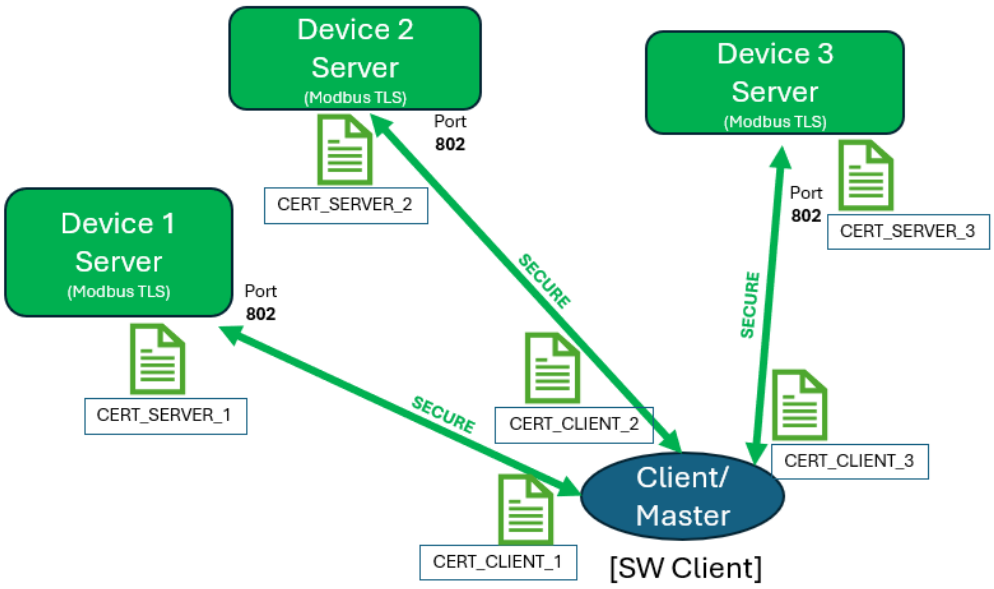
MODBUS TLS - Instructions

What is Modbus TLS

Modbus TLS represents the secure version of the Modbus TCP protocol as defined in "**MODBUS/TCP Security - Protocol Specification**" (→ see *References*). The protocol allows the secure communication with TLS Authentication and Encryption of the channel from/to the Netman and the Software Client, over the network.
Over than the security improvements, it introduces other few changes:

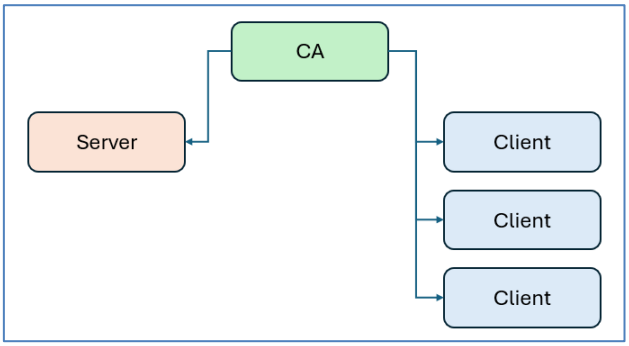
	Modbus TCP	**Modbus TLS (secure)**
Default port	502	802
Communication channel	Network packets not encrypted (unsecure)	Network packets with TLS Encryption with Certificates (secure)
Device requesting data	Master	Client
Device responding	Slave	Server
Operations allowed	Read all / Write all *(from/to anyone)*	Read / Write specific per user *(role based in the Netman/Server device, reading the RoleOID/RBAC extension in the Client certificate received)*



TLS specific implementations and ciphers for Modbus TLS

Modbus TLS envelopes the already known Modbus TCP protocol inside a secure TLS communication, based on specific certificates crafted with RoleOID extension. The RoleOID, needed only for client certificates, defines what the software Client can do or not in the Netman/Server device.

Certificates require X.509v3 (OpenSSL with x509 extensions).
The best Trust Certificates should consider:



The client certificates require the RoleOID extension mandatory (from RBAC / Role-Based Access Control) (oID: 1.3.6.1.4.1.50316.802.1).
TLS and Ciphers accepted from the Netman are:

✓ **TLS 1.3** allowed**:**
TLS_AES_128_GCM_SHA256
TLS_AES_256_GCM_SHA384

✓ **TLS 1.2** allowed:
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

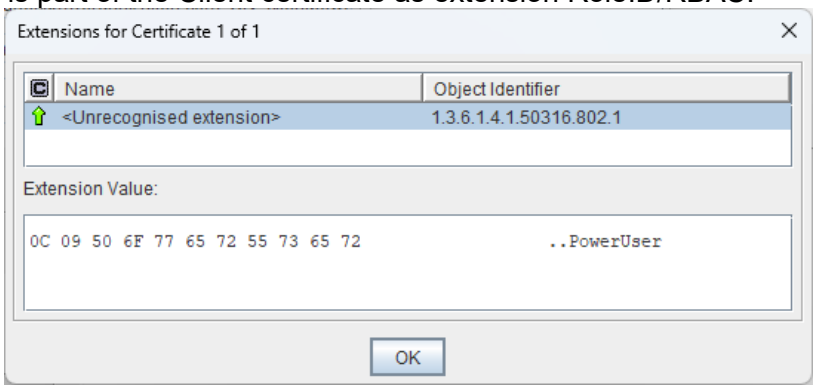
✕ **Old TLS and old Ciphers** are not allowed:
no TLS 1.1
no TLS 1.0
no ciphers related to TLS 1.1
no ciphers related to TLS 1.0

User Roles accepted

Client Roles accepted from the Netman are:

Role Oid	*Read*	*Write*
PowerUser	✓ **Read all**	✓ **Write all**
User	✓ **Read all**	✕ (no write)
any other role	✕ (no read)	✕ (no write)
Role not defined	✕ (no read)	✕ (no write)

The role definition is part of the Client certificate as extension RoleID/RBAC:



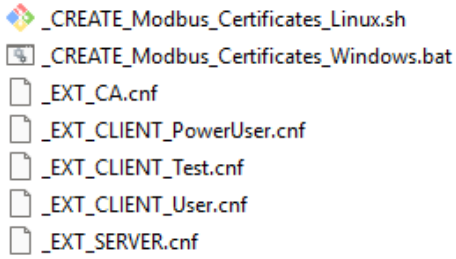
What is needed

The minimal setup for enabling Modbus TLS:

- the set of Certificates and Keys generated from the script
- a software Client with Modbus TLS support

How to create the certificates

Refer to the script "**Modbus_TLS_Certificates_Creation_Scripts**" available (for Windows and Linux):



Requirements for correct execution of the generator scripts:

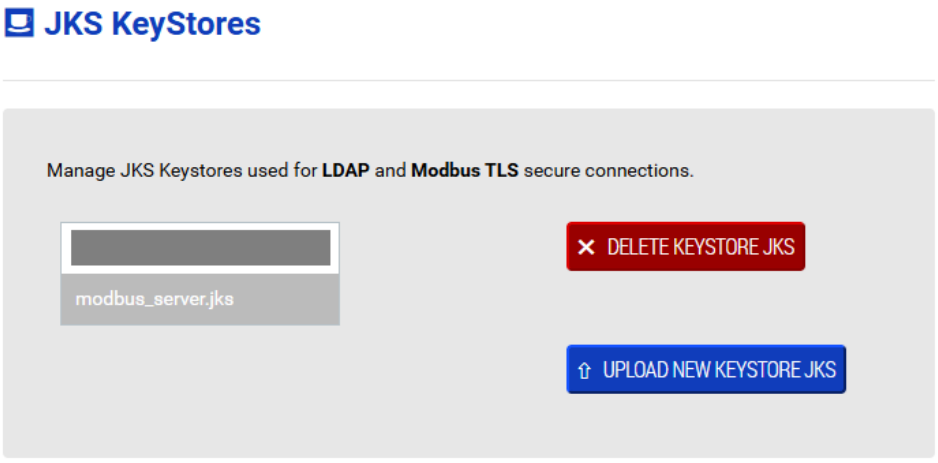
- ***Keytool*** (from Java)
- ***Openssl***

Once executed, all the certificates are available:

modbus_client_poweruser_certificate.pem modbus_client_poweruser_private_key.pem modbus_client_poweruser.jks modbus_client_poweruser.pfx modbus_client_poweruser_password_[JKS][PFX].txt modbus_client_user_certificate.pem modbus_client_user_private_key.pem modbus_client_user.jks modbus_client_user.pfx modbus_client_user_password_[JKS][PFX].txt	modbus_ca_cert.der modbus_ca_cert.pem modbus_ca_key.pem modbus_server_certificate.pem modbus_server_private_key.pem modbus_server.jks modbus_server_password_[JKS].txt

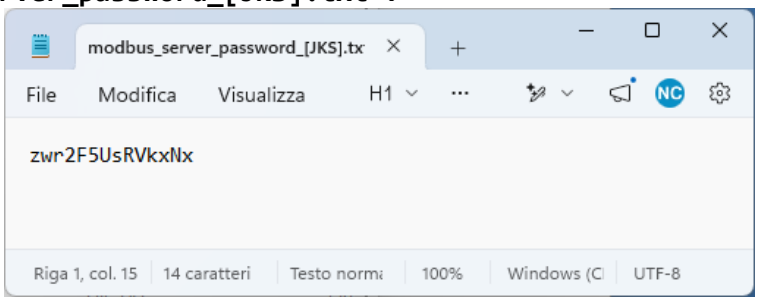
How to configure the Netman 208C

Upload the generated certificate "**modbus_server.jks**" in the WEB/Certificates section:

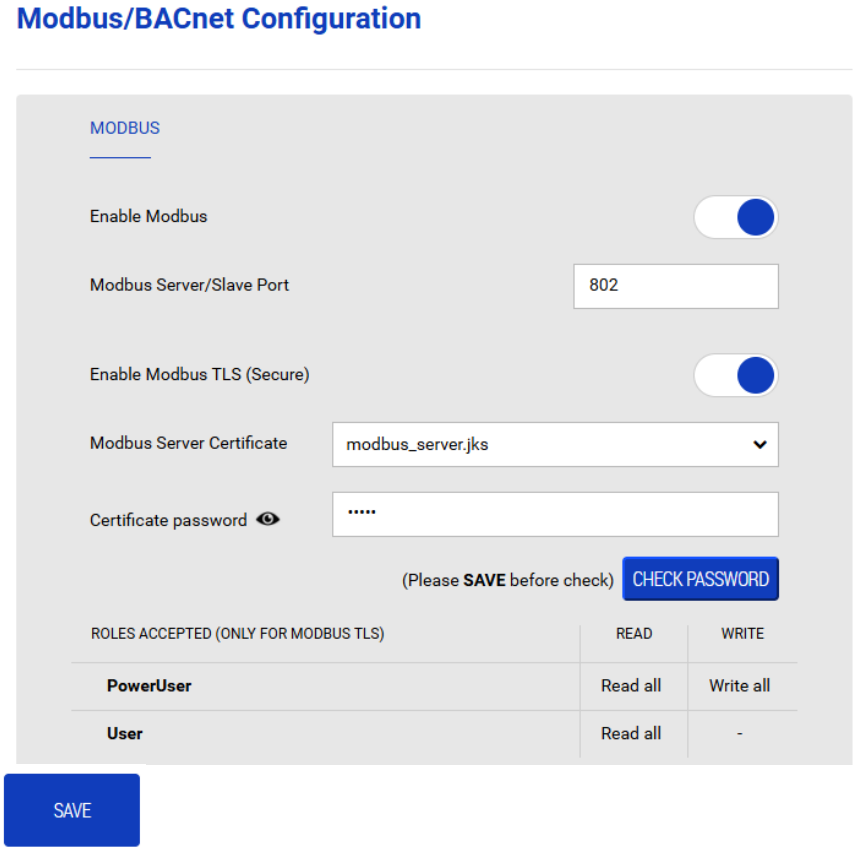


Get the password from "**modbus_server_password_[JKS].txt**" needed for JKS in the Netman:

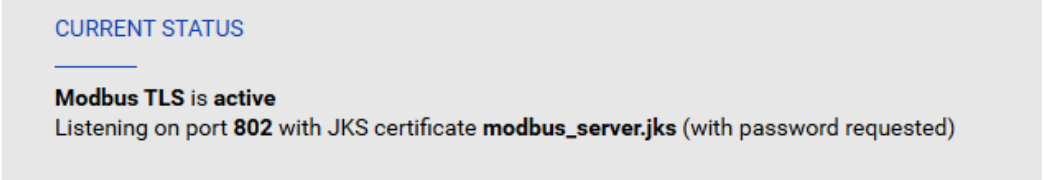
modbus_server_password_[JKS].txt →



Enable and configure Modbus TLS with correct port (802, default), select the certificate "**modbus_server.jks**" from the list, then set the password of the certificate:

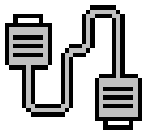


After pressing the SAVE button and a Reload of the Netman the service becomes enabled:



How to configure the SW Clients

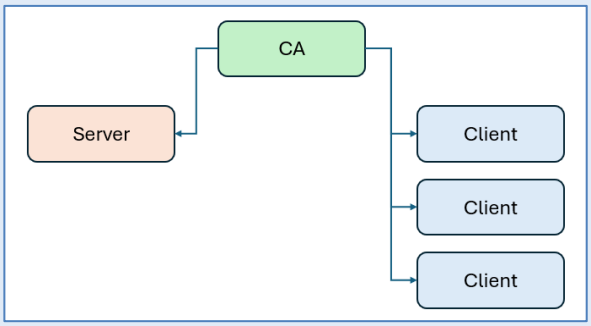
There are still few software clients available with support of Modbus TLS.
Software clients may require a variety of form of certificates, sometimes just in .PEM format, otherwise in other formats with a password.
Some examples tested:

Modbus Poll for Windows 	The client requires: - host -> Netman - port -> 802 - certificate file in PEM format -> e.g. "**modbus_client_poweruser_certificate.pem**" - key file in PEM format -> e.g. "**modbus_client_poweruser_private_key.pem**"
Modbus Client for Windows 	The client requires: - host -> Netman - port -> 802 - certificate file in PKCS#12 / PFX format -> e.g. "**modbus_client_poweruser.pfx**" - password of the certificate file -> e.g. "**modbus_client_poweruser_password_[JKS][PFX].txt**"
Custom code based on **Python + "tmodbus"** 	The given code requires: - host -> Netman - port -> 802 - certificate file in PEM format -> e.g. "**modbus_client_poweruser_certificate.pem**" - key file in PEM format -> e.g. "**modbus_client_poweruser_private_key.pem**"
Dedicated "**STunnel**" (from Windows PC to the Netman) to be used with a **standard software client for Modbus TCP** 	The "Stunnel" software requires the the configuration with: - host -> Netman - port -> 802 - certificate file in PEM format -> e.g. "**modbus_client_poweruser_certificate.pem**" - key file in PEM format -> e.g. "**modbus_client_poweruser_private_key.pem**" - CA file in PEM format -> e.g. "**modbus_ca_cert.pem**"

Compatibility for TLS 1.2/TLS 1.3 in Windows

Some software Clients for Windows may require the Security Support Provider (SSP) "Schannel" layer provided by Windows itself. The Modbus TLS protocol supported is limited to TLS 1.2/1.3 and only some Windows versions can support them:

- **TLS 1.3**: available from Windows 11 and Windows Server 2022 and later
- **TLS 1.2**: available from Windows 8 and Windows Server 2012 and later
- older Windows versions may not support TLS 1.2/1.3 as needed or at all

Certificates and Keys specifications – Modbus TLS	
Certificate type:	X.509v3 (OpenSSL with x509 extensions needed)
Certificate chain:	**Server** and **Client** certificates signed from same **CA**) 
Extensions:	**Client** certificates with RoleOID extension mandatory (from RBAC / Role-Based Access Control) oID: 1.3.6.1.4.1.50316.802.1
TLS Versions	**TLS 1.3 recommended** *(available from Windows 11 and Windows Server 2022)* **TLS 1.2 minimum** (*available from Windows 8 and Windows Server 2012*) Older TLS versions are not allowed
Authentication:	Mutual Authentication (Client and Server with respect to CA)
Cipher suites:	**TLS 1.3:** TLS_AES_128_GCM_SHA256 TLS_AES_256_GCM_SHA384 **TLS 1.2:** TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 **Older cipher suites not allowed:** - all ciphers related to TLS 1.1 - all ciphers related to TLS 1.0

References

MODBUS/TCP Security - Protocol Specification (from Modbus Organization)
https://www.modbus.org/file/secure/modbussecurityprotocol.pdf