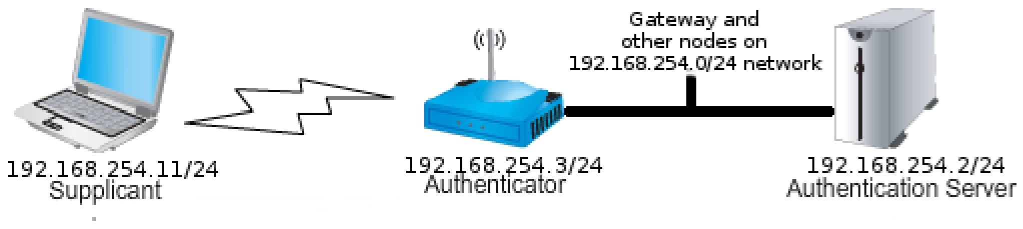
General information

Introduction

802.1X is a standard that is part of 802.1, it consists in access control to a network by allowing/blocking all packets at a given device's point of access into the network, including all broadcast packets, etc. It is possible on Wi-Fi, ethernet and possibly other mediums.

Upon connecting a device to an ethernet port on a switch or associating to a wireless SSID, the network switch or wifi access point (sometimes not necessarily very correctly referred to as wifi router) concerned will initially not allow any data to be transmitted/received by the device. It will request the connected device to identify itself and, upon approval of the supplied credentials by an authentication server, start accepting packets between the newly connected device and the rest of the network it serves, or continue dis-allowing, in the case the authentication server did not approve the credentials.

Architecture



802.1X defines how EAP messages are transmitted over an IEEE 802 network (e.g. ethernet, wifi).

RADIUS is a protocol to exchange information between an authenticator (also known as NAS (Network Access Server)) (see above picture) and an Authentication server. Each packet has a packet type, and contains multiple ValueName-Value attributes with relevant information as may be the case. By default, it operates on UDP port 1812. A well known RADIUS server software is called FreeRADIUS.

In the above example, a wifi access point is providing access to the 192.168.254.0/24 network. When a device (called a supplicant) attempts to connect to the wifi network, the wireless access point starts an EAP conversation with the supplicant over 802.1X, requesting it to supply credentials. The access point then connects to the authentication server (which, in the example above is also part of the 192.168.254.0/24 network) and sends the EAP response it received from the supplicant over the RADIUS protocol in a packet of type Access-Request. The authentication server may wish to request the supplicant for more information, it may thus answer the wifi access point with a RADIUS packet of type Access-Challenge containing an EAP message to be forwarded to the supplicant. The wifi access point facilitates this conversation between authentication server and supplicant until the authentication server sends a RADIUS packet to the wifi access point which is of either type Access-Accept or Access-Reject.

EAP itself is an encapsulation protocol, inside it a variety of different protocols can be run to perform authentication. An authentication protocol can be encapsulated directly inside the EAP tunnel or an encryption protocol can be, inside which then, (at least in the cases of EAP-PEAP and EAP-TTLS) eventually another instance of the EAP protocol is encapsulated, inside which, finally, the actual authentication protocol is encapsulated. According to the comment in the mods-available/eap config file at the beginning of the 'ttls' section, the hierarchy with EAP-TTLS is RADIUS EAP TLS Diameter (protocol comparable in scope to RADIUS) again EAP the actual protocol used for authentication). EAP-PEAP is a comparable encryption protocol that can be used instead of EAP-TTLS. An example of an authentication protocol that can be used for the actual authentication is MSCHAPv2.

In the case an encrypted tunnel is used, the conversation outside the encrypted tunnel is called the outer tunnel. The conversation within the encrypted tunnel is called the "Inner Tunnel". At the time of setting up the encrypted tunnel, the authentication server presents a certificate identifying itself which the supplicant may (and should) choose to verify before sending its login credentials to the server.

In the case of an Access-Accept, the wifi access point now allows the supplicant to join the network, or, in the case of Access-Reject, will not. Once the NAS has granted access, for 802.1X/RADIUS/authentication server, the job is done, and the supplicant becomes part of the 192.168.254.0/24 network's broadcast domain. The authentication server can specify attributes in the replying packet to give the NAS additional instructions, for example, it might request the NAS to place the newly connected supplicant in a specific VLAN, or it might specify for how long the supplicant is allowed to remain connected. The authentication server is able to log that the user connected along with information from attributes the NAS might have sent, generally this includes the MAC address of the supplicant, MAC address of the NAS, username (if authentication was done by username/password) and more depending on the NAS model. The supplicant can now initiate a DHCP request for an IP address or any other action as may be appropriate.

EAP and RADIUS have support for a great variety of features or different methods that can be encapsulated there-in to perform authentication (e.g. it is possible to use different protocols that authenticate the user using a username and password, or using client certificates, or even SIM cards), but it is still up to the supplicant, NAS and authentication server implementations to choose which ones they support or not (even different NASes have been seen to send MAC addresses in different format, e.g. TP-Link Archer C20 sends 01-23-45-67-89-AB, while TP-Link TD-W8968 sends 0123456789ab. TP-Link TL-WR740N, in the case of the NAS MAC address, sends 01-23-45-67-89-AB:SSID, where SSID is the configured wireless SSID).

FreeRADIUS configuration files

FreeRADIUS configuration files are many. They are located in /etc/freeradius/3.0 on Debian systems. Apparently, many other environments use a slightly different location. Most of them contain plenty of comments explaining what the configuration does and advice and warnings, but some understanding of the protocols, or getting used to, is often necessary to understand them. The FreeRADIUS technical guide (link in "Support / Knowledge places" section) describes them in more detail along with more useful information. Here is a brief look at some of the configuration files/folders:

clients.conf - List of clients that will be connecting to the FreeRADIUS server, including their IP addresses and passwords that they will use to authenticate to FreeRADIUS. What is called a FreeRADIUS client is a NAS, not a supplicant. Supplicants do not speak directly to RADIUS servers.

mods-available - Folder containing config files of modules that can be used with FreeRADIUS

mods-enabled - Folder containing symlinks to files in the mods-available folder, for modules that should be enabled

mods-config - Folder containing more config related to modules and things like, e.g. .sql files containing empty schemas for initial creation of databases for use with the sql module

certs - Folder containing certificates usable by FreeRADIUS and respective configuration files/makefile/etc. needed to generate them

sites-available - "Sites" that can be served by FreeRADIUS.

sites-enabled - Folder containing symlinks to files in the sites-available folder, for sites that should be enabled

mods-available/eap - Configuration file for EAP module

users - In the default configuration, a text file based user database

By default, the 'default' and 'inner-tunnel' sites are enabled. 'default' is the outer tunnel, it listens for incoming requests from the NASes, 'inner-tunnel' receives requests forwarded by the outer-tunnel site containing the data from the inner tunnel.

The site files contain multiple sections, here are some of them:

authorize - This section lists modules/code that are run when a request is received, in preparation for authentication. One of the important tasks is to find out which authentication method/protocol the supplicant is trying to use and which FreeRADIUS module is appropriate to deal with it. When one of the listed modules finds that it is able to deal with the request, it informs FreeRADIUS. Another important task is to load in memory information that might be needed for authentication. For example, the sql module loads the relevant credentials from the SQL database so that the appropriate authentication module that later runs in the authenticate section can access them in order to compare them with what the supplicant sent.

authenticate - After the authorize section ran, this section takes care of the actual authentication. In here modules are called that assess the information made available to them and answer in regards of what action should be taken.

post-auth - After it has been determined what action should be taken, this section is run in case authentication succeeded, otherwise the Post-Auth-Type Reject subsection in case authentication failed. This sections take care of any extra tasks required to be carried out, for example logging, and can also add /modify attributes to be sent back to the NAS as part of the reply packet.

Misc. Information

1. The RADIUS protocol also includes the possibility for the NAS to send accounting information to the authentication server for the purpose of logging user activity, this is on a separate port, UDP 1813, this has not been investigated as it seems to be not supported by the NASes we have, and presumably is not on any NAS in the price range.
2. The RADIUS protocol also includes the possibility for the authentication server to initiate a connection to the NAS and inform it to disconnect a currently logged in user, but again, this has not been investigated much as it seems to be not supported by the NASes we have, and presumably is not on any NAS in the price range.
3. In absence of the above mentioned feature, once a user has connected, there is no way to disconnect them other than rebooting the NAS. If the system administrator chooses to disable a user account on the authentication server, the user will be denied access only from the next time they try to connect, if they are already connected at the time of disabling the account, they will remain connected for the moment. It is possible for the authentication server to specify with the "Session-Timeout" and "Termination-Action" attributes of the Access-Accept packet for how long a user is allowed to remain connected without repeating authentication (again subject to NAS supporting). Depending on the NAS and supplicant models it has been observed that there can be a new authentication request, in the case of wifi, when the supplicant moves out of range of the wifi and then comes back, or when the user switches wifi off and on again on the supplicant, but might also not be the case.
4. A "User-Name" attribute is usually provided in both the outer and inner tunnels, often referred to as outer identity and inner identity, respectively. It is often suggested to send the real username only in the inner tunnel for privacy reasons as it is encrypted, and send the string 'anonymous' as User-Name in the outer tunnel. It was found, however, to be unreasonably difficult, if at all possible, to configure Mac OS, Windows and iPhone clients to do so. The 'User-Name' attribute in the outer tunnel is still useful as a realm for the user may be specified and a RADIUS server might need to forward the request to another RADIUS server, based on the specified realm and might be unable to read the encrypted inner-tunnel. If the outer user name is anonymized and a realm needs to be specified, the attribute would read 'anonymous@realm'.
5. It is possible for a FreeRADIUS server to forward a request to an upstream RADIUS server to handle. The answer then gets sent back and the local FreeRADIUS server forwards it to the client. It is possible (or maybe most of the time/always the case?) that intermediary RADIUS servers are not able to decrypt the data within the encrypted inner-tunnel, and do not have access to it. One way for a RADIUS server to determine whether it should answer a given query itself or forward it upstream is to look at the realm part of the outer identity provided in the outer tunnel.
6. According to Wikipedia, there is a similar protocol evolved from RADIUS, called Diameter: https://en.wikipedia.org/wiki/Diameter_(protocol)
7. In Debian, the FreeRADIUS daemon is called freeradius. Apparently, in many other environments it is called radiusd.
8. When debugging, it is recommended to stop the FreeRADIUS daemon and run freeradius -X in a terminal. A lot of debug information is then printed on the screen, which can be used to see what's happening. This is stressed a lot on the FreeRADIUS wiki and users' mailing list.
9. A system called RSN pre-authentication (sometimes called WPA2 pre-authentication) can be used, when multiple NASes are installed in a network, which allows the NASes to talk amongst themselves and inform other NASes about supplicants that have authenticated through them. This allows a wireless supplicant that moves within the building to connect to a different NAS without having to repeat RADIUS authentication. This feature has not been investigated much. It might be that it is only available on wireless LAN NASes. It is suspected that this feature is the cause of the Session-Timeout / Termination-Action attributes not working in some corner cases (observed sometimes when the supplicant roamed between different NASes in a particular order), resulting in the supplicant remaining connected to the network longer than the specified limit, without repeating authentication to the RADIUS server.
10. Through a rather limited amount of research in this regards, it is my understanding that for each supplicant a different key is generated and used to encrypt wireless communication between it and the NAS once authenticated, and is generated in part using also a key provided by the RADIUS server, and supplicant and NAS MAC addresses. A supplicant, during authentication specifies its MAC address as well as login

credentials, which is logged in the RADIUS server along with user details. As per tests, it is inferred that, after a supplicant connects to the wifi network, the NAS does not allow the supplicant to send packets with a source MAC address other than the one specified when connecting. This was tested on a TP-Link TL WR740N, with 802.1X wireless security as well as with wireless security disabled, and on a TP-Link Archer C20 with wireless security disabled using 'ostinato' (package available in Debian repositories) to transmit test packets. This protects against MAC spoofing once a user is logged into the network.

11. There are two different WPA standards, WPA and WPA2. The latter is the proper one, and the former was released as an interim solution to be applied on existing devices with inadequate hardware for running WPA2 due to the urgency of upgrading from WEP which was at the time discovered to be significantly insecure. Similarly there is an "encryption" setting in most NASes with two possible settings, TKIP and AES, where AES is the proper solution, and TKIP the interim one.
12. Please see the "Supplicant configuration" and "WiFi access point (NAS) configuration" sections of the "802.1X secured wifi installation" page's "Installation" child page for more information about features supported by and behavior specific to some NAS models and supplicants.
13. EAP-TLS is an authentication protocol that can be used to authenticate supplicants based on client certificates.
14. Depending on the chosen authentication method, different encryption algorithms may or may not be supported for storage of the passwords in the user database. A table showing some of the authentication methods with the supported algorithms is here: http://deployingradius.com/documents/protocols/compatibility.html

Support / Knowledge places

A lot of documentation regarding FreeRADIUS and many modules, and some examples, can be found on the FreeRADIUS wiki:

https://wiki.freeradius.org/

FreeRADIUS technical guide:

https://networkradius.com/doc/FreeRADIUS%20Technical%20Guide.pdf

A lot more useful information can be found on the FreeRADIUS mailing list:

https://wiki.freeradius.org/guide/Users-Mailing-List

http://lists.freeradius.org/mailman/listinfo/freeradius-users

It is stressed in the FreeRADIUS wiki to only follow documentation from the official wiki, as third party documentation is often outdated and/or incorrect.

Another good site (linked to in FreeRADIUS wiki):

http://deployingradius.com

Page with information on finding documentation / support:

https://freeradius.org/documentation/

Sources

https://en.wikipedia.org/wiki/IEEE_802.1X

https://en.wikipedia.org/wiki/Extensible_Authentication_Protocol

https://en.wikipedia.org/wiki/Extensible_Authentication_Protocol#PEAP

https://en.wikipedia.org/wiki/Extensible_Authentication_Protocol#EAP_Tunneled_Transport_Layer_Security_(EAP-TTLS)

https://en.wikipedia.org/wiki/Diameter_(protocol)

https://wiki.freeradius.org/

https://wiki.freeradius.org/guide/Basic-configuration-HOWTO

https://wiki.freeradius.org/guide/SQL-HOWTO-for-freeradius-3.x-on-Debian-Ubuntu

https://wiki.freeradius.org/modules/Rlm_python

https://wiki.freeradius.org/config/Certificates

http://deployingradius.com/

http://deployingradius.com/documents/configuration/certificates.html

http://deployingradius.com/documents/protocols/compatibility.html

https://www.cwnp.com/uploads/802-11i_key_management.pdf