TALLER HE

PRÁCTICA Auditar contrasinal Wi-Fi WPA2 (PSK)

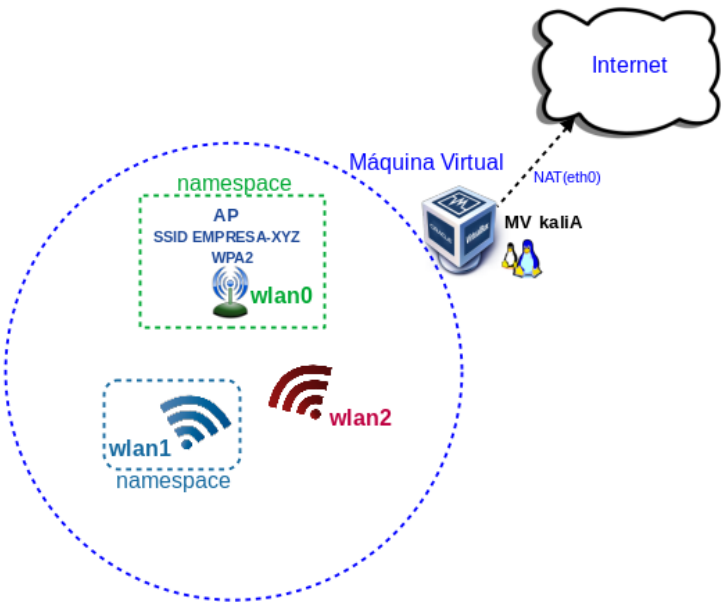
Apelidos	Nome

ESCENARIO: AP → SSID EMPRESA-XYZ → WPA2 (PSK)

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

mac80211_hwsim
- radios=3 → wlan0, wlan1, wlan2
- wlan0 → AP (hostapd, ip netns)
- wlan1 → Cliente
- wlan2 → unknown (tools suite: aircrack-ng →auditar handshake)

Namespaces
- phy0 → wlan0 → AP aillado
- phy1 → wlan1 → cliente autenticado aillado



Material necesario	Práctica: Auditar contrasinal Wi-Fi WPA2 (PSK)
■ Host alumnado ■ Máquina virtual GNU/Linux Kali ■ [1] aircrack-ng ■ [2] wordlists ■ [3] ISO descarga GNU/Linux Kali ■ [4] tonyharris.io ■ [5] mac80211_hwsim ■ [6] hostapd and wpa_supplicant ■ [7] WEP, WPA, WPA2 y WPA3: diferencias y explicación ■ [8] WPA3: ¿Qué es y en qué se diferencia de la WPA2? ■ [9] namespaces	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 e aillar ■ Conectar co cliente simulado e aillar ■ Investigar co cliente unknown(airodump-ng) →desconectar cliente simulado → capturar handshake →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.

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 [3]
v. Rede: 1 tarxeta en modo NAT *(as wlanX serán simuladas por mac80211_hwsim)*
vi. Nome: Practica-Kali-Auditar-PSK

(b) Executar nunha consola (consola1):

```
$ setxkbmap es #Configurar teclado en español
$ ip addr show eth0 #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=3 #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 e wlan2
$ 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 e hwsim0
```

(c) Rol interfaces Wi-Fi:

Imos empregar:

i. wlan0 para AP WPA2 (PSK) (shell bash consola1)
ii. wlan1 para o cliente que se conecta a AP (shell bash consola2)
iii. wlan2 como un cliente que non sabe o contrasinal para conectarse ao AP (shell bash consola3).

(d) Configura wlan0 como AP e aillala do resto de interfaces: na consola1

`$ sudo su -` #Acceder á consola de root(administrador) a través dos permisos configurados co comando sudo (/etc/sudoers, visudo)

`# apt update && apt -y install hostapd` #Actualizar o listado de paquetes dos repositorios (/etc/apt/sources.list, /etc/apt/sources.list.d/) e se este comando ten éxisto ($?=0) faise o segundo comando, o cal instálase o paquete hostapd. Co parámetro -y automaticamente asumimos yes a calquera pregunta que ocorra na instalación do paquete.

`# ip netns add wifi_ap_wlan0` #Crear un novo **namespace de rede** chamado **wifi_ap_wlan0** no sistema. Un **namespace de rede** é unha funcionalidade de Linux que permite crear espazos de rede illados, onde cada espazo pode ter as súas propias interfaces de rede, routers e configuracións independentes. Neste caso, o nome **wifi_ap_wlan0** identifica o namespace que se está a crear, e permitirá que as interfaces de rede, como **wlan0**, operen de forma illada dentro dese namespace, sen interferir co resto das interfaces ou redes do sistema. Isto é útil para crear ambientes de proba ou para a xestión de múltiples redes illadas na mesma máquina sen que compartan recursos.

`# ip netns exec wifi_ap_wlan0 bash` #Executar unha instancia do **shell bash** dentro do namespace de rede chamado **wifi_ap_wlan0**. Isto significa que, ao executar o comando, ábrese un terminal onde as interfaces de rede e as rutas de rede serán específicas para o namespace **wifi_ap_wlan0**, permitindo interactuar con redes ou dispositivos dentro dese espazo illado. A principal utilidade é traballar cunha rede virtual illada sen afectar a rede principal do sistema, o que é útil para tarefas como probar configuracións de rede ou simular entornos de rede controlados.

`# echo $BASHPID || echo $$` #Conseguir o PID deste proceso (shell bash)

`# echo $BASHPID > /tmp/consola1-pid.txt || echo $$ > /tmp/consola1-pid.txt` #Conseguir o PID deste proceso (shell bash) e gardar o seu valor no ficheiro consola1-pid.txt en /tmp

(e) Abrir outra consola(consola2) e executar:

`$ PID=$(cat /tmp/consola1-pid.txt)` #Xerar a variable ${PID} que garda o valor do contido do ficheiro /tmp/consola1-pid.txt, é dicir, garda o identificador de namespace de rede no sistema.

`$ sudo iw phy phy0 set netns ${PID}` #Asignar a interface de radio **phy0** (que representa a primeira interfaz de hardware de rede Wi-Fi) ao namespace de rede identificado por **${PID}**. Isto significa que a interface **phy0** pasará a formar parte do espazo de rede illado co id **${PID}**, o que permite que a súa configuración e operación (como a asignación de IPs ou a conexión a redes Wi-Fi) se realicen de maneira independente das outras interfaces ou do namespace principal do sistema. Este comando é útil cando se traballa con múltiples namespaces de rede para crear entornos de rede separados e illados, por exemplo, en simulacións ou probas de redes.

`$ ip link` #Veremos que agora a interface wlan0 non aparece. Isto é debido a que está aillada do proceso deste shell bash e está asignada ao namespace definido anteriormente.

(f) Executar na consola1:

```
# echo 'interface=wlan0
driver=nl80211
country_code=ES
ssid=EMPRESA-XYZ
channel=0
hw_mode=b
wpa=2
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP CCMP
wpa_passphrase=spongebob19
auth_algs=3
beacon_int=100' > wpa-psk.conf
```

#Este comando crea un ficheiro de configuración chamado **wpa-psk.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=0: Define o canal a usar como **0**, o que significa que o sistema seleccionará automaticamente un canal dispoñible.
hw_mode=b: Establece o modo de hardware como **b** (Wi-Fi 802.11b).
wpa=2: Activa **WPA2** (Wi-Fi Protected Access 2) como protocolo de seguridade.
wpa_key_mgmt=WPA-PSK: Define o método de xestión de chave como **WPA-PSK** (Pre-Shared Key).
wpa_pairwise=TKIP CCMP: Establece os algoritmos de cifrado como **TKIP** e **CCMP**.
wpa_passphrase=spongebob19: Define a clave de acceso (**spongebob19**) para a rede Wi-Fi.
auth_algs=3: Permite os dous métodos de autenticación **Open System** e **Shared Key**.
beacon_int=100: Establece o intervalo de beacons (paquetes de publicidade de rede) en **100 ms**.

`# hostapd wpa-psk.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 **wpa-psk.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 WPA-PSK, 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.

```
wlan0: interface state UNINITIALIZED->COUNTRY_UPDATE
ACS: Automatic channel selection started, this may take a bit
wlan0: interface state COUNTRY_UPDATE->ACS
wlan0: ACS-STARTED
wlan0: ACS-COMPLETED freq=2437 channel=6
wlan0: interface state ACS->ENABLED
wlan0: AP-ENABLED
```

(g) Configura wlan1 como cliente autentificado e aillala do resto de interfaces. Executar na consola2:

`$ sudo su -` #Acceder á consola de root(administrador) a través dos permisos configurados co comando sudo (/etc/sudoers, visudo)

`# ip netns add wifi_cliente_wlan1` #Crear un novo **namespace de rede** chamado wifi_cliente_wlan1 no sistema. Un **namespace de rede** é unha funcionalidade de Linux que permite crear espazos de rede illados, onde cada espazo pode ter as súas propias interfaces de rede, routers e configuracións independentes. Neste caso, o nome wifi_ap_wlan0 identifica o namespace que se está a crear, e permitirá que as interfaces de rede, como **wlan1**, operen de forma illada dentro dese namespace, sen interferir co resto das interfaces ou redes do sistema. Isto é útil para crear ambientes de proba ou para a xestión de múltiples redes illadas na mesma máquina sen que compartan recursos.

`# ip netns exec wifi_cliente_wlan1 bash` #Executar unha instancia do **shell bash** dentro do namespace de rede chamado wifi_cliente1_wlan1. Isto significa que, ao executar o comando, ábrese un terminal onde as interfaces de rede e as rutas de rede serán específicas para o namespace wifi_cliente_wlan1, permitindo interactuar con redes ou dispositivos dentro dese espazo illado. A principal utilidade é traballar cunha rede virtual illada sen afectar a rede principal do sistema, o que é útil para tarefas como probar configuracións de rede ou simular entornos de rede controlados.

`# echo $BASHPID || echo $$` #Conseguir o PID deste proceso (shell bash)

`# echo $BASHPID > /tmp/consola2-pid.txt || echo $$ > /tmp/consola2-pid.txt` #Conseguir o PID deste proceso (shell bash) e gardar o seu valor no ficheiro consola2-pid.txt en /tmp

(h) Abrir outra consola(consola3) e executar:

`$ PID=$(cat /tmp/consola2-pid.txt)` #Xerar a variable ${PID} que garda o valor do contido do ficheiro /tmp/consola2-pid.txt, é dicir, garda o identificador de namespace de rede no sistema.

`$ sudo iw phy phy1 set netns ${PID}` #Asignar a interface de radio phy1 (que representa a segunda interfaz de hardware de rede Wi-Fi) ao namespace de rede identificado por **${PID}**. Isto significa que a interface phy1 pasará a formar parte do espazo de rede illado co id **${PID}**, o que permite que a súa configuración e operación (como a asignación de IPs ou a conexión a redes Wi-Fi) se realicen de maneira independente das outras interfaces ou do namespace principal do sistema. Este comando é útil cando se traballa con múltiples namespaces de rede para crear entornos de rede separados e illados, por exemplo, en simulacións ou probas de redes.

`$ ip link` #Veremos que agora a interface wlan1 non aparece. Isto é debido a que está aillada do proceso deste shell bash e está asignada ao namespace definido anteriormente.

(i) Na consola2 executar:

```
# echo 'network={
ssid="EMPRESA-XYZ"
key_mgmt=WPA-PSK
psk="spongebob19"
 }' >> wpa_supplicant.conf
```
#Este comando engade unha configuración a un ficheiro chamado **wpa_supplicant.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=WPA-PSK**: Establece o método de xestión de chaves como **WPA-PSK** (Pre-Shared Key).
- **psk="spongebob19"**: Define a clave de acceso á rede Wi-Fi (**spongebob19**).
- **}**: Pecha a sección de configuración da rede.

`# wpa_supplicant -B -i wlan1 -c wpa_supplicant.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.conf**: Indica o ficheiro de configuración que contén os parámetros da rede Wi-Fi (como SSID, clave, etc.), neste caso **wpa_supplicant.conf**.

```
wlan0: STA 02:00:00:00:01:00 IEEE 802.11: authenticated

wlan0: STA 02:00:00:00:01:00 IEEE 802.11: associated (aid 1)

wlan0: AP-STA-CONNECTED 02:00:00:00:01:00

wlan0: STA 02:00:00:00:01:00 RADIUS: starting accounting session 2B69A95944FF01FC

wlan0: STA 02:00:00:00:01:00 WPA: pairwise key handshake completed (RSN)

wlan0: EAPOL-4WAY-HS-COMPLETED 02:00:00:00:01:00
```

(j) Executar na consola3:

`$ sudo su -` #Acceder á consola de root(administrador) a través dos permisos configurados co comando sudo (/etc/sudoers, visudo)

`# airmon-ng check kill` #Deter calquera proceso que poida interferir co funcionamento das ferramentas de auditoría Wi-Fi, como **airodump-ng** ou **aireplay-ng**, cando unha interface está en modo monitor. Estes procesos inclúen xestores de rede (como **NetworkManager** ou **wpa_supplicant**) que automaticamente configuran as interfaces de rede, podendo interferir co modo monitor. Deténdoos, **airmon-ng** garante que a interface poida funcionar correctamente en modo monitor sen interrupcións.

`# airmon-ng start wlan2` #Habilitar o modo monitor na interface de rede sen fíos chamada **wlan2**, permitindo que esta capture todos os paquetes Wi-Fi que estean no aire, sen estar asociada a unha rede específica. O **modo monitor** é esencial para tarefas de auditoría ou análise de redes Wi-Fi, xa que permite escoitar o tráfico de calquera dispositivo na mesma canle sen necesidade de estar conectado. Ademais, este comando crea unha nova interface virtual (xeralmente chamada **wlan2mon**) asociada á interface orixinal para usar en operacións de monitoraxe.

`# airodump-ng wlan2mon` #Iniciar a ferramenta **airodump-ng** para capturar paquetes de redes Wi-Fi, utilizando a interface **wlan2mon**, que debe estar en modo monitor. Este comando permite escanear e listar todas as redes Wi-Fi dispoñibles no rango da interface, mostrando información como o nome das redes (**SSID**), os enderezos MAC dos puntos de acceso (**BSSID**), os canais que están a usar, o tipo de cifrado (WPA, WPA2, etc.), e unha lista de clientes conectados a esas redes. É unha ferramenta comumente utilizada para auditorías de seguridade en redes sen fíos.

`# Ctrl^C` #Envíar unha sinal **SIGINT** (Interrupt) ao proceso en execución no terminal actual, indicándolle que debe deter a súa execución. Esta combinación de teclas úsase para interromper ou finalizar procesos que están en curso, como scripts, programas ou comandos que se están executando.

`# mkdir capturas && airodump-ng wlan2mon -c 6 -w capturas/cap` #Crear un directorio chamado **capturas** e logo executa **airodump-ng** na interface **wlan2mon** (configurada en modo monitor) para capturar paquetes Wi-Fi no canal **6***, gardando os datos capturados no directorio **capturas** co prefixo de ficheiro **cap**. Como resultado, os ficheiros xerados (por exemplo, **cap-01.cap**) conterán os paquetes capturados, útiles para análises ou auditorías de redes sen fíos.

***(sustituír polo canal que está a usar o AP →** `wlan0: ACS-COMPLETED freq=2437 channel=6`**)**

(k) Voltar a conectar o cliente1. Executar na consola2:

`# pkill -f wpa_supplicant || true` #Eliminar os procesos `wpa_supplicant` existentes.
`# wpa_supplicant -B -i wlan1 -c wpa_supplicant.conf` #Iniciar o proceso **wpa_supplicant**, que é usado para xestionar a conexión dunha interface Wi-Fi a unha rede inalámbrica. A opción **-B** executa o proceso en segundo plano (background), **-i wlan1** especifica que a interface Wi-Fi a utilizar é **wlan1**, e **-c wpa_supplicant.conf** indica que o ficheiro de configuración a usar é **wpa_supplicant.conf**, que contén os detalles de autenticación e parámetros da rede, como o SSID, método de cifrado e contrasinal. Este comando configura e conecta a interface Wi-Fi a unha rede segundo a configuración proporcionada.

(l) Consola3. Unha vez capturado o handshake:

`CH  6 ][ Elapsed: 48 s ][ 2026-01-13 17:24 ][ WPA handshake: AA:BB:CC:DD:EE:FF` ← ***BSSID***

Executar na consola3:

`# Ctrl^C` #Envíar unha sinal **SIGINT** (Interrupt) ao proceso en execución no terminal actual

Auditar contrasinal

`# 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 capturas/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.

```
                              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
```

(2) Contrasinal atopado → `spongebob19`