Product Development

SOA Gateway
Deployment and User Guide

Version 1.0

Version Date: June 11, 2019

UNCONTROLLED COPY IF PRINTED

FORD CONFIDENTIAL

Table of Contents

1 Purpose

This document provides a brief description of the overall SOA Gateway architecture and describes how it is deployed on a module and setup for use.

2 Introduction

The SOA Gateway is a component in the SOA middleware that establishes a secure SOA connection between an ECU and the SOA broker on the ECG and brokers SOA communication between the ECU and the ECG. The process executes as a daemon and can be supported on any Posix compliant OS. The connection to the ECG is established over a single MQTT connection between the SOA Gateway and the SOA broker, over TLS. Once the connection is established, a SOA client application on the ECU uses the SOA API to communicate over SOA.

3 SOA Gateway Functions

SOA clients on the host ECU register with the SOA gateway and the connect to the SOA system. Once connected, clients can send and receive messages.

3.1 Communication with the ECG

The SOA Gateway establishes a secure MQTT connection to the ECG. This connection uses Transport Layer Security (TLS). The gateway will need access to the SOA certificates and public/private keys on the host ECU to establish this connection.

If the connection is broken, the Gateway will attempt to re-establish the connection.

3.1.1 TLS session

The SOA Gateway will use Transport Layer Security and public/private key pairs when establishing a SOA connection to the ECG. If the host ECU has a valid Ford signed security certificate, it will be authorized to connect to the ECG SOA broker.The SOA Gateway needs access to SOA security keys and certificates that are needed to establish a secure connection to the ECG. These files need to be provided by the host ECU

3.2 Communication with SOA Clients

SOA clients on the host ECU communicate with the SOA gateway using the SOA framework API. The Gateway communicates status information back to the clients via a status listener. Clients receive notifications on connection establishment, connection loss etc.

The SOA Gateway uses peer credentials to identify and authorize SOA clients to connect to it. Each client connecting to the Gateway is identified by its UID.

3.3 Service registration and ACL

SOA components on the host ECU that provide services on the FNV2 network must be registered with the SOA Service Manager. Also, SOA clients on the host ECU must be granted permission to publish and subscribe on SOA endpoints before any communication over SOA can occur. This is achieved via SOA service descriptors that SOA clients on the ECU must provide. These service descriptors are XML files that must be encrypted and stored in a location accessible to the SOA Gateway. As part of it's initialization sequence, the SOA Gateway will, sends service registration requests to the SOA Service Manager to configure the required read/write permissions for the SOA service providers and SOA service consumers on the host ECU.

IDL file: https://github.ford.com/FNV/idl/blob/soa-develop/service-manager/EcuStatusPayload.proto

Endpoint: SERVICES/REQUEST/FNV/SOA/SERVICEMANAGER/SERVICELIFECYCLE

3.4 Module swap

The SOA Gateway also extracts and logs the FESN of the module and sends the information to the ECG (FNV-VNM).

Endpoint: SERVICES/DATA/GATEWAY/ESN

3.5 Module connection

The SOA Gateway publishes connection state information on the SOA bus. The information sent is described in the protobuf below.

IDL file: https://github.ford.com/FNV/idl/blob/master/soa/gateway/SoaGatewayConnectionState.proto

Endpoint: SERVICES/DATA/GATEWAY/CONNECTIONSTATE

4 SOA Gateway Integration on a Host ECU

The SOA Gateway deployment package consists of the following :

Directory	Contents	Size on disk	Notes
include	headers files		Optional. Only needed for source level access to SOA APIs
lib	shared libraries	~40 Mb	Dependent libraries for the soagateway process
bin	executables	~725 Kb	
etc	configuration files	350b	

4.1 Deploying the SOA Gateway

In addition to including the contents of the SOA Gateway deployment package as part of the host ECU's firmware image, the following are required :

4.2 Unix domain socket directory

The gateway uses Unix Domain Sockets to communicate with SOA clients on the ECU. The default location of the socket files is **/var/run/soa**.

This path can be configured using the “`ipcRootPath`” key in the Gateway configuration file (see section 'configuration file' below).

The SOA gateway and SOA clients will require permissions to create files in this directory.

4.2.1 Access Control for SOA Client / SOA Gateway communication

A supplementary group is used to control access to the SOA Gateway sockets.

1. The host ECU must define a unix user to control access to the sockets. It is recommended to use "soagateway"
2. The socket directory must have it's owner and group ids set to this user.
3. The socket directory must remove rwx permissions for 'other' and only have them set for 'owner' and 'group'.
4. The socket directory should have it's setguid bit set so that files created within it will inherit the 'soagateway' group id.
5. The soagateway process' owner and group credentials should be set to 'soagateway' or whatever user id is chosen in no. 1 above.
6. All SOA clients on the host ECU should belong to the 'soagateway' supplementary group.

4.2.2 'soa' directory showing setguid bit set and owner and group credentials

```
drwxrws---    2 soagateway soagateway       280 Jan  1 03:37 soa
```

4.3 soagateway user and group

As mentioned above, a 'soagateway' user and group id has to be defined on the host ECU.

Clients can be added to the soagateway supplementary group by adding an entry in the /etc/groups file similar to the line below :

soagateway:x:65:soagateway,<soagateway_client#1>,<soagateway_client#2>

If SLM us used on QNX, the group has be to defined in the SLM configuration as it will override what is defined in the /etc/group.

4.4 Configuration file

The SOA Gateway configuration file must be stored in a read-only file system to prevent it from being modified. See Gateway configuration file for more details.

If the host ECU supports debug tokens, the SOA gateway can read an alternate configuration file from a writeable partition as well. This is useful for changing the default configuration during development and testing.

4.5 ECU software version

ECU software versions are used to let ECUs register service descriptors to SOA service manager, by using which SOA service manager decides if SOA service registrations are needed. Currently TCU and SYNC software versions are automatically acquired by SOA gateway from /etc/Version.inf for TCU and /artifact.json for SYNC. For those ECUs, software versions are not publicly available shall provide APIs to SOA gateway.

4.6 Device type access

The SOA gateway will require an API from the host ECU to determine the device security type (secure/insecure, development/production). Note that this functionality is only needed if the host ECU would like to configure the SOA gateway to be configured to connecting without using TLS, in some cases as described in the "Security Certificates and Keys" section below.

4.7 Security certificates and keys

The SOA Gateway can connect to the ECG with or without TLS. For TLS connections it requires access to the SOA security keys and certificates on the host ECU. These will have to reside in secure storage and the ECU will have to provide APIs for the access to these files.

Typically production secure hardware will use a TLS connection to the ECG. If the ECU has debug token support, this behaviour could be modified by loading a debug token.

The following table summarizes the connection type possibilities and the corresponding certificate and key pairs used. This behaviour is configurable on the host ECU.

Device type	Gateway connection to ECG	Keys & CA cert
Insecure	with / without TLS	developer
Dev secure	TLS	developer
Dev secure + debug token	with / without TLS	developer
Prod secure	TLS	production
Prod secure + debug token	with / without TLS	production

Please check TLS connection Documentation for further details on how the connection type can be configured.

Secure devices will always use a TLS connection. If the host ECU has support for debug tokens, the SOA gateway can support an alternate configuration based on the presence of a debug token as described in the "Configuration file" section above. In this case the connection type can be modified in the Gateway configuration file.

4.8 Network stack

The SOA gateway must have the required permissions to create and use sockets. On certain systems this may include adding it to the 'inet' supplementary group.

4.9 Launching and restarting

The host ECU must provide a mechanism to launch the SOA gateway and restart it in instances where it terminates abnormally.

The SOA Gateway can be started using the following command:

```
soagateway --config <path_to_configuration_file>.json
```

It is recommended to start the Gateway early in the boot sequence, usually from a startup script.

4.10 TLS keys retrieval

The soa gateway, if configured to connect over TLS, needs to get access to private and public keys as well as CA certificate. These should be provided as files in the PEM format (Base64 encoded ASCII files).

Vendors integrating the SOA gateway must provide access to these keys and certificates through corresponding APIs to retrieve the 6 files (2 keys and the certificate for development and production devices).

```
// Return fully qualified path to the required key or certificate file.
// Return an empty string if not supported.
std::string vendor_API_get_Prod_CA_Cert_Path();
std::string vendor_API_get_Prod_Device_Cert_Path();
std::string vendor_API_get_Prod_Device_Key_Path();

// Return fully qualified path to the required key or certificate file.
// Return an empty string if not supported.
std::string vendor_API_get_Dev_CA_Cert_Path();
std::string vendor_API_get_Dev_Device_Cert_Path();
std::string vendor_API_get_Dev_Device_Key_Path();
```

As both production and development signed hardware could be available, two sets of methods are defined above, one for development and one for production. If an ECU only supports production hardware, the corresponding "Dev" functions can simply return an empty string.

4.11 Dependency with fnv components

The SOA gateway has dependencies with fnv components which are abstracted out for ECUs not including those components. An API (adapter) is provided to be able to implement the missing features for non FNV ECUs.

- Keymgr (mandatory)

If a TLS connection is setup, the gateway will need to get access to the public/private keys and certificate.

- Tokenmgr (mandatory)

TLS connection with the Gateway can only be disabled if the device is insecure or a debug token is found.

- Heart Beat (Optional)

This mechanism is in charge of receiving periodic heart beats sent from the gateway. If the heart beats are not received after a determined timeout, the mechanism will restart the Gateway

- Logging/Telemetry (optional)

In charge of logging from the Gateway and SOA framework API. This is recommended as the is the main source of information for debug purpose.

- Analytics/Diagnostics (optional)

Sends event to the Ford cloud. The gateway sends some events in case of critical errors as failure to connect to the ECG, crash, etc.

4.12 Registration of SOA service producers & consumers

The SOA gateway also registers all SOA components on the host ECU with the SOA middleware to provide the clients access to services declared in their SOA descriptors.

Gateway will publish on <ecu-id_gateway>/SERVICES/REQUEST/FNV/SOA/SERVICEMANAGER/DMZ]

5 Appendix A

5.1 Runtime dependencies:

Library	Type	Size, kB	Comment
libfnvthread.so	FNV	791	
libservicemanageridl.so	FNV	3109	Can be reduced, does not need full library
libsoaframework.so	FNV	5755	
libsoaidl.so	FNV	1798	
libipclite.so	FNV	201	
libkeyclient.so	FNV	51	Will be removed for host ECUs without FNV2 KeyMgr
libfnvdaemonprocess.so	FNV	66	
libtelemetry.so	FNV	262	Needed for logging and telemetry. Can be removed.
libtelemetryidl.so	FNV	279	Needed for logging and telemetry. Can be removed.
libbmetrics.so	FNV	12	Will be removed.
libanalytics.so	FNV	156	Can be removed for host ECUs without FNV2 Analytics/Diagnostics
libanalyticsidl.so	FNV	922	Can be removed for host ECUs without FNV2 Analytics/Diagnostics
libosssmidl.so	FNV	111	Can be removed for host ECUs without FNV2 Stability monitor
libfdtokenclient.so	FNV	84	Will be removed for host ECUs without FNV2 TokenMgr
libRCN_DiagnosticsLib.so	FNV	156	Can be removed for host ECUs without FNV2 Analytics/Diagnostics
libpaho-mqtt3as.so.1	External	637	
libpaho-mqtt3cs.so.1	External	606	
libprotobuf.so.13	External	6716	Google protobuf library.
libssl.so.2	SDK	509	

Total size on disk ~22 Mb.