ANTECEDENTES

WPA es un protocolo más seguro que WEP a la hora de establecer la comunicación Punto de Acceso (router casero) - Máquina (pc con dispositivo wifi).

La prueba la he realizado con los siguientes drivers:

Driver	Tipo de dispositivo
p54pci	pcmcia
ipw2200	integrado

1. (Todo como root) aptitude update && aptitude install wpasupplicant

2. Establecer en el router la contraseña que queramos. Por ejemplo se puede generar una desde aquí:

http://www.kurtm.net/wpa-pskgen

Seleccionar 'Maximum WPA Security (63 characters)' y pulsar 'generate'. Copiar y pegar la cadena de texto en la casilla adecuada del router, en mi caso:

Network Name (SSID):	mi_nombre_de_red
Select Security Option	WPA Mixed Mode
Select Authentication Method:	PSK (Pre Shared Key
WPA Pass Phrase:	mi_clave_de_63_dígitos

3. Para que se asocie a esa red en cada arranque, editar /etc/network/interfaces y, suponiendo que la interfaz inalámbrica sea eth1, dejarlo tal que así:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug eth1
iface eth1 inet dhcp



wpa-ssid mi_nombre_de_red
wpa-passphrase mi_clave_de_63_dígitos
wpa-key-mgmt WPA-PSK
wpa-pairwise TKIP CCMP
wpa-group TKIP CCMP
wpa-proto WPA RSN
```

NOTA Probar el tema dhcp, porque yo lo tengo con ip estática, pero supongo que funciona igual

4. Reiniciar el equipo y listo

Si queremos asociarnos a mano:

1. Seguir los pasos 1 y 2 anteriormente especificados

2. Crear el fichero /etc/wpa_supplicant/wpa_supplicant.conf, con el siguiente contenido:

```
##### Example wpa_supplicant configuration file
###############################
# Empty lines and lines starting with # are ignored

# NOTE! This file may contain password information and should probably be
made
# readable only by root user on multiuser systems.

# global configuration (shared by all network blocks)
#
# Interface for separate control program. If this is specified,
wpa_supplicant
# will create this directory and a UNIX domain socket for listening to
requests
# from external programs (CLI/GUI, etc.) for status information and
# configuration. The socket file will be named based on the interface name,
so
# multiple wpa_supplicant processes can be run at the same time if more than
# one interface is used.
# /var/run/wpa_supplicant is the recommended directory for sockets and by
# default, wpa_cli will use it when trying to connect with wpa_supplicant.
ctrl_interface=/var/run/wpa_supplicant

# Access control for the control interface can be configured by setting the
# directory to allow only members of a group to use sockets. This way, it is
# possible to run wpa_supplicant as root (since it needs to change network
# configuration and open raw sockets) and still allow GUI/CLI components to
be
# run as non-root users. However, since the control interface can be used to
# change the network configuration, this access needs to be protected in
many
# cases. By default, wpa_supplicant is configured to use gid 0 (root). If
you
# want to allow non-root users to use the control interface, add a new group
# and change this value to match with that group. Add users that should have
# control interface access to this group. If this variable is commented out
or
# not included in the configuration file, group will not be changed from the
# value it got by default when the directory or socket was created.
#
# This variable can be a group name or gid.
#ctrl_interface_group=wheel
#ctrl_interface_group=0
```

```
# IEEE 802.1X/EAPOL version
# wpa_supplicant was implemented based on IEEE 802-1X-REV-d8 which defines
# EAPOL version 2. However, there are many APs that do not handle the new
# version number correctly (they seem to drop the frames completely). In
order
# to make wpa_supplicant interoperate with these APs, the version number is
set
# to 1 by default. This configuration value can be used to set it to the new
# version (2).
eapol_version=1

# AP scanning/selection
# By default, wpa_supplicant requests driver to perform AP scanning and then
# uses the scan results to select a suitable AP. Another alternative is to
# allow the driver to take care of AP scanning and selection and use
# wpa_supplicant just to process EAPOL frames based on IEEE 802.11
association
# information from the driver.
# 1: wpa_supplicant initiates scanning and AP selection
# 0: driver takes care of scanning, AP selection, and IEEE 802.11
association
#    parameters (e.g., WPA IE generation); this mode can also be used with
#    non-WPA drivers when using IEEE 802.1X mode; do not try to associate
with
#    APs (i.e., external program needs to control association)
# 2: like 0, but associate with APs using security policy and SSID (but not
#    BSSID); this can be used, e.g., with ndiswrapper and NDIS driver to
#    enable operation with hidden SSIDs and optimized roaming; in this mode,
#    only the first network block in the configuration file is used and this
#    configuration should have explicit security policy (i.e., only one
option
#    in the lists) for key_mgmt, pairwise, group, proto variables
ap_scan=1

# EAP fast re-authentication
# By default, fast re-authentication is enabled for all EAP methods that
# support it. This variable can be used to disable fast re-authentication.
# Normally, there is no need to disable this.
fast_reauth=1

# network block
#
# Each network (usually AP's sharing the same SSID) is configured as a
separate
# block in this configuration file. The network blocks are in preference
order
# (the first match is used).
#
# network block fields:
#
# ssid: SSID (mandatory); either as an ASCII string with double quotation or
#  as hex string; network name
#
# scan_ssid:
#  0 = do not scan this SSID with specific Probe Request frames (default)
#  1 = scan with SSID-specific Probe Request frames (this can be used to
#      find APs that do not accept broadcast SSID or use multiple SSIDs;
#      this will add latency to scanning, so enable this only when needed)
#
# bssid: BSSID (optional); if set, this network block is used only when
#  associating with the AP using the configured BSSID
#
# priority: priority group (integer)
# By default, all networks will get same priority group (0). If some of the
# networks are more desirable, this field can be used to change the order in
# which wpa_supplicant goes through the networks when selecting a BSS. The
# priority groups will be iterated in decreasing priority (i.e., the larger
the
# priority value, the sooner the network is matched against the scan
results).
# Within each priority group, networks will be selected based on security
# policy, signal strength, etc.
# Please note that AP scanning with scan_ssid=1 is not using this priority
to
# select the order for scanning. Instead, it uses the order the networks are
in
# the configuration file.
#
# mode: IEEE 802.11 operation mode
# 0 = infrastructure (Managed) mode, i.e., associate with an AP (default)
# 1 = IBSS (ad-hoc, peer-to-peer)
# Note: IBSS can only be used with key_mgmt NONE (plaintext and static WEP)
# and key_mgmt=WPA-NONE (fixed group key TKIP/CCMP). In addition, ap_scan
has
# to be set to 2 for IBSS. WPA-None requires following network block
options:
# proto=WPA, key_mgmt=WPA-NONE, pairwise=NONE, group=TKIP (or CCMP, but not
# both), and psk must also be set.
#
# proto: list of accepted protocols
# WPA = WPA/IEEE 802.11i/D3.0
# RSN = WPA2/IEEE 802.11i (also WPA2 can be used as an alias for RSN)
# If not set, this defaults to: WPA RSN
#
# key_mgmt: list of accepted authenticated key management protocols
# WPA-PSK = WPA pre-shared key (this requires 'psk' field)
# WPA-EAP = WPA using EAP authentication (this can use an external
#  program, e.g., Xsupplicant, for IEEE 802.1X EAP Authentication
# IEEE8021X = IEEE 802.1X using EAP authentication and (optionally)
dynamically
#  generated WEP keys
# NONE = WPA is not used; plaintext or static WEP could be used
# If not set, this defaults to: WPA-PSK WPA-EAP
#
# auth_alg: list of allowed IEEE 802.11 authentication algorithms
# OPEN = Open System authentication (required for WPA/WPA2)
# SHARED = Shared Key authentication (requires static WEP keys)
# LEAP = LEAP/Network EAP (only used with LEAP)
# If not set, automatic selection is used (Open System with LEAP enabled if
# LEAP is allowed as one of the EAP methods).
#
# pairwise: list of accepted pairwise (unicast) ciphers for WPA
# CCMP = AES in Counter mode with CBC-MAC [RFC 3610, IEEE 802.11i/D7.0]
# TKIP = Temporal Key Integrity Protocol [IEEE 802.11i/D7.0]
# NONE = Use only Group Keys (deprecated, should not be included if APs
support
#  pairwise keys)
# If not set, this defaults to: CCMP TKIP
#
# group: list of accepted group (broadcast/multicast) ciphers for WPA
# CCMP = AES in Counter mode with CBC-MAC [RFC 3610, IEEE 802.11i/D7.0]
# TKIP = Temporal Key Integrity Protocol [IEEE 802.11i/D7.0]
# WEP104 = WEP (Wired Equivalent Privacy) with 104-bit key
# WEP40 = WEP (Wired Equivalent Privacy) with 40-bit key [IEEE 802.11]
# If not set, this defaults to: CCMP TKIP WEP104 WEP40
#
# psk: WPA preshared key; 256-bit pre-shared key
# The key used in WPA-PSK mode can be entered either as 64 hex-digits, i.e.,
# 32 bytes or as an ASCII passphrase (in which case, the real PSK will be
# generated using the passphrase and SSID). ASCII passphrase must be between
# 8 and 63 characters (inclusive).
# This field is not needed, if WPA-EAP is used.
# Note: Separate tool, wpa_passphrase, can be used to generate 256-bit keys
# from ASCII passphrase. This process uses lot of CPU and wpa_supplicant
# startup and reconfiguration time can be optimized by generating the PSK
only
# only when the passphrase or SSID has actually changed.
#
# eapol_flags: IEEE 802.1X/EAPOL options (bit field)
# Dynamic WEP key require for non-WPA mode
# bit0 (1): require dynamically generated unicast WEP key
# bit1 (2): require dynamically generated broadcast WEP key
#   (3 = require both keys; default)
#
# Following fields are only used with internal EAP implementation.
# eap: space-separated list of accepted EAP methods
#  MD5 = EAP-MD5 (unsecure and does not generate keying material ->
#      cannot be used with WPA; to be used as a Phase 2 method
#      with EAP-PEAP or EAP-TTLS)
#       MSCHAPV2 = EAP-MSCHAPv2 (cannot be used separately with WPA; to be
used
#    as a Phase 2 method with EAP-PEAP or EAP-TTLS)
#       OTP = EAP-OTP (cannot be used separately with WPA; to be used
#    as a Phase 2 method with EAP-PEAP or EAP-TTLS)
#       GTC = EAP-GTC (cannot be used separately with WPA; to be used
#    as a Phase 2 method with EAP-PEAP or EAP-TTLS)
#  TLS = EAP-TLS (client and server certificate)
#  PEAP = EAP-PEAP (with tunnelled EAP authentication)
#  TTLS = EAP-TTLS (with tunnelled EAP or PAP/CHAP/MSCHAP/MSCHAPV2
#       authentication)
#  If not set, all compiled in methods are allowed.
#
# identity: Identity string for EAP
# anonymous_identity: Anonymous identity string for EAP (to be used as the
#  unencrypted identity with EAP types that support different tunnelled
#  identity, e.g., EAP-TTLS)
# password: Password string for EAP
# ca_cert: File path to CA certificate file. This file can have one or more
#  trusted CA certificates. If ca_cert is not included, server certificate
#  will not be verified. This is insecure and the CA file should always be
#  configured.
# client_cert: File path to client certificate file (PEM/DER)
# private_key: File path to client private key file (PEM/DER/PFX)
#  When PKCS#12/PFX file (.p12/.pfx) is used, client_cert should be
#  commented out. Both the private key and certificate will be read from
#  the PKCS#12 file in this case.
# private_key_passwd: Password for private key file
# dh_file: File path to DH/DSA parameters file (in PEM format)
#  This is an optional configuration file for setting parameters for an
#  ephemeral DH key exchange. In most cases, the default RSA
#  authentication does not use this configuration. However, it is possible
#  setup RSA to use ephemeral DH key exchange. In addition, ciphers with
#  DSA keys always use ephemeral DH keys. This can be used to achieve
#  forward secrecy. If the file is in DSA parameters format, it will be
#  automatically converted into DH params.
# subject_match: Substring to be matched against the subject of the
#  authentication server certificate. If this string is set, the server
#  sertificate is only accepted if it contains this string in the subject.
#  The subject string is in following format:
#  /C=US/ST=CA/L=San Francisco/CN=Test AS/emailAddress=as@example.com
# phase1: Phase1 (outer authentication, i.e., TLS tunnel) parameters
#  (string with field-value pairs, e.g., "peapver=0" or
#  "peapver=1 peaplabel=1")
#  'peapver' can be used to force which PEAP version (0 or 1) is used.
#  'peaplabel=1' can be used to force new label, "client PEAP encryption",
#  to be used during key derivation when PEAPv1 or newer. Most existing
#  PEAPv1 implementation seem to be using the old label, "client EAP
#  encryption", and wpa_supplicant is now using that as the default value.
#  Some servers, e.g., Radiator, may require peaplabel=1 configuration to
#  interoperate with PEAPv1; see eap_testing.txt for more details.
#  'peap_outer_success=0' can be used to terminate PEAP authentication on
#  tunneled EAP-Success. This is required with some RADIUS servers that
#  implement draft-josefsson-pppext-eap-tls-eap-05.txt (e.g.,
#  Lucent NavisRadius v4.4.0 with PEAP in "IETF Draft 5" mode)
#  include_tls_length=1 can be used to force wpa_supplicant to include
#  TLS Message Length field in all TLS messages even if they are not
#  fragmented.
#  sim_min_num_chal=3 can be used to configure EAP-SIM to require three
#  challenges (by default, it accepts 2 or 3)
# phase2: Phase2 (inner authentication with TLS tunnel) parameters
#  (string with field-value pairs, e.g., "auth=MSCHAPV2" for EAP-PEAP or
#  "autheap=MSCHAPV2 autheap=MD5" for EAP-TTLS)
# Following certificate/private key fields are used in inner Phase2
# authentication when using EAP-TTLS or EAP-PEAP.
# ca_cert2: File path to CA certificate file. This file can have one or more
#  trusted CA certificates. If ca_cert2 is not included, server
#  certificate will not be verified. This is insecure and the CA file
#  should always be configured.
# client_cert2: File path to client certificate file
# private_key2: File path to client private key file
# private_key2_passwd: Password for private key file
# dh_file2: File path to DH/DSA parameters file (in PEM format)
# subject_match2: Substring to be matched against the subject of the
#  authentication server certificate.
#
# EAP-PSK variables:
# eappsk: 16-byte (128-bit, 32 hex digits) pre-shared key in hex format
# nai: user NAI
# server_nai: authentication server NAI
#
# EAP-FAST variables:
# pac_file: File path for the PAC entries. wpa_supplicant will need to be
able
#  to create this file and write updates to it when PAC is being
#  provisioned or refreshed.
# phase1: fast_provisioning=1 option enables in-line provisioning of EAP-
FAST
#  credentials (PAC)
#
# wpa_supplicant supports number of "EAP workarounds" to work around
# interoperability issues with incorrectly behaving authentication servers.
# These are enabled by default because some of the issues are present in
large
# number of authentication servers. Strict EAP conformance mode can be
# configured by disabling workarounds with eap_workaround=0.

# see /etc/wpa_supplicant.conf.example for more examples

network={
  ssid="mi_nombre_de_red"
  proto=WPA
  key_mgmt=WPA-PSK
  psk="mi_clave_de_63_dígitos"
  priority=99
}
```

3. Arrancar tal que así (suponiendo que la interfaz inalámbrica sea eth1):

```
wpa_supplicant -ieth1 -c/etc/wpa_supplicant/wpa_supplicant.conf&
```

4. Si el Punto de Acceso tiene servidor dhcp:

```
dhclient3 eth1
```

From:
http://wiki.legido.com/ - **Legido Wiki**

Permanent link:
http://wiki.legido.com/doku.php?id=informatica:linux:wifi:wpa

Last update: **2015/04/13 20:19**