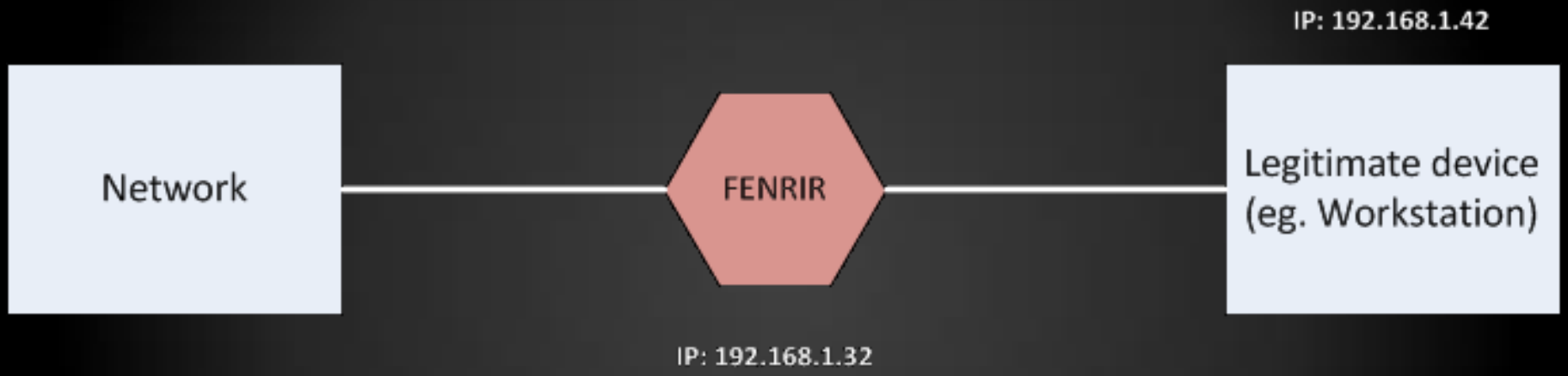
  → Switch's port state changes from uncontrolled to controlled

- Step 1.5 : optional automatic configuration
  - Passive tapping to gather legitimate host's MAC/IP addresses, TTL, etc…

- Step 2 :
  - FENRIR will perform per frame analysis (for legitimate and rogue hosts' frames)
  - Frames from/to rogue host will be rewritten

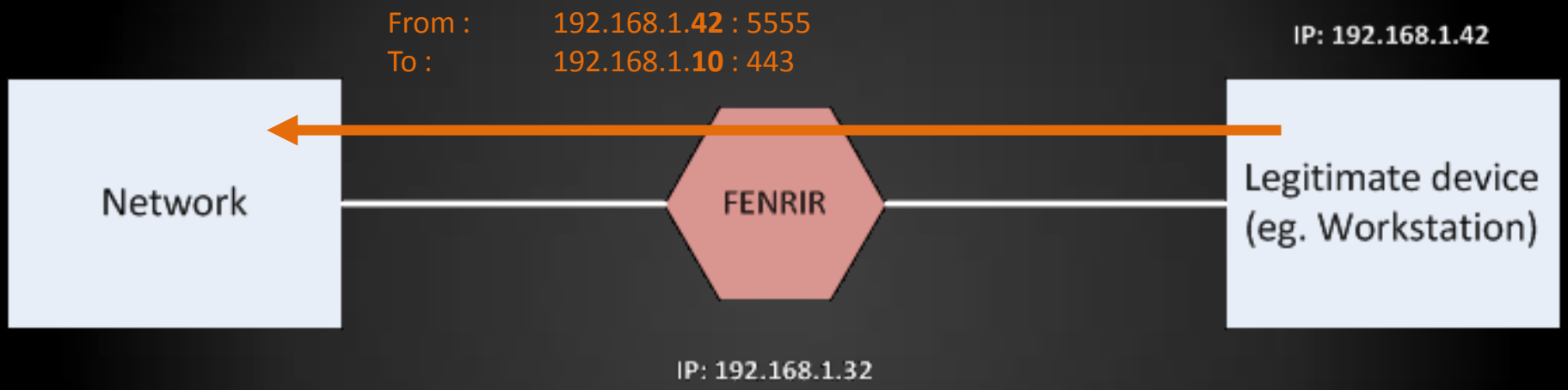
  → We need to keep the legitimate host's network access up in order to bypass periodic re-authentications

HOW IT WORKS



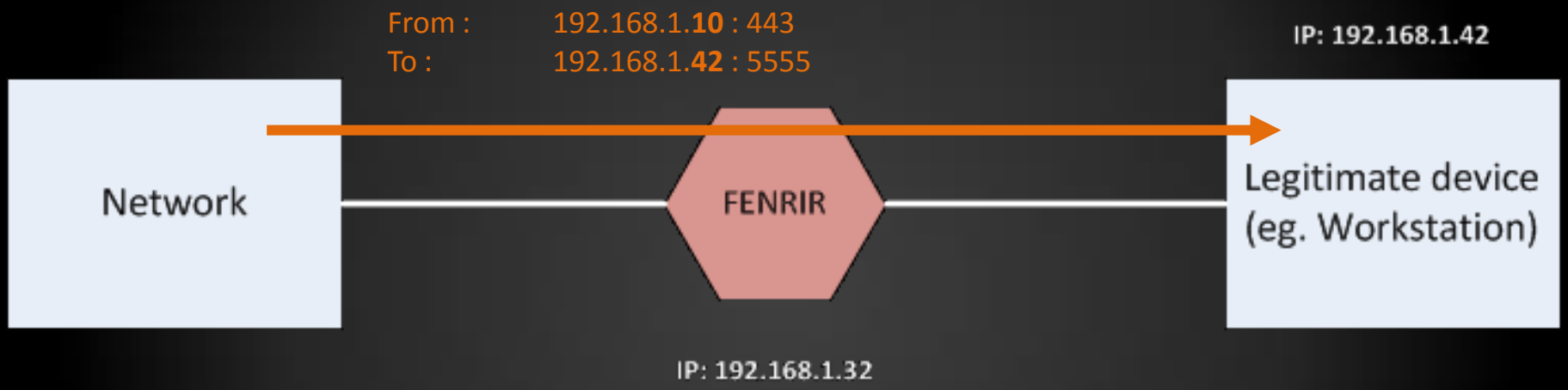
Entry	IP address A	Port A	IP address B	Port B
Entry n°1				
Entry n°2				
Entry n°3				

HOW IT WORKS



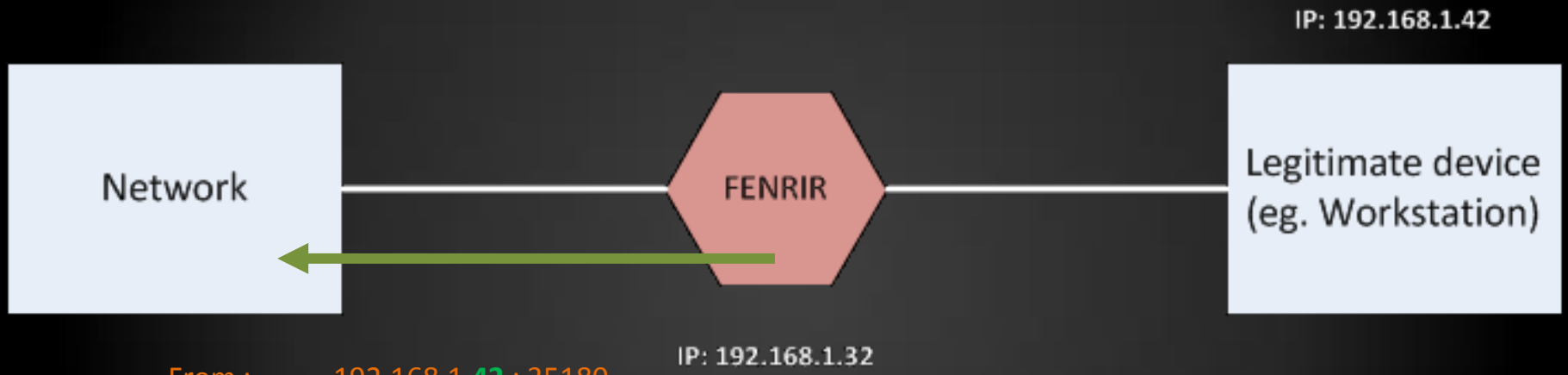
Entry	IP address A	Port A	IP address B	Port B
Entry n°1	192.168.1.42	5555	192.168.1.10	443
Entry n°2				
Entry n°3				

HOW IT WORKS



Entry	IP address A	Port A	IP address B	Port B
Entry n°1	192.168.1.42	5555	192.168.1.10	443
Entry n°2				
Entry n°3				

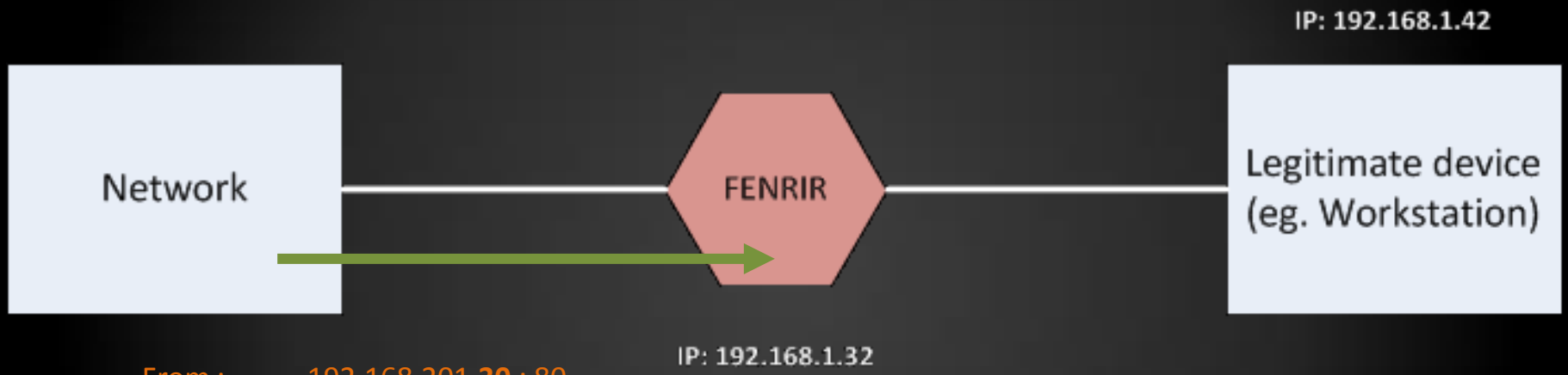
HOW IT WORKS



From : 192.168.1.**42** : 35180
To : 192.168.201.**30** : 80

Entry	IP address A	Port A	IP address B	Port B
Entry n°1	192.168.1.42	5555	192.168.1.10	443
Entry n°2	192.168.1.32	35180	192.168.201.30	80
Entry n°3				

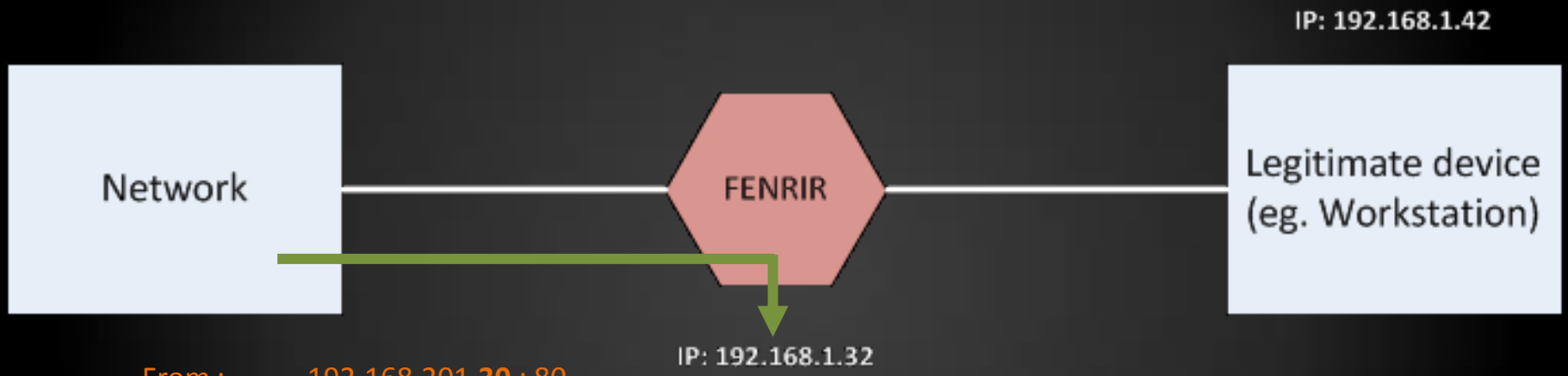
HOW IT WORKS



From : 192.168.201.**30** : 80
To : 192.168.1.**42** : 35180

Entry	IP address A	Port A	IP address B	Port B
Entry n°1	192.168.1.42	5555	192.168.1.10	443
Entry n°2	192.168.1.32	35180	192.168.201.30	80
Entry n°3				

HOW IT WORKS



From : 192.168.201.**30** : 80
To : 192.168.1.**32** : 35180

Entry	IP address A	Port A	IP address B	Port B
Entry n°1	192.168.1.42	5555	192.168.1.10	443
Entry n°2	192.168.1.32	35180	192.168.201.30	80
Entry n°3				

FENRIR - DEMO

GODDAMMIT WE WANT SHELLS

REVERSE CONNECTIONS

- FENRIR provides the possibility to **capture reverse connections** (connections initiated from the network)
  - Useful for :
    - **Reverse shells**
    - **Fake servers** (think Responder for example)
    - …
- **Rules system**, “à la” iptables that can be added/deleted on the fly to allow interception of specific frames

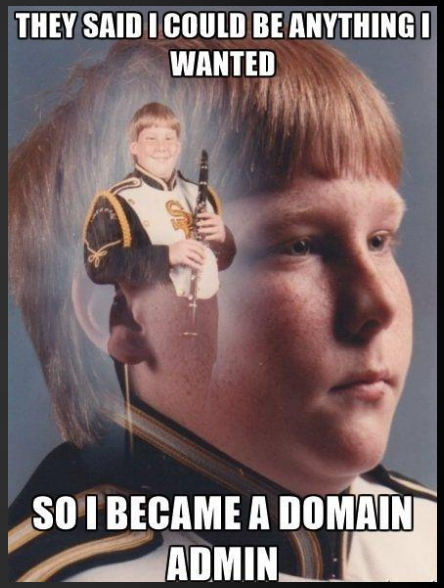
```
Rule 1 :
        port = 137
        type = multi
        proto = IP
Rule 2 :
        port = 5355
        type = multi
        proto = IP
Rule 3 :
        port = 80
        type = multi
        proto = IP
```

GOING WILD

- Once FENRIR is set up, you find yourself in a **perfect MitM spot**
  - A whole new world of network fun !

- **Attack modules ?**
  - Injecting malicious exe on the wire
  - Modifying network traffic for the legitimate host
  - Responder
  - …

- Classic attacks
  - FENRIR works with all **TCP/UDP** tools (and new protocols can be added)
    - nmap
    - netcat
    - Metasploit
    - CrackmapExec
    - Empire
    - …



GIMME SHELLS DEMO

TAKE AWAYS

- 802.1x protection is **great** (really it is), but is just **a brick in the wall**
- **Ways of bypassing it exist** - Do not consider your network secure because you implemented it ! 802.1x =/= physical access protection
- 802.1x protects the door but **not what goes through**
  - No encryption

(Does anyone realize this kid is eating sand ?!?)

https://github.com/Orange-Cyberdefense/fenrir