802.1X Overview

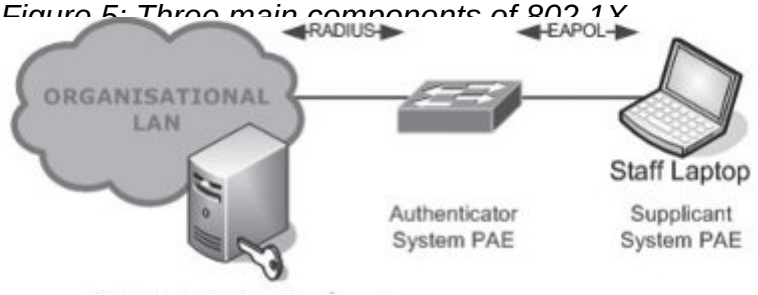
How 802.1X works

There are three main components in the 802.1X authentication cycle:

- Supplicant Port Access Entity (PAE). This is software on the client device which handles the client side of the authentication conversation.
- Authenticator PAE. This is the device which controls the port access to the network. Supplicant PAE communications are tunnelled through the Authenticator PAE.
- Network Authentication Server (NAS). This is the server which authenticates the client through EAP authentication conversations with the Supplicant PAE.

The deployment of 802.1X is over-simplified in the diagrams. However, following the implementation of redundant network authentication servers within the LAN, authenticator systems PAE will form part of the access layer switches.

Figure 5: Three main components of 802.1X



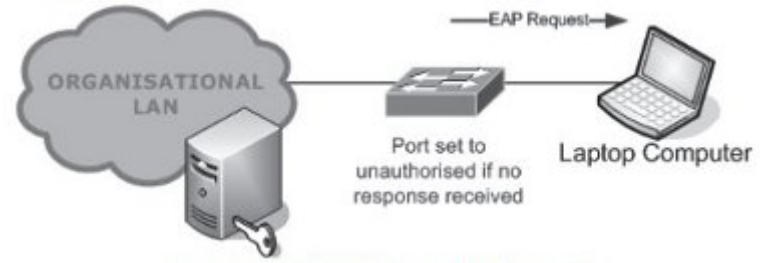
[1]

When a device is connected to a 802.1X enabled switch port, the link physically becomes active and both the 802.1X supplicant on the client and the switch detect the presence of a device at the either end of the link.

Modern switches are fairly complex and there are a number of stages before full connectivity is established. This is important for the debugging of the connection stage as many problems can cause 802.1X to fail including: a badly configured supplicant, faulty cable, network card or switch configuration file.

1. Initially when a client connects to the network, the Authenticator PAE (the switch or AP) initiates an authentication conversation by sending an EAP-Request. If the Authenticator PAE does not receive a response it will time-out and set the port into an unauthorised state.

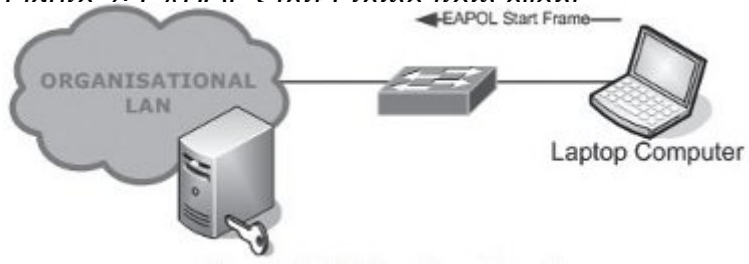
Figure 6: Initial client connection to the network



[2]

2. Once connected, if the Supplicant PAE does not receive an authentication initiation frame, or misses it due to timing, the Supplicant PAE will send an EAPOL-Start frame to the authenticator. Either the Authenticator PAE or the Supplicant PAE can start an 802.1X authentication conversation.

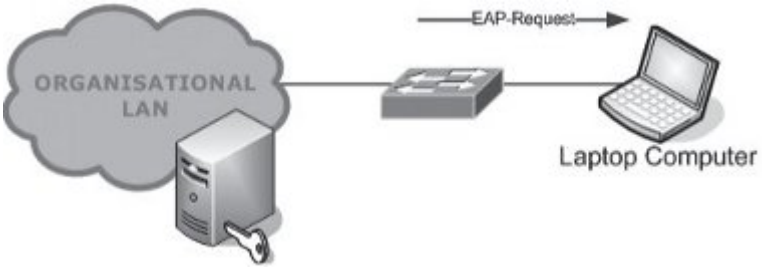
Figure 7: EAPOL-Start Frame from client



[3]

3. Upon receiving the EAPOL-Start the Authenticator PAE returns an EAP-Request to the Supplicant PAE.

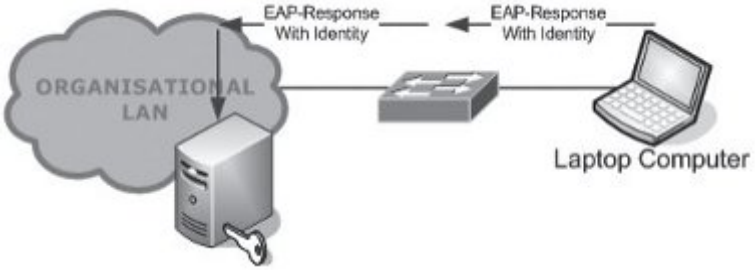
Figure 8: EAP-Request sent to Supplicant PAE



[4]

4. After receiving the EAP-Request the Supplicant PAE sends an EAP-Response containing the client's identity. This is relayed by the Authenticator PAE to the Network Authentication Server.

Figure 9: EAP-Response Relayed to the Network Access Control via the Authenticator



[5]

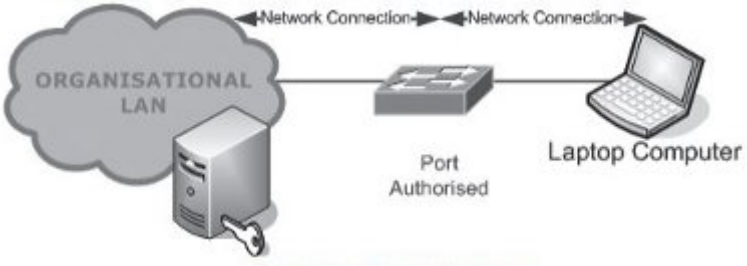
5. The Network Authentication Server, via the Authenticator PAE, performs an EAP authentication conversation with the Supplicant PAE, the result of which is either an EAP-Success or an EAP-Fail.

Figure 10: EAP Authentication Conversation

[6]

6. Upon receiving an EAP-Success the Authenticator PAE authorises the port and the client joins the network.

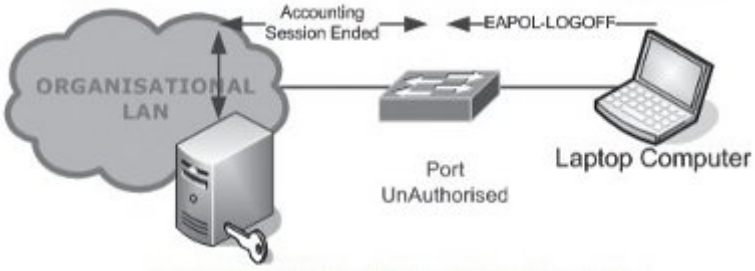
Figure 11: Port Authorisation



[7]

7. Finally, to log off, the Supplicant PAE sends an EAPOL-LOGOFF. The Authenticator PAE unauthorises the port and ends the accounting session for the Supplicant PAE.

Figure 12: EAPOL-LogOFF and Client Disconnection



[8]

Within 802.1X authentication, EAP is used as a mechanism for communicating authentication information between the supplicant and the authentication server.

Within the EAP-Request the Authenticator PAE can request a particular EAP method (such as PEAP). If the Supplicant PAE supports the requested EAP method then an authentication can take place. If the Supplicant PAE does not support the EAP method then the Authenticator PAE will request a different EAP Method.

EAP Methods

EAP is a framework which has numerous implemented mechanisms for authentication. These mechanisms are more commonly referred to as EAP types or EAP methods. There are many EAP types; however this Technical Guide will focus on the most commonly used EAP methods: EAP-PEAPv0 (MSCHAPv2) and EAP-TTLS (MSCHAPv2). EAP methods will often use an inner authentication protocol, with the EAP method being the wrapper for the inner authentication.

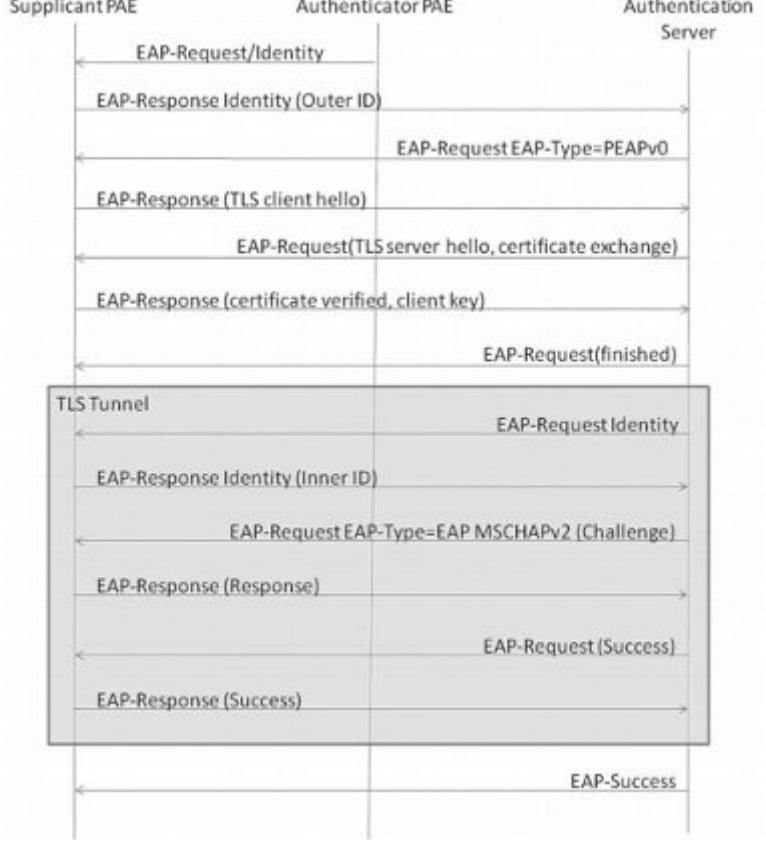
EAP-PEAPv0

Protected Extensible Authentication Protocol (PEAP) is an EAP method, based upon EAPTLS, that was originally developed by Microsoft®, Cisco® and RSA Security (Kamath, Palekar, & Wodrich, 2002 [9]). PEAP is commonly deployed in Microsoft® Windows® environments as it is the EAP method favoured by Microsoft®. Both the Windows XP® 802.1X supplicant and the Windows Vista® supplicant support PEAP natively. PEAP is most commonly used with the Microsoft® Challenge Handshake Authentication Protocol version 2 (MSCHAPv2) as an inner authentication method.

PEAP-MSCHAPv2 authenticates the server using a PKI (Public Key Infrastructure) certificate and the client using password based credentials. User credentials are protected by the formation of a TLS (Transport Layer Security) tunnel between the Supplicant PAE and the authentication server. The creation of the TLS tunnel is achieved using a server side certificate which authenticates the server's identity. Inside of the TLS Tunnel MSCHAPv2 can be used for either user authentication, using the client's username and password, or for machine authentication using the machine name and active directory credentials.

Figure 13 shows a typical PEAP-MSCHAPv2 authentication. PEAP authentication has two phases: an initial phase to set up the TLS tunnel, and then a second phase to perform the authentication of the user or machine using MSCHAPv2 within the TLS tunnel. The PEAP authentication ends with either an EAP-Success or an EAP-Failure sent in clear text.

Figure 13: Example PEAP-MSCHAPv2 Conversation



[10]

EAP-TTLS

EAP-Tunnelled Transport Layer Security (EAP-TTLS) is an EAP method developed by Funk Software and Certicom (Funk & Blake-Wilson, 2002 [11]). Like PEAP, EAP-TTLS is an EAP method which builds upon EAP-TLS. EAP-TTLS establishes a TLS tunnel through which the authentication can take place. In the establishment of the TLS tunnel the client / server authentication can either be mutual, like EAP-TLS, or just authentication of the server. As with other EAP-TLS based methods the authentication is through PKI certificates. After establishment of a TLS tunnel EAP-TTLS supports a number of inner authentication protocols, including PAP and MSCHAPv2.

RADIUS

A key part of the 802.1X mechanism is the authentication server. The authentication server, as the title suggests, handles the authentication requests from the client.

Probably the most commonly used authentication server (for 802.1X) is the RADIUS server. RADIUS servers are AAA servers which allow not only the authentication of 802.1X clients but also handle authorisation and the accounting of the users' session.

Microsoft® Windows Server® 2003 includes an industry-recognised RADIUS server called IAS (Internet Authentication Service). It is a technically capable implementation of RADIUS and is well regarded within the industry.

An alternative is the Cisco® ACS implementation which provides integration with the Cisco® portfolio of technology. Some protocols like LEAP and EAP-FAST are easier to implement with the Cisco® ACS; however many organisations use Open Source RADIUS

implementations like FreeRADIUS.

Links
[1] http://community.ja.net/system/files/images/tg-ieee8021x-05.jpg
[2] http://community.ja.net/system/files/images/tg-ieee8021x-06.jpg
[3] http://community.ja.net/system/files/images/tg-ieee8021x-07.jpg
[4] http://community.ja.net/system/files/images/tg-ieee8021x-08.jpg
[5] http://community.ja.net/system/files/images/tg-ieee8021x-09.jpg
[6] http://community.ja.net/system/files/images/tg-ieee8021x-10.jpg
[7] http://community.ja.net/system/files/images/tg-ieee8021x-11.jpg
[8] http://community.ja.net/system/files/images/tg-ieee8021x-12.jpg
[9] http://www.tools.ietf.org/html/draft-kamath-pppext-peapv0-00
[10] http://community.ja.net/system/files/images/tg-ieee8021x-13.jpg
[11] http://tools.ietf.org/html/draft-funk-eap-ttls-v1-01