



802.1X and MAB Authentication

Teldat Dm783-I

Copyright© Version 11.04 Teldat SA

Legal Notice

Warranty

This publication is subject to change.

Teldat offers no warranty whatsoever for information contained in this manual.

Teldat is not liable for any direct, indirect, collateral, consequential or any other damage connected to the delivery, supply or use of this manual.

Table of Contents

I	Related Documents.	1
Chapter 1	Introduction	2
1.1	802.1X Authentication Introduction	2
1.2	MAB Authentication Introduction	2
1.3	Controlled and uncontrolled ports	2
1.4	802.1X Authentication Elements	3
1.5	EAPOL Frames	4
1.5.1	Ethernet type and destination address.	4
1.5.2	Types of frames	4
1.5.3	Frame format	4
1.5.4	Authentication Process	5
1.6	MAB authentication elements	6
1.7	RADIUS packet for MAB authentication	6
1.7.1	Processing authentication	7
Chapter 2	Configuring the 802.1X and MAB Authentication	8
2.1	Using 802.1X authentication in Teldat devices	8
2.2	Using MAB authentication in Teldat devices	8
2.3	802.1X Authentication Configuration Menus	8
2.3.1	Accessing the global configuration menu.	8
2.3.2	Accessing the configuration menu for an Ethernet interface	8
2.3.3	Accessing the configuration menu for a one switch port	9
2.4	Global Configuration Commands	9
2.4.1	? (HELP)	9
2.4.2	NO	9
2.4.3	SYSTEM-AUTH-CONTROL	9
2.4.4	EXIT	10
2.5	Per Port Configuration Commands	10
2.5.1	? (HELP)	10
2.5.2	AUTHENTICATOR	10
2.5.3	NO	13
2.5.4	PAE	13
2.5.5	SUPPLICANT	14
2.5.6	EXIT	17
Chapter 3	802.1X Authentication Monitoring	18
3.1	802.1X Authentication Monitoring Menus	18
3.1.1	Accessing the global monitoring menu.	18
3.1.2	Accessing the monitoring menu for an Ethernet interface.	18
3.1.3	Accessing the monitoring menu for a one switch port	18

3.2	Global Monitoring Commands	18
3.2.1	? (HELP)	19
3.2.2	SHOW	19
3.2.3	SUPPLICANT	19
3.2.4	EXIT	19
3.3	Per Port Monitoring Commands	19
3.3.1	? (HELP)	20
3.3.2	MAB REAUTHENTICATE	20
3.3.3	REAUTHENTICATE	20
3.3.4	SHOW AUTHENTICATOR.	20
3.3.5	SHOW SUPPLICANT	23
3.3.6	EXIT	25
Chapter 4	Configuration Examples	26
4.1	802.1X Authentication in a port	26
4.1.1	Scenario.	26
4.1.2	Globally configuring 802.1X authentication	26
4.1.3	Configuring 802.1X authentication in Ethernet ports	26
4.1.4	Configuring Radius	27
4.1.5	Complete configuration	27
4.2	802.1X Supplicant in an Ethernet interface	28
4.2.1	Scenario.	28
4.2.2	802.1X supplicant: Global configuration	29
4.2.3	Configuring the 802.1X supplication in the Ethernet port	29
4.3	MAB Authentication in a port	29
4.3.1	Scenario.	30
4.3.2	Configuring the MAB authentication	31
Chapter 5	Annex	32
5.1	Third Party Software	32

I Related Documents

Teldat Dm733-I Radius Protocol

Teldat Dm739-I IPSec

Teldat Dm771-I Wireless LAN Interface

Teldat Dm800-I AAA Feature

Chapter 1 Introduction

1.1 802.1X Authentication Introduction

Sometimes, access to certain areas on a LAN must be restricted so that only authorized users can reach them. 802.1X authentication (described in the IEEE standard) allows you to control LAN access by defining an authentication and authorization mechanism associated with a connection point to said LAN. This connection point to the LAN is known as a port.

802.1X authentication is used, for example, to allow or deny access to certain segments in a LAN connected through a bridge. If authentication for a device connected to a bridge port fails, then the device is denied access.



Note

802.1X authentication makes sense for point-to-point connections where a single device is connected to a port with authentication. For point-to-multipoint connections, you need to establish a point-to-point logical association to ensure authentication operates correctly. In WLAN interfaces, the 802.11i standard describes how to establish said association in order to use 802.1X authentication.

1.2 MAB Authentication Introduction

Not all devices are compatible with 802.1X authentication (network printers for example). For these devices to use a protected network environment, alternative authentication mechanisms are necessary.

One option is to deactivate the 802.1X authentication in the port. However, this leaves the port unprotected and open. Another, slightly more reliable, option is to use MAC Authentication Bypass (MAB). When MAB is configured in a port, it triggers an authentication process with a RADIUS server using the connected device's MAC as the user name and password.

The network administrator must take into account the configuration of the RADIUS server when it comes to MAC address authentication. These addresses can be included as regular users or added to the network's database for verification.



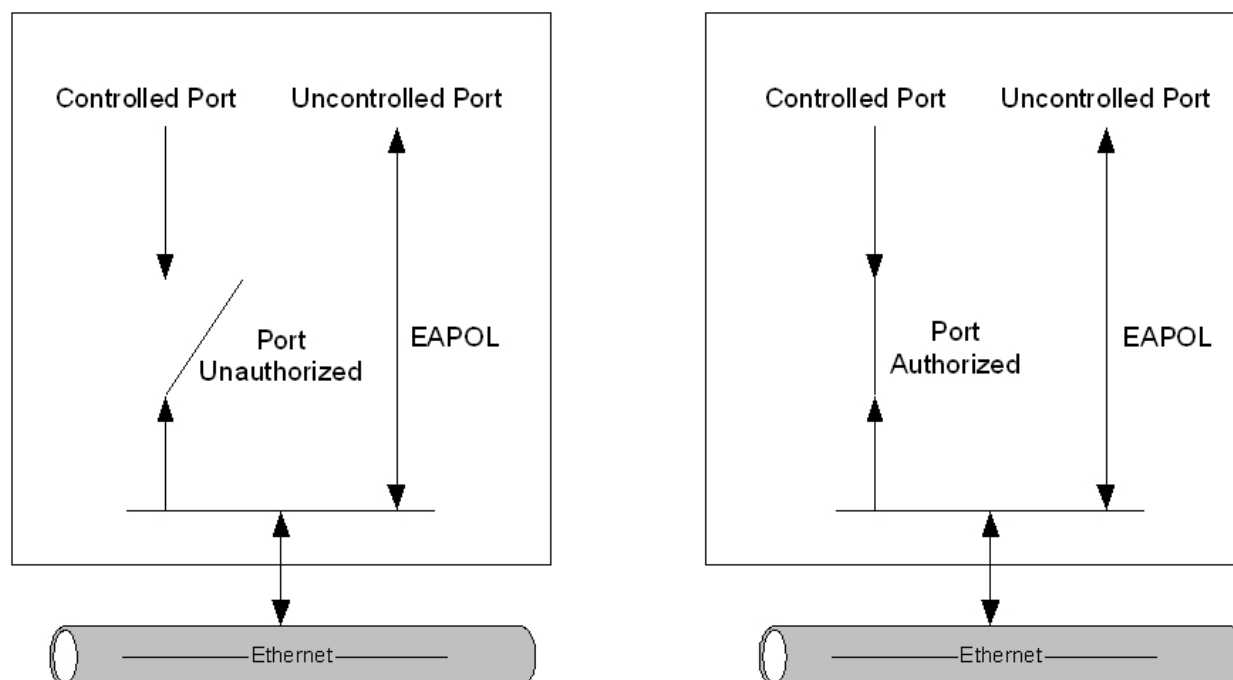
Note

MAB authentication is not available in all switches.

1.3 Controlled and uncontrolled ports

In 802.1X authentication, the physical port connected to a LAN segment is divided into two logical ports: the controlled port and the uncontrolled port. When a port receives a packet, it appears in both the controlled and uncontrolled ports. The controlled port drops all packets when 802.1X authentication has not been carried out. The uncontrolled port allows the access of all packets used for authentication.

Consequently, while the authentication process is being carried out, only packets used by said authentication have access. The controlled port is considered open, meaning data flow is not permitted. Only when the authentication process has successfully completed does the controlled port close, allowing normal data flow.



Controlled and uncontrolled port

1.4 802.1X Authentication Elements

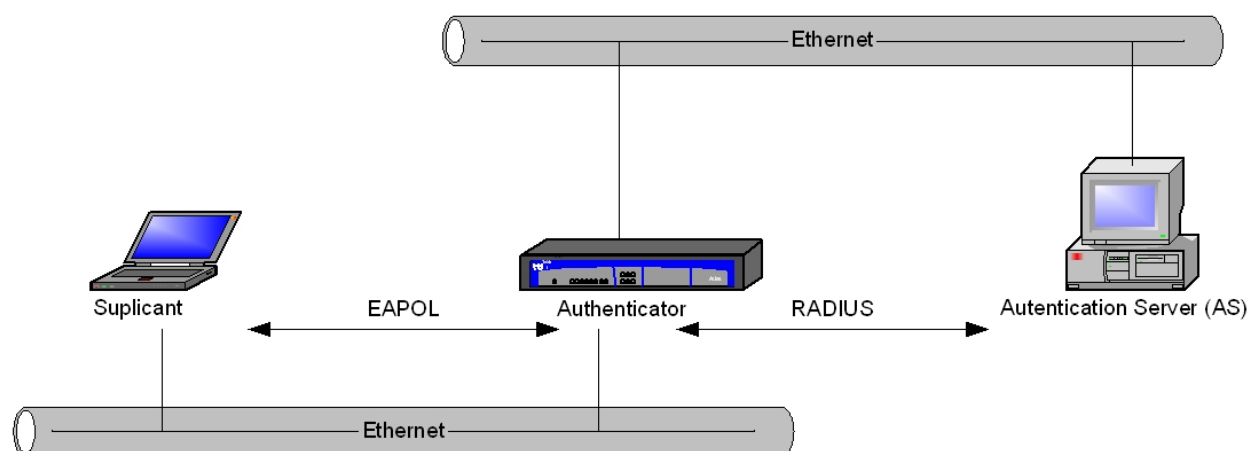
The authentication process is based on the Extensible Authentication Protocol (EAP), described in RFC 3748 together with the use of EAP in LAN networks (wireless or cabled). Said protocol describes EAP encapsulation so that packets can be sent by the LAN. It also creates new types of packets to simplify the use of EAP in LAN networks. The encapsulation defined in the 802.1X standard is known as EAPOL (EAP Over LAN).

In EAP authentication and, by extension, in 802.1X authentication, three components can be seen:

- Supplicant: element that wants to access the network.
- Authenticator: element controlling network access. This allows communication between the supplicant and the authentication server.
- Authentication server: this is the element that authenticates the supplicant, deciding if the latter can access the network or not. The authentication server can be the authenticator itself, but it's usually an external element.

The role of each element in the network access process can be easily understood by means of a simple example. Imagine a stranger arrives at the company and wants to enter the building. This is the supplicant. The building's doorman controls the access, allowing people into the building or not. However, he doesn't have the authority to decide who can come in or not. This is the authenticator. When the stranger requests entrance, the doorman calls the person in charge and asks if the visitor can come in. The boss probably asks for data on the visitor. The doorman repeats the boss's questions and relays the answers to the latter. Finally, depending on the responses given, the boss decides whether the visitor can enter or not. The boss is the authentication server.

The following figure shows the elements that make up 802.1X authentication.



802.1X authentication



Note

Although the figure shows RADIUS as the communication protocol between the authenticator and the authentication server, you can use any authentication protocol that transports EAP.

Authentication is carried out between the supplicant and the authentication server. When the supplicant wants to access the network, it triggers the process. The authenticator, through the uncontrolled port, passes the packets exchanged between the supplicant and the authentication server during authentication. Thus, the authenticator acts as the relay for the packets exchanged between the supplicant and the authentication server. The authenticator only looks at the end result: if authentication is successful, the controlled port allows the supplicant to access the network.

Communication between the supplicant and authenticator is carried out using the EAPOL protocol.

For authentication to be carried out safely, the communication channel between the authenticator and the authentication server must be completely secure. There are several options to transfer EAP packets between the authenticator and the authentication server. The Radius protocol is one of the most commonly used to send the EAP packets.

EAP is really a large mixture of authentication protocols. During the authentication phase, the specific EAP protocol to be used is negotiated. Several authentication methods use EAS bases. Some of these are:

- PEAP: Protected EAP.
- EAP-TLS: EAP-Transport Layer Secure.
- EAP-TTLS: EAP-Tunneled Transport Layer Secure.

1.5 EAPOL Frames

1.5.1 Ethernet type and destination address

The EAPOL frames are Ethernet frames that use 0x888E. As a rule, EAPOL frames go to a special destination address (known as PAE group address) that takes value 01-80-C2-00-00-03. Only in point-to-multipoint environments is the device MAC address used as the packet destination address.

1.5.2 Types of frames

There are five types of frames:

- (1) EAP-Packet: EAP frames encapsulated in Ethernet format.
- (2) EAPOL-Start: frame used by the supplicant to start the authentication process.
- (3) EAPOL-Logoff: frame used by the supplicant to tell the authenticator to abandon the session.
- (4) EAPOL-Key: frames used to send information on the keys.
- (5) EAPOL-Encapsulated-ASF-Alert: ASF alert frames (Alerting Standards Forum). These allow ASF frames to pass through an unauthorized port.

1.5.3 Frame format

0	2	3	5	7	N
PAE Ethernet type	Protocol version	Packet Type	Packet Body Length	Packet Body	

PAE Ethernet type: 0x888E.

Protocol version: The current version is 2 (defined in the 802.1X-2004 standard).

Packet Type: EAP-Packet (0), EAPOL-Start (1), EAPOL-Logoff (2), EAPOL-Key (3) or EAPOL-Encapsulated-ASF-Alert (4).

Packet Body Length: data length in bytes. A value equal to zero indicates there is no further data.

Packet Body: data. Depending on the type of packet, the data is interpreted in a different way. E.g. an EAP-Packet contains an EAP packet.

1.5.4 Authentication Process

The authentication process is as follows: when the authenticator detects an element connected to a port configured with 802.1X authentication, authentication starts. To do this, an EAP identity request frame is sent.

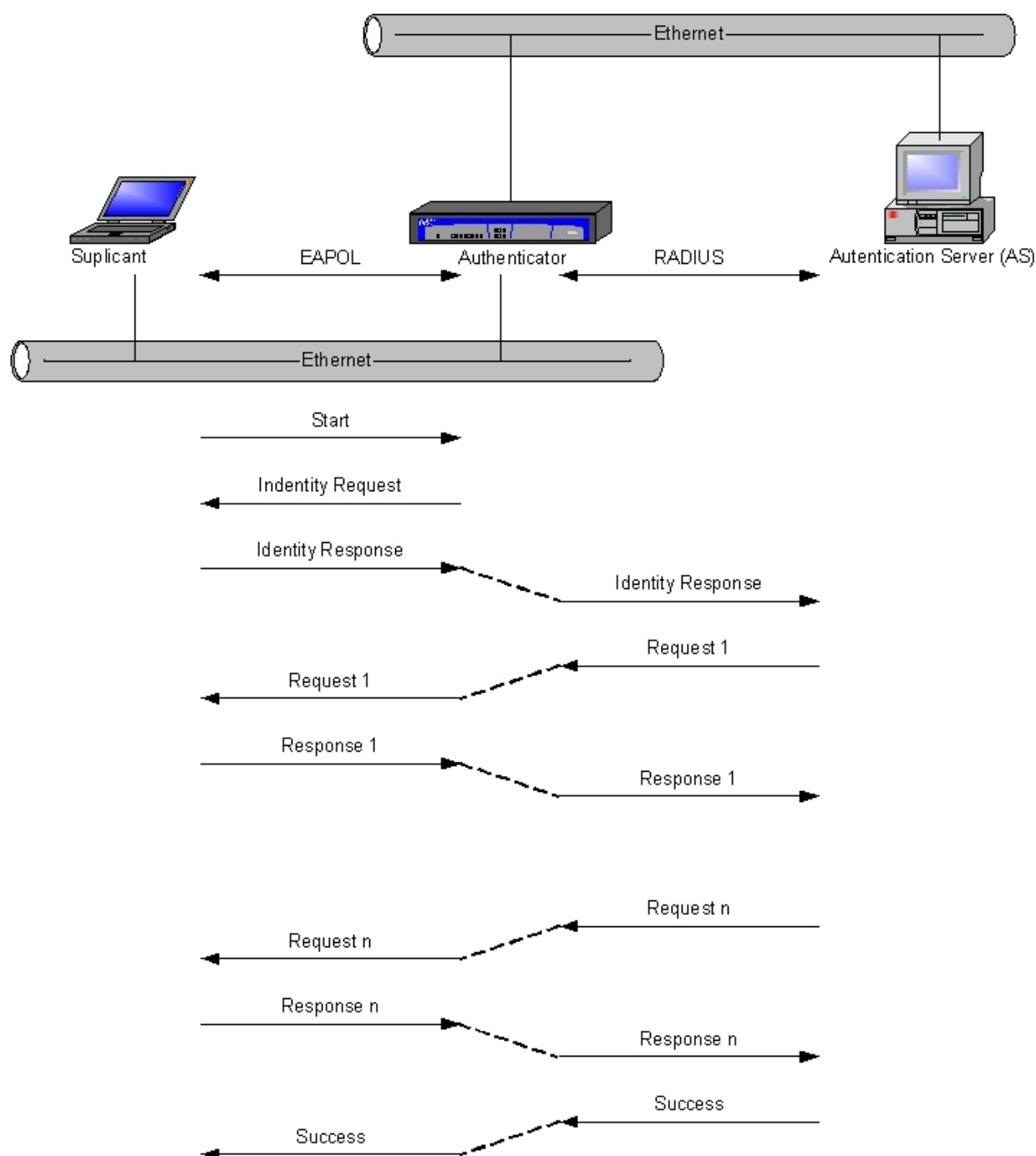


Note

The authentication process can also be started by the supplicant through an EAPOL-Start frame. The authenticator then sends an identity request frame as a response to said frame.

At this point an EAP dialog between the supplicant and the authentication server is established, with the authenticator serving as a simple frame translator. The EAP dialog is basically a process where the authentication server sends EAP Request packets and the supplicant replies with EAP Response packets. When the EAP authentication terminates, the authentication server either sends an EAP Success packet or an EAP Fail packet. The packet is read by the authenticator, who then grants or refuses access to the controlled port.

The following figure shows the authentication process.



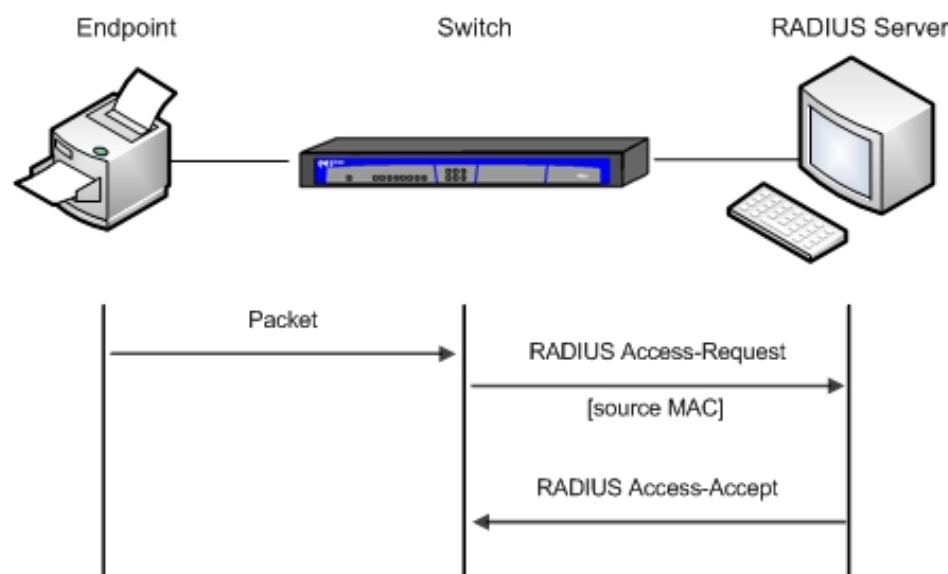
802.1X authentication process

1.6 MAB authentication elements

The basis of the authentication process is the Remote Authentication Dial In User Service (RADIUS) protocol, described in the RFC 2865 standard. The RADIUS protocol is an authentication and authorization protocol on network access. For further information, please see manual *Teldat-Dm 733-I "Radius Protocol"*.

There are three components in MAB authentication:

- Terminal: device that wants to access the network.
- Switch: element that controls network access.
- RADIUS Server: element that authenticates the terminal, granting access to the network or not.



MAB authentication

The switch begins the MAB authentication process by opening the port to accept a single packet through which the terminal's MAC address is learned. Once the switch has learned the MAC address, it begins to exchange messages with the RADIUS server.

1.7 RADIUS packet for MAB authentication

The RADIUS Access-Request packet the switch generates for the MAB authentication must include the terminal's MAC address in three attributes:

- Attribute 1 (Username)
- Attribute 2 (Password)
- Attribute 31 (Calling-Station-Id)

Although the MAC address is the same in each attribute, the address format is different. This is important, as the different RADIUS servers can use different attributes to validate the MAC address. Some RADIUS servers can only focus on Attribute 31 (Calling-Station-Id), while others verify the user and the password in Attributes 1 and 2.

The following table shows the MAC address format for each attribute.

RADIUS Attribute	Format	Example
1 (Username)	12 hexadecimal digits, all lower case and without punctuation.	0019f910dec8
2 (Password)	This is the same format as the user but encrypted.	065836c1f1069758e6206f8096a24463
31 (Calling-Station-Id)	6 groups of 2 hexadecimal digits all upper case and separated by hyphens.	00-19-F9-10-DE-C8

Given that the MAB authentication uses the MAC address as user and password, make sure that the RADIUS server can differentiate between MAB petitions and other types of petitions for network access. This allows you to prevent

other clients from trying to use the MAC address as a valid credential. To uniquely identify a MAB petition, use Attribute 6 (Service-Type) with a value of 10 (Call-Check) in the RADIUS Access-Request message.

1.7.1 Processing authentication

Authentication begins when the switch receives a packet in a port configured with MAB authentication. A RADIUS Access-Request message is then sent to the RADIUS server. It includes the terminal's MAC address in the three attributes indicated above.

At this point, a dialog between the switch and the RADIUS server is created.

- If the MAC address is valid, the RADIUS server responds with a RADIUS Access-Accept message. This message indicates the switch must allow the terminal to access the network.
- If the MAC address isn't valid (or doesn't have permission to access the network), the RADIUS server responds with a RADIUS Access-Reject message. This message indicates that the switch cannot allow the terminal to access the network.

Chapter 2 Configuring the 802.1X and MAB Authentication

2.1 Using 802.1X authentication in Teldat devices

802.1X authentication can be configured in the following interfaces:

- Ethernet interfaces. This can act as authenticator or supplicant.
- Ethernet interfaces with switch. It's possible to configure 802.1X authentication in each switch port. This feature is not available for all switches. Only the authenticator is supported in these interfaces.
- WLAN interfaces. 802.1X authentication in these interfaces does not follow the same configuration system as the rest of the interfaces. For further information on 802.1X authentication configuration in WLAN interfaces, please see manual *Teldat-Dm 771-I "Wireless LAN Interface"*.



Note

From here onwards, this manual shall focus on 802.1X authentication in Ethernet interfaces.

For 802.1X authentication to operate in an Ethernet interface or port, it must be globally enabled and not simply configured in the corresponding menu.

2.2 Using MAB authentication in Teldat devices

MAB authentication can be configured in the following interfaces:

- Ethernet interfaces with switch. It's possible to configure MAB authentication per switch port.

MAB authentication is configured by accessing a *dot1X* configuration menu for each switch port.



Note

This feature is not available for all switches.

2.3 802.1X Authentication Configuration Menus

802.1X authentication is configured in the interface or port configuration menu. On top of this option, a global menu allows parameters that are common to all ports using 802.1X authentication to be configured.

2.3.1 Accessing the global configuration menu

To access 802.1X authentication global configuration, run **protocol dot1x** (general configuration menu).

```
*config
Config>protocol dot1x
-- 802.1X User Config --
dot1X config>
```

2.3.2 Accessing the configuration menu for an Ethernet interface

To access the 802.1X authentication configuration menu for an Ethernet interface, run **dot1x** (interface configuration menu).

```
*config
Config>network ethernet0/0
-- Ethernet Interface User Configuration --
ethernet0/0 config>dot1x
-- 802.1X User Config --
ethernet0/0 dot1X config>
```

2.3.3 Accessing the configuration menu for a one switch port

To access the 802.1X authentication configuration menu for a one-switch port internally connected to an Ethernet interface, run **port <n-port> dot1x** (switch configuration menu).

```
*config
Config>network ethernet1/0
-- Ethernet Interface User Configuration --
ethernet1/0 config>repeater
-- Repeater User Config --
ethernet1/0 repeater config>port 1 dot1x
-- 802.1X User Config --
ethernet1/0 (port 1) dot1x config>
```



Note

The configuration commands for a one-switch port are a subset of the configuration commands for an Ethernet interface. When detailing the different commands, those unavailable to a switch port are pointed out.

2.4 Global Configuration Commands

This section summarizes the 802.1X authentication global configuration commands for all devices.

The following table shows the 802.1X authentication global configuration commands. These are further explained in the following paragraphs.

Command	Function
<i>? (HELP)</i>	Displays the configuration commands or their options.
<i>NO</i>	Configures parameters with their default values.
<i>SYSTEM-AUTH-CONTROL</i>	Globally enables 802.1X authentication in the device.
<i>EXIT</i>	Exits the 802.1X authentication global configuration menu.

2.4.1 ? (HELP)

Displays the configuration commands or their options.

2.4.2 NO

Sets parameters at their default value or deletes the configuration.

Syntax:

```
dot1x config>no ?
  system-auth-control    Globally enables 802.1X
```

Example:

Globally disables 802.1X authentication in the device.

```
dot1x config>no system-auth-control
```

2.4.3 SYSTEM-AUTH-CONTROL

Globally enables 802.1X authentication in the device.

Syntax:

```
dot1x config>system-auth-control
```

802.1X authentication is globally disabled by default.



Note

For 802.1X authentication to run in Ethernet ports, you need to globally enable it in the device. This command does not affect 802.1X authentication operations in WLAN interfaces.

2.4.4 EXIT

Exits the 802.1X authentication global configuration menu.

Syntax:

```
dot1X config>exit
```

2.5 Per Port Configuration Commands

This section summarizes the different configuration commands for 802.1X authentication and MAB authentication per port.

The following table shows the 802.1X authentication configuration commands per port. These are further explained in the following paragraphs.

Command	Function
<i>? (HELP)</i>	Displays the configuration commands or their options.
<i>AUTHENTICATOR</i>	Configures the authenticator parameters.
<i>NO</i>	Configures parameters with their default values.
<i>PAE</i>	Configures the authenticator or supplicant mode.
<i>SUPPLICANT</i>	Configures the supplicant parameters.
<i>EXIT</i>	Exits the 802.1X authentication configuration menu for a port.

2.5.1 ? (HELP)

Displays the available commands or their options.

2.5.2 AUTHENTICATOR

Configures the authenticator parameters.

Syntax:

```
ethernet0/0 dot1X config>authenticator
  authentication      Set AAA authentication options
    dot1x <list name>  AAA list name to use for 802.1X
  controlled-directions 802.1X controlled directions
    both              Both (In and Out) packet flows controlled by
                      802.1X
    in                Only In packets controlled by 802.1X
  key-transmit        Enables key transmission
  port-control        Port control
    auto              Port status controlled by 802.1X
    force-authorized  Port forced to authorized
    force-unauthorized Port forced to unauthorized
  quiet-period <period> Time between authentications
  reauth-max <value>   Reauthentication attempts permitted before the port
                      becomes Unauthorized
  reauth-period <period> Time between reauthentications
  reauthentication    Enables reauthentication
  server-timeout <period> Server timeout
```

2.5.2.1 AUTHENTICATOR AUTHENTICATION

Configures the method lists used for authentication purposes. Method lists are defined using the AAA feature. For more information on how to configure said feature, please see Teldat Dm800-I AAA feature.

AUTHENTICATOR AUTHENTICATION DOT1X

Configures the methods list to use for 802.1X authentication.

Command history:

Release	Modification
11.01.05	This command was introduced as of version 11.01.05.

2.5.2.2 AUTHENTICATOR CONTROLLED-DIRECTIONS

Configures the direction of the data controlled through 802.1X authentication. If the controlled port is not authorized, data flow is not permitted. This parameter defines whether the data flow is controlled in both directions or only at input.



Note

This command is not available for switch ports.

AUTHENTICATOR CONTROLLED-DIRECTIONS BOTH

802.1X authentication controls the data flow in both directions. Data cannot be sent or received through a controlled port until the 802.1X authentication has placed the port in an authorized state.

AUTHENTICATOR CONTROLLED-DIRECTIONS IN

802.1X authentication controls the data flow in the entry. If the controlled port is in an unauthorized state, it does not receive data. However, it does allow data to exit.

Default is 802.1X authentication controls data flow in both directions (*authenticator controlled-directions both*).

2.5.2.3 AUTHENTICATOR KEY-TRANSMIT

Allows key transmission as part of the 802.1X authentication process.

Key transmission is allowed by default.

2.5.2.4 AUTHENTICATOR Mode

Configures the authentication mode in the port.



Note

This command is only available for switch ports.

AUTHENTICATOR MODE MAB

Enables the MAC Authentication Bypass as the authentic mode in the port.

Default is MAB disabled.

2.5.2.5 AUTHENTICATOR PORT-CONTROL

Configures port control through 802.1X authentication. Port status can be set to authorized or not authorized, or can be configured so that the status reflects the 802.1X authentication results.

AUTHENTICATOR PORT-CONTROL AUTO

Port status and control relies on normal 802.1X authentication operations. If 802.1X authentication is not satisfactorily completed, said port is not authorized.



Note

On top of configuring this port so that 802.1X authentication is carried out, you must also globally enable 802.1X authentication (**system-auth-control-global** configuration menu)

AUTHENTICATOR PORT-CONTROL FORCE-AUTHORIZED

Forces port authorization without executing authentication. Devices connected to this port have network access.

AUTHENTICATOR PORT-CONTROL FORCE-UNAUTHORIZED

Forces port deauthorization. Devices connected to this port do not have network access.

The port status is given via 802.1X authentication (**authenticator port-control auto**) by default.

2.5.2.6 AUTHENTICATOR QUIET-PERIOD

When the authenticator cannot authenticate a supplicant, it waits for the configured time (set using this parameter) to time out before retrying the authentication process.

Syntax:

```
ethernetx/x dot1X config>authenticator quiet-period ?
<0..65535>      Value in the specified range
```

By default, 60 seconds elapse between authentication attempts.

2.5.2.7 AUTHENTICATOR REAUTH-MAX

Configures the maximum number of possible reauthentications without the port switching to an unauthorized state. If the authenticator exceeds the number of reauthentications set by this parameter, the controlled port is unauthorized before new authentication efforts can be attempted.

Syntax:

```
ethernetx/x dot1X config>authenticator reauth-max ?
<1..255>      Value in the specified range
```

By default, two authentication attempts can be made before the port becomes unauthorized.



Note

This command is not available for some switches.

Command history:

Release	Modification
11.01.08	This command was made available to some switches as of version 11.01.08.

2.5.2.8 AUTHENTICATOR REAUTH-PERIOD

Configures the time between reauthorizations. Where reauthorizations are permitted, this parameter indicates how often reauthentication from the supplicant should be forced.

Syntax:

```
ethernetx/x dot1X config>authenticator reauth-period ?
<1..65535>      Value in the specified range
```

A one-hour (3600 seconds) interval between reauthentications is set by default.



Note

This command is not available for some switches.

Command history:

Release	Modification
11.01.08	This command was made available to some switches as of version 11.01.08.

2.5.2.9 AUTHENTICATOR REAUTHENTICATION

Allows reauthentications. Once a port is authorized, this forces reauthorization each time the time period indicated in **authenticator reauth-period** parameter times out.

Reauthentications is disabled by default.



Note

This command is not available for some switches.

Release

11.01.08

Modification

This command was made available to some switches as of version 11.01.08.

2.5.2.10 AUTHENTICATOR SERVER-TIMEOUT

Configures the timer used to detect lack of response in communications between the authenticator and the authentication server.

Syntax:

```
ethernetx/x dot1X config>authenticator server-timeout ?
<1..65535>      Value in the specified range
```

The default value is 30 seconds. If, after sending a packet to the authentication server, a response has not been received within this period, a timeout is considered to have happened and the appropriate actions are taken.

2.5.3 NO

Sets the parameters at their default values or deletes the configuration.

Syntax:

```
ethernetx/x dot1X config>no
  authenticator
    controlled-directions  802.1X controlled directions
    key-transmit            Enables key transmission
    mode                   Authentication mode
    port-control            Port control
    quiet-period            Time between authentications
    reauth-max              Reauthentication attempts permitted before
                           the port becomes Unauthorized
    reauth-period           Time between reauthentications
    reauthentication        Enables reauthentication
    server-timeout          Server timeout
  pae                      Configure Port Access Entity (PAE)
  supplicant
    auth-period             802.1x supplicant state machine authPeriod
    ca-cert                 Trusted CA certificate
    eap-method              Configure EAP method
    held-period             802.1x supplicant state machine heldPeriod
    max-start               802.1x supplicant state machine maxStart
    password                Supplicant authentication password
    start-period            802.1x supplicant state machine startPeriod
    user-cert               User certificate
    username                Supplicant identity configuration
```

Example:

Configuration for the default time period between reauthentications.

```
ethernet0/0 dot1X config>no authenticator reauth-period
```

2.5.4 PAE

Configures the behavior mode as either authenticator or supplicant.

Syntax:

```

ethernet0/0 dot1X config>pae
  authenticator      Authenticator Port Access Entity (PAE)
  supplicant         Supplicant Port Access Entity (PAE)

```

Default is authenticator.

**Note**

This command is not available for switch ports.

2.5.5 SUPPLICANT

Configures the supplicant parameters.

Syntax:

```

ethernet0/0 dot1X config>supplicant
  auth-period      802.1x supplicant state machine authPeriod
  ca-cert          Trusted CA certificate
  ca-cert-inner    Inner method trusted CA certificate
  eap-method       Configure EAP method
    peap           Use PEAP method
      eap-mschapv2  Use EAP-MSCHAPV2 inner method
      eap-md5       Use EAP-MD5 inner method
      eap-gtc       Use EAP-GTC inner method
      eap-otp       Use EAP-OTP inner method
    tls            Use TLS method
    ttls           Use TTLS method
      eap-mschapv2  Use EAP-MSCHAPV2 inner method
      eap-md5       Use EAP-MD5 inner method
      eap-gtc       Use EAP-GTC inner method
      eap-otp       Use EAP-OTP inner method
      eap-tls       Use EAP-TLS inner method
    mschapv2       Use MSCHAPV2 inner method
    mschap         Use MSCHAP inner method
    chap           Use CHAP inner method
    pap            Use PAP inner method
  held-period      802.1x supplicant state machine heldPeriod
  max-start        802.1x supplicant state machine maxStart
  password         Supplicant authentication password
  start-period     802.1x supplicant state machine startPeriod
  user-cert        User certificate
  user-cert-inner  Inner method user certificate
  username         Supplicant identity configuration

```

**Note**

These commands are not available for switch ports.

2.5.5.1 SUPPLICANT AUTH-PERIOD

Time period during which the supplicant waits for a request from the authenticator before assuming the authentication process has failed.

Syntax:

```

ethernetx/x dot1X config> supplicant auth-period ?
<1..65535>      Value in the specified range

```

By default, 30 seconds must go by before authentication is considered failed.

2.5.5.2 SUPPLICANT CA-CERT

Configures the certificate belonging to the trusted certifier to check the validity of the certificate the authenticator sends when triggering 802.1x sessions. If this command is executed in the dynamic configuration, the certificate must be preloaded in the device.

If you don't configure the trusted certifier authority's certificate, authenticator certificate validity is not executed. However, authentication can terminate satisfactorily even if security is lower (since the authenticity of the server cannot be verified). Nevertheless, confidentiality between the device and the authenticator is guaranteed as the data is encrypted.

To load a certificate in device base64, execute **certificate <certname> base64** (ipsec certificates menu) and enter said certificate. For further information, please see the section on *Certificates* in manual *Teldat Dm739-I "IPSEC"*.

Syntax:

```
ethernetx/x dot1X config> supplicant ca-cert ?
<word>      Text
```

2.5.5.3 SUPPLICANT CA-CERT-INNER

Configures the certifier authority certificate used in the internal session when TTLS/TLS is used. Said certificate is used to check the validity of the information the authenticator sends. The certificate must be preloaded in the device if you execute this command in the dynamic configuration. Where you don't configure this parameter, the certificate used is configured through **supplicant ca-cert**.

If you don't configure a certifier authority certificate, authenticator certificate validity is not executed. However, authentication can terminate satisfactorily even if security is lower (since the authenticity of the server cannot be verified). Nevertheless, confidentiality between the device and the authenticator is guaranteed as the data is encrypted.

To load a certificate in base64 in the device, execute **certificate <certname> base64** (ipsec certificates menu) and enter the certificate. For further information, please see the section on *Certificates* under manual *Teldat-Dm 739-I "IPSEC"*.

Syntax:

```
ethernetx/x dot1X config> supplicant ca-cert-inner ?
<word>      Text
```

2.5.5.4 SUPPLICANT EAP-METHOD

Configures the EAP method the supplicant uses in the authentication process. If this is not configured, all supported methods are allowed. The authenticator must choose which method to use.

SUPPLICANT EAP-METHOD PEAP

Configures the use of the Protected Extensible Authentication Protocol (PEAP) in the authentication process. PEAP is a safe method to transmit authentication information. This requires a certificate in the authentication server that can be verified by the client, provided that a certifier authority is configured (**supplicant ca-cert**). An SSL/TLS encrypted tunnel is created to transmit the authentication data. Optionally, you can specify an internal authentication method for the PEAP. If none is configured, all are supported. The internal methods need a user and password to be configured (**supplicant username** and **supplicant password** commands).

Syntax:

```
ethernetx/x dot1X config> supplicant eap-method peap
      <cr>
      eap-mschapv2      Use EAP-MSCHAPV2 inner method
      eap-md5           Use EAP-MD5 inner method
      eap-gtc           Use EAP-GTC inner method
      eap-otp           Use EAP-OTP inner method
```

SUPPLICANT EAP-METHOD TLS

Configures the use of TLS in the authentication process. This is a safe method based on mutual authentication between certificates, which is carried out when a TLS session is established. This requires a certificate in the authentication server that can be verified by the client, provided that a certifier authority is configured (**supplicant ca-cert**). In addition, this method needs a user certificate to be configured with a private password to identify the device. Said certificate is configured by running **supplicant user-cert**.

Syntax:

```
ethernetx/x dot1X config> supplicant eap-method tls
```

SUPPLICANT EAP-METHOD TTLS

Configures the use of Tunneled Transport Layer Security (TTLS) in the authentication process. This is a safe method similar to PEAP and only requires a certificate in the server that can be verified by the client (provided a certifier authority is configured - **supplicant ca-cert**). A TLS tunnel is established to exchange authentication data. Optionally, you can specify an internal authentication method for the TTLS. If none is configured, all are supported. All internal methods, except for **eap-tls**, need a user and password to be configured (**supplicant username** and **supplicant password** commands). The **eap-tls** method requires a certificate to be configured with a private password (**supplicant user-cert-inner**).

Syntax:

```
ethernetx/x dot1X config> supplicant eap-method ttls ?
      <cr>
      eap-mschapv2      Use EAP-MSCHAPV2 inner method
      eap-md5           Use EAP-MD5 inner method
      eap-gtc           Use EAP-GTC inner method
      eap-otp           Use EAP-OTP inner method
      eap-tls           Use EAP-TLS inner method
      mschapv2          Use MSCHAPV2 inner method
      mschap            Use MSCHAP inner method
      chap              Use CHAP inner method
      pap               Use PAP inner method
```

2.5.5.5 SUPPLICANT HELD-PERIOD

Time the supplicant state machine waits before retrying authentication when the authentication process has not finalized.

Syntax:

```
ethernetx/x dot1X config> supplicant held-period ?
<1..65535>      Value in the specified range
```

Default is 60 seconds between retries.

2.5.5.6 SUPPLICANT MAX-START

Configures the maximum number of petition retries to start authentication (EAPOL-Start) before assuming authentication has failed.

Syntax:

```
ethernetx/x dot1X config> supplicant max-start ?
<1..65535>      Value in the specified range
```

Default is 3 EAPOL-Starts sent.

2.5.5.7 SUPPLICANT PASSWORD

Configures the password to use in the authentication process.

SUPPLICANT PASSWORD PLAIN

Lets you enter the password in plain text. To enter the password in clear, run **supplicant password plain**.

Syntax:

```
ethernetx/x dot1X config> supplicant password plain ?
<word>      Text
```

SUPPLICANT PASSWORD CIPHERED

Lets you enter the password in encrypted form. The password is encrypted and saved by running **supplicant password ciphered**.

Syntax:

```
ethernetx/x dot1X config>supplicant password ciphered ?
```

```
<word>      Text
```

Example:

```
ethernet0/0 dot1X config$supplicant password plain secret
ethernet0/0 dot1X config$
ethernet0/0 dot1X config$show config
      supplicant password ciphered 0x6E90D5292F478EB6
ethernet0/0 dot1X config$
```

2.5.5.8 SUPPLICANT START-PERIOD

Time the supplicant state machine waits before trying to send the authentication start petition (EAPOL-Start) again when a response from the authenticator hasn't been received.

Syntax:

```
ethernetx/x dot1X config> supplicant start-period ?
<1..65535>      Value in the specified range
```

Default is 30 seconds between retries.

2.5.5.9 SUPPLICANT USER-CERT

Configures the certificate used by the device when it authenticates through TLS. Said certificate must be preloaded in the device and have an associated private password.

To generate a private password and load the signed certificate in the device, please see the section on *Certificates* found in manual *Teldat-Dm 739-I "IPSEC"*.

Syntax:

```
ethernetx/x dot1X config>supplicant user-cert ?
<word>      Text
```

2.5.5.10 SUPPLICANT USER-CERT-INNER

Configures the certificate used by the device in the internal session when TTLS/TLS is being used. Said certificate must be preloaded in the device and be associated to a private password.

To generate a private password and load the signed certificate in the device, please see the section on *Certificates* found in manual *Teldat-Dm 739-I "IPSEC"*.

If you don't configure this parameter, the certificate used is configured by running **supplicant user-cert**.

Syntax:

```
ethernetx/x dot1X config>supplicant user-cert-inner ?
<word>      Text
```

2.5.5.11 SUPPLICANT USERNAME

Configures the user name used in the authentication process.

Syntax:

```
ethernetx/x dot1X config> supplicant username ?
<word>      Text
```

2.5.6 EXIT

Exits 802.1X authentication for a port configuration menu.

Syntax:

```
ethernetx/x dot1X config>exit
```

Chapter 3 802.1X Authentication Monitoring

3.1 802.1X Authentication Monitoring Menus

This is very similar to configuration, with a global monitoring menu and a menu associated to each Ethernet port.

3.1.1 Accessing the global monitoring menu

To access 802.1X authentication global monitoring, run **protocol dot1x** (general monitoring menu).

```
*monitor
Console Operator
+protocol dot1x
-- 802.1X Console --
dot1X+
```

3.1.2 Accessing the monitoring menu for an Ethernet interface

To access the 802.1X authentication monitoring menu for an Ethernet interface, run **dot1x** (interface monitoring menu).

```
*monitor
Console Operator
+net ethernet0/0
-- Ethernet Console --
ethernet0/0 ETH+dot1x
-- 802.1X Console --
ethernet0/0 DOT1X+
```

3.1.3 Accessing the monitoring menu for a one switch port

To access the 802.1X authentication monitoring menu for a one-switch port internally connected to an Ethernet interface, run **dot1x <n-port>** (switch monitoring menu).

```
*monitor
Console Operator
+net ethernet1/0
-- Ethernet Console --
ethernet1/0 ETH+repeater
-- Repeater Monitoring Console --
ethernet1/0 Repeater+dot1x 1
-- 802.1X Console --
ethernet1/0 (port 1) DOT1X+
```

**Note**

The monitoring commands for a switch port are a subset of monitoring commands for an Ethernet interface. When the various commands are detailed, those unavailable for a switch port are indicated.

3.2 Global Monitoring Commands

This section summarizes the different monitoring commands for 802.1X authentication for all devices.

The following table summarizes the 802.1X authentication global monitoring commands. These are further explained in the following paragraphs.

Command	Function
? (HELP)	Displays the configuration commands or their options.
SHOW	Displays information relative to the 802.1X authentication.
SUPPLICANT	Monitoring the supplicant.
EXIT	Exits the 802.1X authentication global monitoring menu.

3.2.1 ? (HELP)

Displays the available commands or their options.

3.2.2 SHOW

Displays information relative to 802.1X authentication.

Syntax:

```
dot1X+show
  system      system monitoring
             configuration      system configuration
```

SHOW SYSTEM CONFIGURATION

Displays information for each Ethernet port that supports 802.1X authentication, for the EAPOL version being executed and the role the port can take: supplicant, authenticator or both. It also displays the state of the **SystemAuthControl** variable and shows whether 802.1X authentication is globally enabled in the device.

Example:

```
dot1X+show system configuration
SystemAuthControl: Enabled
```

Interface	Port	EAPOL Version	PAE Capabilities
ethernet0/0	1	2	Authenticator
ethernet0/1	1	2	Authenticator
ethernet1/0	1	2	Authenticator
ethernet1/0	2	2	Authenticator
ethernet1/0	3	2	Authenticator
ethernet1/0	4	2	Authenticator

```
dot1X+
```

3.2.3 SUPPLICANT

Supplicant monitoring commands.

Syntax:

```
dot1X+ supplicant ?
  trace-level      Set supplicant trace level
```

SUPPLICANT TRACE-LEVEL

Configures the debugging level for the supplicant's authentication process. Debugging traces are displayed through the DOT1X.026 event. Values between 0 and 5 are allowed, with 5 being the level where the most debugging traces are displayed. Default is 3.

Syntax:

```
dot1X+ supplicant trace-level ?
<0..5>      Value in the specified range
```

3.2.4 EXIT

Exits the 802.1X authentication global monitoring menu.

Syntax:

```
dot1X+exit
```

3.3 Per Port Monitoring Commands

This section summarizes the different monitoring commands for 802.1X authentication per port.

The following table summarizes the 802.1X authentication monitoring commands per port. These are further explained in the following paragraphs.

Command	Function
? (HELP)	Displays the monitoring commands or their options.
MAB REAUTHENTICATE	Forces a MAB reauthentication.
REAUTHENTICATE	Forces a reauthentication.
SHOW AUTHENTICATOR	Displays information relative to the authenticator.
SHOW SUPPLICANT	Displays information relative to the supplicant.
EXIT	Exits the 802.1X authentication monitoring menu for a port.

3.3.1 ? (HELP)

Displays the available monitoring commands or their options.

3.3.2 MAB REAUTHENTICATE

Forces reauthentication when MAC Access Bypass (MAB) is used. It deletes all authorized MACs from the switch address table so that a new authentication takes place.

Syntax:

```
ethernet0/0 (port 1) DOT1X+mab reauthenticate
```

Release

11.01.08

Modification

The "*MAB reauthenticate*" command was introduced as of version 11.01.08.

3.3.3 REAUTHENTICATE

Forces reauthentication. Applicable in authenticator and supplicant mode. If authentication is in progress, reauthentication does not occur until the current authentication is complete.

Syntax:

```
ethernet0/0 dot1X+reauthenticate
```



Note

This command is not available for some switches.

Release

11.01.08

Modification

The "*Reauthenticate*" command was made available for some switches as of version 11.01.08.

3.3.4 SHOW AUTHENTICATOR

Displays information relative to the authenticator.

Syntax:

```
dot1X+show authenticator
  configuration      authenticator configuration
  diagnostics        authenticator diagnostics
  session-statistics authenticator session statistics
  statistics         authenticator statistics
```

SHOW AUTHENTICATOR CONFIGURATION

Displays information relative to the configuration of the authenticator associated to an Ethernet port.

Example:

```
ethernet0/0 DOT1X+show authenticator configuration
Interface: ethernet0/0 (port 1)
Authenticator PAE state : FORCE_AUTH
Backend Authentication state : AUTH_INITIALIZE
AdminControlledDirections: Both
```

```

OperControlledDirection: Both
AuthControlledPortControl: ForceAuthorized
AuthControlledPortStatus: Authorized
quietPeriod: 60
serverTimeout: 30
reAuthPeriod: 3600
reAuthEnabled: Disabled
KeyTransmissionEnabled: Yes
ethernet0/0 DOT1X+

```

The following information is shown:

- **Interface**: information on the interface and port.
- **Authenticator PAE state**: status of the states machine associated to the authenticator communication with the suppliant.
- **Backend Authentication state**: status of the states machine associated to the authenticator communication with the authentication server.
- **AdminControlledDirections**: configuration of directions controlled by 802.1X authentication: incoming packets (In) or incoming and outgoing (Both).
- **OperControlledDirections**: indicates the directions controlled by 802.1X authentication. Normally, this value matches the configured value. However, the execution of the state machines can make this parameter take a value of *Both* when the configured value is *In*.
- **AuthControlledPortControl**: port control configuration through 802.1X authentication. The values allowed are *ForceAuthorized* if the port is always authorized without having to perform 802.1X authentication, *ForceUnauthorized* if the port is always unauthorized without carrying out 802.1X authentication and *Auto* if the port status is determined by 802.1X authentication.
- **AuthControlledPortStatus**: port status, *Authorized* or *Unauthorized*.
- **quietPeriod**: value configured for this variable: this is the time period the authenticator must wait before retrying the authentication process in cases where there is an error.
- **serverTimeout**: value configured for this variable: time period used to wait for a response from the authentication server before deciding there is no response.
- **reAuthPeriod**: time between reauthentications.
- **reAuthEnabled**: indicates if authentication is enabled or not.
- **KeyTransmissionEnabled**: indicates if key transmission is enabled or not.



Note

These statistics do not update correctly for switch ports.

SHOW AUTHENTICATOR DIAGNOSTICS

Displays useful information on authenticator operations. This information refers to the state machines defined in the 802.1X standard. You need to have some training on state machines to be able to interpret the results correctly.

Example:

```

ethernet0/0 DOT1X+show authenticator diagnostics
Interface: ethernet0/0 (port 1)
authEntersConnecting: 0
authEapLogoffsWhileConnecting: 0
authEntersAuthenticating: 0
authAuthSuccessWhileAuthenticating: 0
authAuthTimeoutsWhileAuthenticating: 0
authAuthFailWhileAuthenticating: 0
authAuthEapStartsWhileAuthenticating: 0
authAuthEapLogoffWhileAuthenticating: 0
authAuthReauthWhileAuthenticating: 0
authAuthEapStartsWhileAuthenticated: 0
authAuthEapLogoffWhileAuthenticated: 0
backendResponses: 0
backendAccessChallenges: 0
backendOtherRequestsToSuppliant: 0
backendAuthSuccesses: 0
backendAuthFails: 0
ethernet0/0 DOT1X+

```

The following information is displayed:

- *Interface*: information on the interface and port.
- *authEntersConnecting*: Number of times the states machine switches to a *CONNECTING* state from any other state.
- *authEapLogoffsWhileConnecting*: Number of times the states machine switches from *CONNECTING* to *DISCONNECTED* after receiving an EAPOL-Logoff message.
- *authEntersAuthenticating*: Number of times the states machine switches from *CONNECTING* to *AUTHENTICATING* after receiving an EAPOL-Response/Identity message.
- *authAuthSuccessWhileAuthenticating*: Number of times the states machine switches from *AUTHENTICATING* to *AUTHENTICATED* after a successful authentication.
- *authAuthTimeoutsWhileAuthenticating*: Number of times the states machine switches from *AUTHENTICATING* to *ABORTING* after a timeout in authentication.
- *authAuthFailWhileAuthenticating*: Number of times the states machine switches from *AUTHENTICATING* to *HELD* after a failed authentication.
- *authAuthEapStartsWhileAuthenticating*: Number of times the states machine switches from *AUTHENTICATING* to *ABORTING* after receiving an EAPOL-Start message.
- *authAuthEapLogoffWhileAuthenticating*: Number of times the states machine switches from *AUTHENTICATING* to *ABORTING* after receiving an EAPOL-Logoff message.
- *authAuthReauthWhileAuthenticating*: Number of times the states machine switches from *AUTHENTICATED* to *RE-START* after a reauthentication request.
- *authAuthEapStartsWhileAuthenticated*: Number of times the states machine passes from an *AUTHENTICATED* state to *CONNECTING* after receiving an EAPOL-Start message.
- *authAuthEapLogoffWhileAuthenticated*: Number of times the states machine switches from *AUTHENTICATED* to *DISCONNECTED* after receiving an EAPOL-Logoff message.
- *backendResponses*: Number of times that the first response from the supplicant is sent to the communication layer with the authentication server.
- *backendAccessChallenges*: Number of times that the first request is received from the authentication server after the first response from the supplicant.
- *backendOtherRequestsToSupplicant*: Number of times an EAP-Request packet is sent after sending the first packet to the supplicant.
- *backendAuthSuccesses*: Number of times a success indication is received from the authentication server.
- *backendAuthFails*: Number of times an authentication failure indication is received from the authentication server.

SHOW AUTHENTICATOR SESSION-STATISTICS

Displays the statistics for the current session. When the port is unauthorized, the statistics correspond to the last session established.

Example:

```

ethernet0/0 DOT1X+show authenticator session-statistics
Interface: ethernet0/0 (port 1)
Session Octets Received: 33548
Session Octets Transmitted: 22
Session Frames Received: 332
Session Frames Transmitted: 1
Session Identifier: ethernet0/0-port1-session1
Session Authentication Method: Remote Authentication Server
Session Time: 15s
Session Terminate Cause: Not Terminated Yet
Session User Name: Sampledot1x

```

The following information is displayed:

- *Interface*: information on the interface and port.
- *Session Octets Received*: number of octets received in the current session.
- *Session Octets Transmitted*: number of octets transmitted in the current session.
- *Session Frames Received*: number of frames received in the current session.
- *Session Frames Transmitted*: number of frames transmitted in the current session.
- *Session Identifier*: Session identifier for this authenticator.
- *Session Authentication Method*: authentication method used. This can be local (Local Authentication Server) or remote (Remote Authentication Server).

- *Session Time*: session duration in seconds.
- *Session Terminate Cause* : possible causes are:

Supplicant Logoff: an EAPOL-Logoff packet has been received from the supplicant.

Port Failure: an error has occurred in the port (authentication error, interface down, etc).

Supplicant Restart: an EAPOL-Start packet has been received from the supplicant.

Reauthentication Failure: a failure has occurred when reauthenticating the supplicant.

Port Control forced to unauthorized: the port state has been forcibly changed to unauthorized.

Port re-initialization: the port has been re-initialized.

Port Administratively Disabled: the port is administratively inoperative.

Not Terminated Yet: the session is still active.

- *Session User Name*: supplicant identity.

SHOW AUTHENTICATOR STATISTICS

Displays authenticator statistics

Example:

```
ethernet0/0 DOT1X+show authenticator statistics
Interface: ethernet0/0 (port 1)
EAPOL frames received: 0
EAPOL frames transmitted: 1
EAPOL Start frames received: 0
EAPOL Logoff frames received: 0
EAP Resp/Id frames received: 0
EAP Response frames received: 0
EAP Initial Request frames transmitted: 0
EAP Request frames transmitted: 0
Invalid EAPOL frames received: 0
EAP length error frames received: 0
Last EAPOL frame version: 0
```

The following information is displayed:

- *Interface*: information on the interface and port.
- *EAPOL frames received*: Number of valid EAPOL frames received.
- *EAPOL frames transmitted*: Number of EAPOL frames transmitted.
- *EAPOL Start frames received*: Number of *EAPOL-Start* frames received.
- *EAPOL Logoff frames received*: Number of *EAPOL-Logoff* frames received.
- *EAP Resp/Id frames received*: Number of *EAP-Response/Identity* frames received.
- *EAP Response frames received*: Number of *EAP-Response* frames different from the *EAP-Response/Identity* received.
- *EAP Initial Request frames transmitted*: Number of *EAP Initial Request* frames transmitted.
- *EAP Request frames transmitted*: Number of *EAP-Request* frames different from the *EAP Initial Request* transmitted.
- *Invalid EAPOL frames received*: Number of EAPOL frames received with an unknown frame type.
- *EAP length error frames received*: Number of EAPOL frames received with an error in the field indicating packet body length.
- *Last EAPOL frame version*: Protocol version indicated in the latest EAPOL frame received.

3.3.5 SHOW SUPPLICANT

Displays information relative to the supplicant.

Syntax:

```
dot1X+ show supplicant ?
  statistics      Supplicant statistics
  status          Supplicant configuration
```

**Note**

These commands are not available for switch ports.

SHOW SUPPLICANT STATISTICS

Displays supplicant statistics.

Example:

```
ethernet0/0 DOT1X+ show supplicant statistics
dot1xSuppEapolFramesRx: 3
dot1xSuppEapolFramesTx: 7
dot1xSuppEapolStartFramesTx: 5
dot1xSuppEapolLogoffFramesTx: 0
dot1xSuppEapolRespFramesTx: 2
dot1xSuppEapolReqIdFramesRx: 2
dot1xSuppEapolReqFramesRx: 0
dot1xSuppInvalidEapolFramesRx: 1
dot1xSuppEapLengthErrorFramesRx: 0
dot1xSuppLastEapolFrameVersion: 2
dot1xSuppLastEapolFrameSource: 00:a0:26:46:2f:d9
```

The following information is displayed:

- dot1xSuppEapolFramesRx: Number of EAPOL frames received.
- dot1xSuppEapolFramesTx: Number of EAPOL frames transmitted.
- dot1xSuppEapolStartFramesTx: Number of *EAPOL-Start* frames transmitted.
- dot1xSuppEapolLogoffFramesTx: Number of *EAPOL-Logoff* frames transmitted.
- dot1xSuppEapolRespFramesTx: Number of *EAP-Response* frames transmitted.
- dot1xSuppEapolReqIdFramesRx: Number of *EAP-Request/Identity* frames received.
- dot1xSuppEapolReqFramesRx: Number of *EAP-Response* frames received.
- dot1xSuppInvalidEapolFramesRx: Number of invalid EAPOL frames received.
- dot1xSuppEapLengthErrorFramesRx: Number of EAPOL frames received with a length error.
- dot1xSuppLastEapolFrameVersion: Version of the last EAPOL frame received.
- dot1xSuppLastEapolFrameSource: Source MAC address of the last EAPOL frame received.

SHOW SUPPLICANT STATUS

Displays the status of the supplicant.

Example:

```
ethernet0/0 DOT1X+ show supplicant status
Supplicant PAE state: AUTHENTICATED
suppPortStatus: Authorized
heldPeriod: 60
authPeriod: 30
startPeriod: 30
maxStart: 3
portControl: Auto
Supplicant Backend state: IDLE
EAP state: SUCCESS
selectedMethod: 13 (EAP-TLS)
reqMethod: 0
methodState: DONE
decision: UNCOND_SUCC
ClientTimeout: 60
```

The following information is displayed:

- Supplicant PAE state: state of the supplicant's state machine.
- suppPortStatus: state of the port (i.e. *Authorized* or *Unauthorized*).
- heldPeriod: amount of time the supplicant's state machine waits before retrying the authentication when the authentication process has not completed.
- authPeriod: amount of time the supplicant's state machine waits for a request from the authenticator before assuming the authentication process has failed.
- startPeriod: amount of time the supplicant's state machine waits before retrying to send the authentication start request (EAPOL-Start) when it doesn't receive a response from the authenticator.
- maxStart: configures the maximum number of authentication start requests (EAPOL-Start) to be sent before assuming authentication has failed.
- portControl: state of port control.
- Supplicant Backend state: state of the supplicant's Backend state machine.
- EAP state: state of the EAP transaction.
- selectedMethod: EAP authentication method selected.
- reqMethod: method received in the current EAP petition.
- methodState: state of the authentication method.
- decision: authentication method results.
- ClientTimeout: amount of time spent before assuming an EAP authentication process has timed out.

3.3.6 EXIT

Exits the 802.1X authentication monitoring menu per port.

Syntax:

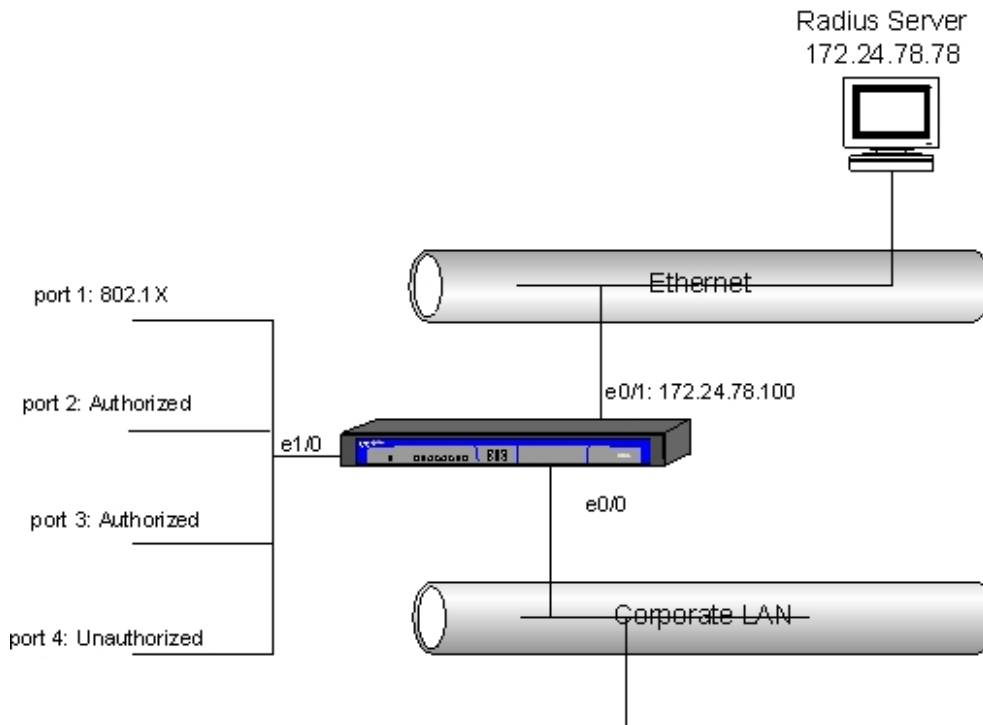
```
ethernet0/0 DOT1X+exit
```

Chapter 4 Configuration Examples

4.1 802.1X Authentication in a port

In the following example, we're going to explain the steps required to configure 802.1X authentication in a one-switch port.

4.1.1 Scenario



802.1X authentication

In the scenario displayed in the figure, we have a corporate LAN connected to the ethernet0/0 interface. To access this LAN, use the switch found in the device. We want the devices connected to port 1 to execute 802.1X authentication before gaining access to the corporate LAN. Devices connected to ports 2 and 3 have direct access to the corporate LAN without having to authenticate, while access to the corporate LAN from the remaining switch ports is denied.

4.1.2 Globally configuring 802.1X authentication

First, globally enable 802.1X authentication in the device.

```
*config
Config>protocol dot1x
-- 802.1X User Config --
dot1X config>system-auth-control
```

4.1.3 Configuring 802.1X authentication in Ethernet ports

The Ethernet0/0 interface, to which the corporate LAN is connected to, does not execute 802.1X authentication. Consequently, the status is forced to authorize access.

```
Config>net ethernet0/0
-- Ethernet Interface User Configuration --
ethernet0/0 config>dot1x
-- 802.1X User Config --
ethernet0/0 dot1X config>authenticator port-control force-authorized
```

In the switch, port 1 executes 802.1X authentication, ports 2 and 3 are authorized and the rest of the ports are unauthorized.

```

Config>net ethernet1/0
-- Ethernet Interface User Configuration --
ethernet1/0 config>repeater
-- Repeater User Config --
ethernet1/0 repeater config>port 1 dot1X
-- 802.1X User Config --
ethernet1/0 (port 1) dot1X config>authenticator port-control auto
ethernet1/0 (port 1) dot1X config>exit
ethernet1/0 repeater config>port 2 dot1X
-- 802.1X User Config --
ethernet1/0 (port 2) dot1X config>authenticator port-control force-authorized
ethernet1/0 (port 2) dot1X config>exit
ethernet1/0 repeater config>port 3 dot1X
-- 802.1X User Config --
ethernet1/0 (port 3) dot1X config>authenticator port-control force-authorized
ethernet1/0 (port 3) dot1X config>exit
ethernet1/0 repeater config>port 4 dot1X
-- 802.1X User Config --
ethernet1/0 (port 4) dot1X config>authenticator port-control force-authorized
ethernet1/0 (port 4) dot1X config>exit

```

4.1.4 Configuring Radius

Finally, configure access to the Radius server. For further information on Radius configuration, please see manual *Teldat-Dm 733-I "Radius Protocol"*.

- (1) Access the Radius configuration menu:

```

Config>feature radius
-- RADIUS User Configuration --
RADIUS config>

```

- (2) Configure the Radius server address:

```

RADIUS config>primary-address 172.24.78.78          RADIUS config>

```

- (3) Configure the key for the Radius server:

```

RADIUS config>primary-secret whatever
RADIUS config>

```

- (4) Enable Radius:

```

RADIUS config>enable radius
RADIUS enabled
RADIUS config>

```

- (5) Exit the Radius configuration menu:

```

RADIUS config>exit
Config>

```

- (6) Configure the IP address in the ethernet0/1 interface to access the Radius server:

```

Config>net ethernet0/1
-- Ethernet Interface User Configuration --
ethernet0/1 config>ip address 172.24.78.100 255.255.0.0

```

4.1.5 Complete configuration

The full configuration is shown below:

```

log-command-errors
no configuration
;
network ethernet0/0
; -- Ethernet Interface User Configuration --
    dot1x
; -- 802.1X User Config --
    authenticator port-control force-authorized
    exit
;
exit

```

```

;
network ethernet0/1
; -- Ethernet Interface User Configuration --
ip address 172.24.78.100 255.255.255.255
;
exit
;
network ethernet1/0
; -- Ethernet Interface User Configuration --
repeater
; -- Repeater User Config --
port 2 dot1X
; -- 802.1X User Config --
authenticator port-control force-authorized
exit
;
port 3 dot1X
; -- 802.1X User Config --
authenticator port-control force-authorized
exit
;
port 4 dot1X
; -- 802.1X User Config --
authenticator port-control force-unauthorized
exit
;
exit
;
exit
;
protocol dot1X
; -- 802.1X User Config --
system-auth-control
exit
;
feature radius
; -- RADIUS User Configuration --
primary-address 172.24.78.78
primary-secret whatever
enable radius
exit
;
dump-command-errors
end

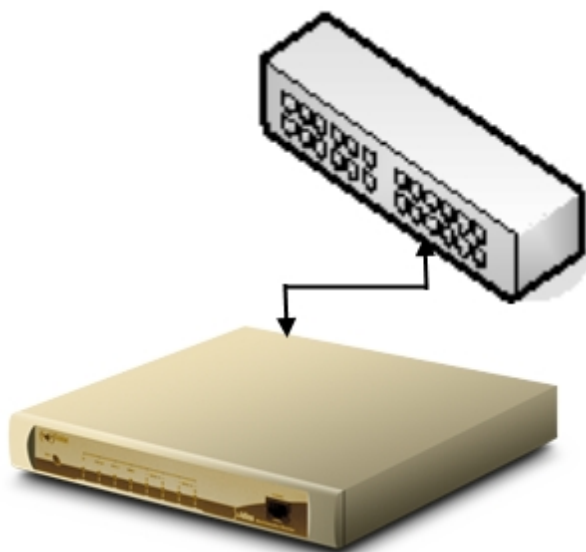
```

4.2 802.1X Supplicant in an Ethernet interface

In this example, we are going to describe the configuration for an 802.1X supplicant in an Ethernet port.

4.2.1 Scenario

The router's ethernet0/0 interface is connected to a switch with 802.1X authentication enabled. This means authentication is mandatory before traffic can be transmitted through the port. We don't want the router to have certificates, so we won't use EAP-TLS authentication, and the validity of the server certificate won't be checked. Additionally, we don't want to specify any particular method. Consequently, the authentication server can choose whatever it wants.



Scenario

4.2.2 802.1X supplicant: Global configuration

First, globally enable 802.1X authentication in the device.

```
*config
Config>protocol dot1x
-- 802.1X User Config --
dot1X config>system-auth-control
```

4.2.3 Configuring the 802.1X supplication in the Ethernet port

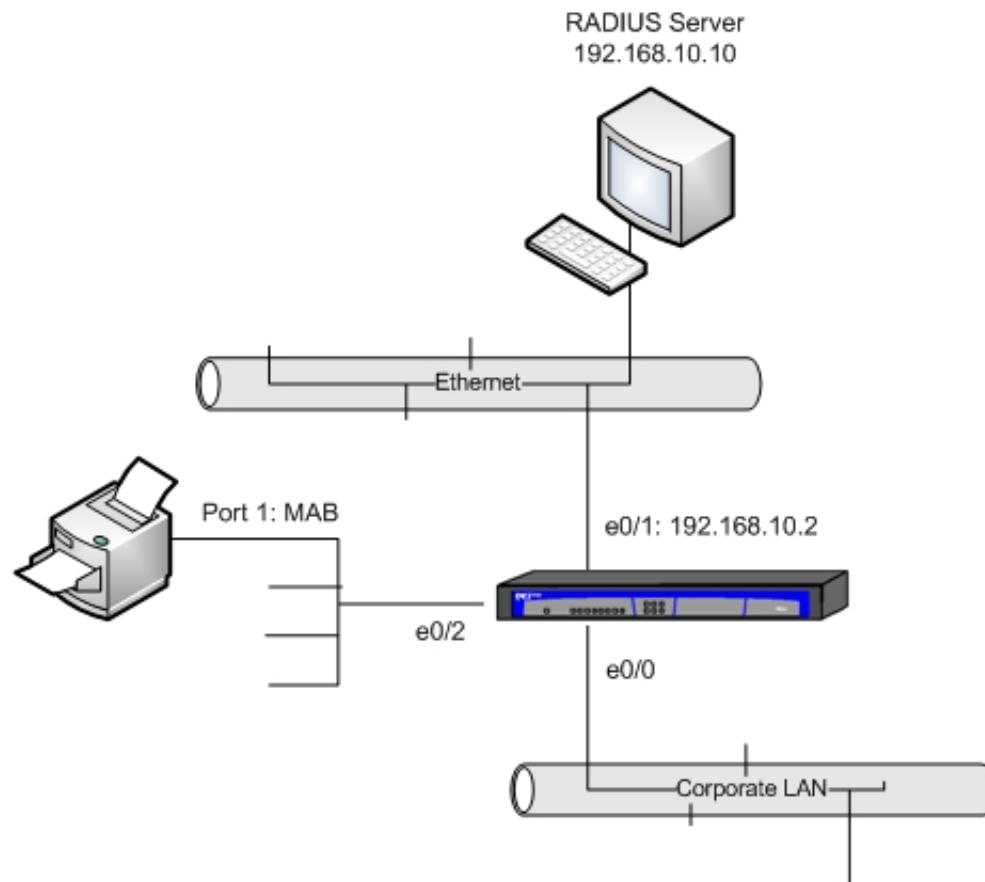
Configure the supplicant mode, the user and the password in the ethernet0/0 interface.

```
Config$network ethernet0/0
-- Ethernet Interface User Configuration --
ethernet0/0 config$dot1x
-- 802.1X User Config --
ethernet0/0 dot1x config$ pae supplicant
ethernet0/0 dot1x config$ supplicant username sample
ethernet0/0 dot1x config$ supplicant password plain secret
ethernet0/0 dot1x config$ show config
; -- 802.1X User Config --
    pae supplicant
    supplicant username sample
    supplicant password ciphered 0x6E90D5292F478EB6
ethernet0/0 dot1x config$
```

4.3 MAB Authentication in a port

In this example, we are going to detail the steps required to configure MAB authentication in a switch port.

4.3.1 Scenario



Scenario

In the scenario displayed in the figure, we have a corporate LAN connected to the ethernet0/0 interface. To access this LAN, use the switch in the device. We want the devices connected to port 1 to execute MAB authentication before gaining access to the corporate LAN. However, devices connected to the rest of the switch ports have direct access to said LAN without having to authenticate.

4.3.2 Configuring the MAB authentication

```
log-command-errors
no configuration
;
network ethernet0/2
; -- Ethernet Interface User Configuration --
;
    repeater-switch
; -- Switch User Config --
    port 1 dot1X
; -- 802.1X User Config --
    authenticator mode mab
    exit
;
    exit
;
    exit
;
;
network ethernet0/1
; -- Ethernet Interface User Configuration --
    ip address 192.168.10.2 255.255.255.0
;
    exit
;
;
feature radius
; -- RADIUS User Configuration --
    primary-address 192.168.10.10
    primary-secret whatever
    enable radius
    exit
;
dump-command-errors
end
```

Chapter 5 Annex

5.1 Third Party Software

A WPA Supplicant code is used for the 802.1X supplicant.

The WPA Supplicant license can be seen below:

WPA Supplicant

Copyright (c) 2003-2009, Jouni Malinen <j@w1.fi> and contributors

All Rights Reserved.

This program is dual-licensed under both the GPL version 2 and BSD license. Either license may be used at your discretion.

These are the terms of the BSD license:

Redistribution and use in source and binary forms, with or without modification, are permitted provided the following conditions are met:

- (1) Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- (2) Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- (3) Neither the name(s) of the above-listed copyright holder(s) nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The openssl library code is used for TLS negotiation.

The openssl license can be seen below:

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL please contact opensslcore@openssl.org.

OpenSSL License

Copyright (c) 1998-2003 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- (1) Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- (2) Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- (3) All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (<http://www.openssl.org/>)"
- (4) The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact opensslcore@openssl.org.
- (5) Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.
- (6) Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit"

(<http://www.openssl.org/>)"

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT

OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Original SSLeay License:

Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)

All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com).

The implementation was written so as to conform with Netscape's SSL.

This library is free for commercial and non-commercial use as long as the following conditions are adhered to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- (1) Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
- (2) Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- (3) All advertising materials mentioning features or use of this software must display the following acknowledgement: "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)"
The word 'cryptographic' can be left out if the routines from the library being used are not cryptographic related.
- (4) If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The license and distribution terms for any publicly available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution license [including the GNU Public License.]