Tizen Security Services

API Specification

Document version 1.0.0

Document Information

Document Details

Revision	1.0.0
Author	MMS Development Team

Revision Information

Revision	Revision Date	Author	Details
1.0.0	06/25/2014	MMS Development Team	Created

Contents

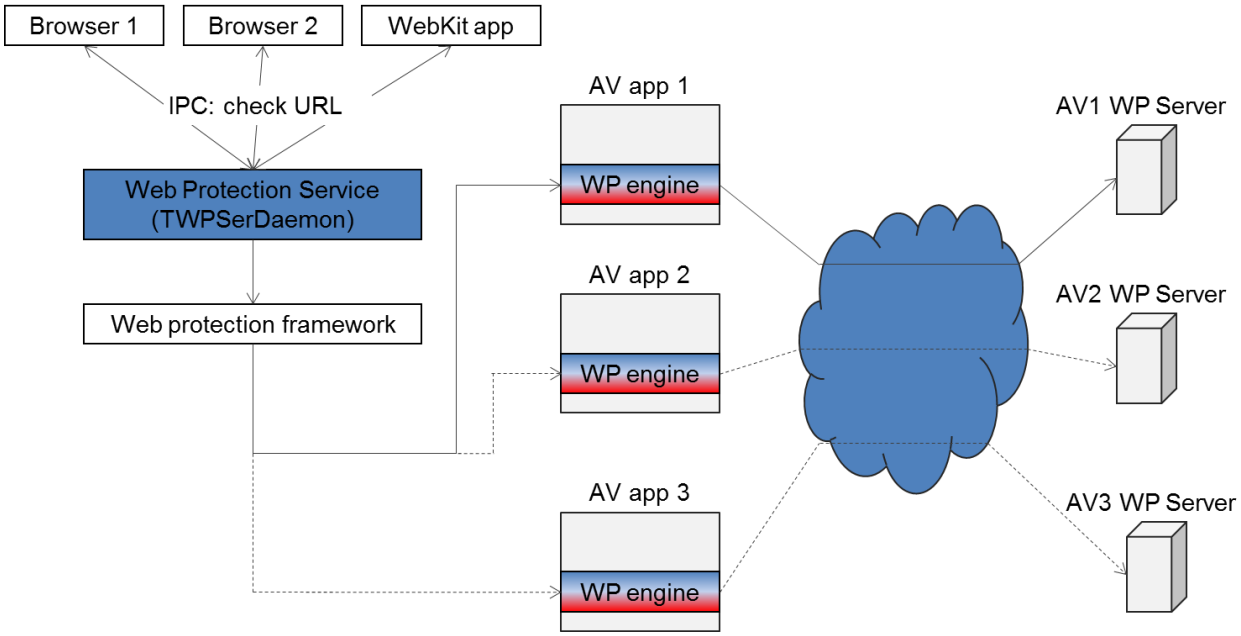
Terms, Abbreviations, Definitions, Conventions

Items	Description
SDK	Software Development Kit
API	Application Programming Interface
TWPSerDeamon	Web protection service Daemon
Module	Program, service or any execution entity in the Tizen platform
Application	Executable provided by either system or third-party
Application ID	Application ID generated by Tizen, eg. Z646N19o1u. It also identifies unique part of directory path where the application is installed.

Tizen Security Services Overview

Web Protection Service

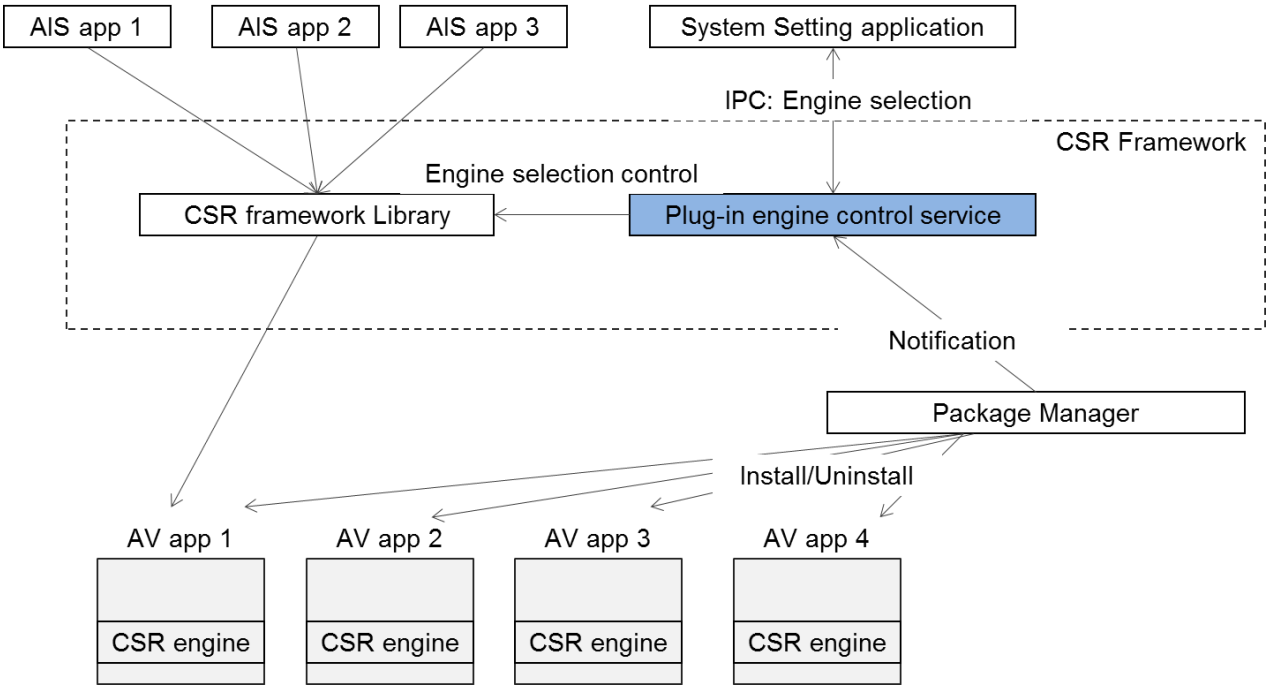
Web Protection service provides the risk level of URL.



The delivery of the Daemon will be a binary file: `TWPSerDaemon`.

Plug-in Engine Service

Plugin control service provides services such as install uninstall, set active plugin and plugin info.



The delivery of the Daemon will be a binary file: `TPCSSerDaemon`

IPC Client Library

The delivery of IPC client library will be .so library: `libscclient.so`

Initialize Functions

Summary

Methods	
`IpcClientInfo*`	`IpcClientOpen(void)`
	Initializes and returns IPC info of client.
`void`	`IpcClientClose(IpcClientInfo *pInfo)`
	Close the client-side IPC and release the resources.

Methods

```
IpcClientInfo* IpcClientOpen(void)
```

Initialize IPC Client library. For example, allocating memory for internal data use, loading signature database, etc.

Parameters

None.

Returns

Valid Pointer	On success, An Instance to the IPC Client library.
`Null`	On failure

```
void IpcClientClose(IpcClientInfo *pInfo)
```

Close the client-side IPC and release the resources.

Parameters

`[in] pInfo`	IPC Client library instance returned by call to `IpcClientOpen()`.

Returns

None.

IpcClientInfo

Description

Data structure to manage client side connection to the dbus daemon.

Summary

Fields	
dbconn	Handles the connection to the DBus.
req_name	Name and Process ID of the DBus IPC Client.
pid	Process ID of the IPC Client.

Send Message Functions

Summary

Methods	
int	TSCSendMessageN(IpcClientInfo *pInfo, const char *service_name, const char *szMethod, int argc, char **argv, int *argc_reply, char ***argv_reply, int timeout_milliseconds);
	Sends request to security services and returns back reply.
int	TSCSendMessageAsync(IpcClientInfo *pInfo, const char *service_name, const char *szMethod, int argc, char **argv, TSC_CALL_HANDLE *pCallHandle, TSCCallback pCallback, void *pPrivate, int timeout_milliseconds);
	Sends request to security services asynchronously.

Methods

```
int TSCSendMessageN(IpcClientInfo *pInfo,
                    const char *service_name,
                    const char *szMethod,
                    int argc,
                    char **argv,
                    int *argc_reply,
                    char ***argv_reply,
                    int timeout_milliseconds);
```

Send requests to the security services and returns back the reply.

Parameters

[in] pInfo	Client side IPC info. See IpcClientInfo.
[in] service_name	Name of the service.
[in] szMethod	Name of the method called.
[in] argc	Number of parameters passed in "argv"
[in] argv	Array of strings representing parameters for method called.
[out] argc_reply	Length of the string in "argv_reply".
[out] argv_reply	Array of strings representing result value from method called.
[in] timeout_milliseconds	Timeout in milliseconds. -1 for default or 0 for no timeout.

Returns

0	Send Success.
-1	Send Failure.

```
int TSCSendMessageAsync(IpcClientInfo *pInfo,
                        const char *service_name,
                        const char *szMethod,
                        int argc,
                        char **argv,
                        TSC_CALL_HANDLE *pCallHandle,
                        TSCCallback pCallback,
                        void *pPrivate,
                        int timeout_milliseconds);
```

Parameters

[in] pInfo	Client side IPC info. See IpcClientInfo.
[in] service_name	Name of the service.
[in] szMethod	Name of the method called.
[in] argc	Number of parameters passed in "argv".
[in] argv	Array of strings representing parameters for method called.
[out] pCallHandle	Pointer to handle of the asynchronous message sent.
[in] pCallback	Callback function for the asynchronous reply.
[in] pPrivate	API caller's context information, to be supplied with callback.
[in] timeout_milliseconds	Timeout in milliseconds. -1 for default or 0 for no timeout.

Returns

0	Send Success.
-1	Send Failure.

CallBack Functions

Summary

Callback Methods	
void	(*TSCCallback)(void *pPrivate, int argc, void **argv)
	CallBack Function for Async method supported by the IPC.

Methods

```
void (*TSCCallback)(void *pPrivate, int argc, void **argv)
```

The above callback method is used to get notification when method execution completes on the server side.

Parameters

[in] pPrivate	API caller's context information, supplied with TSCSendMessageAsync() earlier.
[in] argc	Length of the string in argv.
[in] argv	Array of strings representing result value of asynchronous reply.

Returns

None.

Web Protection Service

The client application should use the send message API to talk with this service, this chapter is to define all the possible parameters accepted by this service and corresponding replies.

TWPSerGetVersion Method

Request Parameter	Value
req_argc	0
req_argv	NULL
service_name	TSC_DBUS_SERVER_WP_CHANNEL
method_name	TWPSerGetVersion

Reply Parameter	Value
req_argc	0 – on failure error occurs with sending message. 1- on failure occurs at the deamon side rep_argv[0] contains the error code. 4 – on success.
req_argv	rep_argv[0] – Status code. rep_argv[1] – Framework Version. rep_argv[2] – Plugin Version. rep_argv[3] – Daemon Version.

Code Example

```
IpcClientInfo *info = IpcClientOpen();
       // Request args.
       int req_argc;
       char *req_argv[0];
       // Response args.
       int rep_argc = 0;
       char **rep_argv = NULL;

         //synchronous API
       if (TSCSendMessageN(info, TSC_DBUS_SERVER_WP_CHANNEL, "TWPSerGetVersion",
                       req_argc, req_argv, &rep_argc, &rep_argv, -1))
       {
          printf("client_stub: send message error");
       }
         else
         {
            if (rep_argc == 4)
            printf("Status code | Framework Version | Plugin Version | Daemon version",
                rep_argv[0], rep_argv[1], rep_argv[2], rep_argv[3]);
         }

         //Asynchronous API
       TSC_CALL_HANDLE handle = NULL;
       if (TSCSendMessageAsync(info, TSC_DBUS_SERVER_WP_CHANNEL,
                            "TWPSerGetVersion", req_argc, req_argv, &handle, CB,
                            NULL, -1))
       {
         printf("client_stub: send message error");
       }

       //Callback function example
         void CB(void *pPrivate, int argc, char **argv)
         {
            if (rep_argc == 4)
            printf("Status code | Framework Version | Plugin Version | Daemon version",
                rep_argv[0], rep_argv[1], rep_argv[2], rep_argv[3]);
         }
```

TWPSerGetURLReputation Method

Request Parameter	Value
req_argc	1
req_argv	URL e.g. www.twptest.com
service_name	TSC_DBUS_SERVER_WP_CHANNEL
method_name	TWPSerGetURLReputation

Reply Parameter	Value
req_argc	0 – on failure with sending message. 1 – on failure at the deamon side rep_argv[0] contains the error code. 2 – on success. 3 – on success rep_argv[2] contains the redirect URL.
req_argv	rep_argv[0] – Status code. rep_argv[1] – Risk Level. rep_argv[2] – Redirect URL.

Code Example

```
IpcClientInfo *info = IpcClientOpen();
// Request args.
int req_argc = 1;
char *req_argv[1] = {};
req_argv[0] = "http://www.zcrack.com";
// Response args.
int rep_argc = 0;
char **rep_argv = NULL;

if (TSCSendMessageN(info, TSC_DBUS_SERVER_WP_CHANNEL,
                 "TWPSerGetURLReputation",
                  req_argc, req_argv, &rep_argc, &rep_argv, -1))
{
   printf("client_stub: send message error");
}
  else
  {
     if (rep_argc == 3)
     printf("Status code | Risk Level | Redirect URL",
         rep_argv[0], rep_argv[1], rep_argv[2]);
  }

TSC_CALL_HANDLE handle = NULL;
if (TSCSendMessageAsync(info, TSC_DBUS_SERVER_WP_CHANNEL,
                     " TWPSerGetURLReputation",
                     req_argc, req_argv, &handle, CB, NULL, -1))
{
  printf("client_stub: send message error");
}

//Callback function example
  void CB(void *pPrivate, int argc, char **argv)
  {
     if (rep_argc == 3)
     printf("Status code | Risk Level | Redirect URL",
         rep_argv[0], rep_argv[1], rep_argv[2]);
  }
```

Plug-in Control Service

The client application should use the send message API to talk with this service, this chapter is to define all the possible parameters accepted by this service and corresponding replies.

TPCSGetInfoPlugin Method

Request Parameter	Value
req_argc	0
req_argv	NULL
service_name	TSC_DBUS_SERVER_PLUGIN_CHANNEL
method_name	TPCSGetInfoPlugin

Reply Parameter	Value
req_argc	0 – on failure with sending message. 1 – on failure at the deamon side rep_argv[0] contains the error code. 2 – on success.
req_argv	rep_argv[0] – Status code "0" on success, "1" on failure rep_argv[1] – config file content

Code Example

```
IpcClientInfo *info = IpcClientOpen();
// Request args.
int req_argc = 0;
char *req_argv = NULL;

// Response args.
int rep_argc = 0;
char **rep_argv = NULL;

if (TSCSendMessageN(info, TSC_DBUS_SERVER_PLUGIN_CHANNEL,
                 "TPCSGetInfoPlugin",req_argc, req_argv,
                  &rep_argc, &rep_argv, -1))
{
   printf("client_stub: send message error");
}
else
{
   if (rep_argc == 2)
   printf("Status code | Config content",
         rep_argv[0], rep_argv[1]);
}
```

TPCSInstallPlugin Method

Request Parameter	Value
req_argc	1
req_argv	Application Id
service_name	TSC_DBUS_SERVER_PLUGIN_CHANNEL
method_name	TPCSInstallPlugin

Reply Parameter	Value
req_argc	0 – on failure with sending message. 1 – on failure at the deamon side rep_argv[0] contains the error code. 2 – on success.
req_argv	rep_argv[0] – Status code "0" on success, "1" on failure rep_argv[1] – config file content

Code Example

```
IpcClientInfo *info = IpcClientOpen();
// Request args.
int req_argc = 0;
char *req_argv = NULL;

// Response args.
int rep_argc = 0;
char **rep_argv = NULL;

if (TSCSendMessageN(info, TSC_DBUS_SERVER_PLUGIN_CHANNEL,
                "TPCSInstallPlugin",req_argc,
                 req_argv, &rep_argc, &rep_argv, -1))
{
   printf("client_stub: send message error");
}
else
{
   if (rep_argc == 2)
   printf("Status code | Config content",
        rep_argv[0], rep_argv[1]);
}
```

TPCSSetActivePlugin Method

This sets the plugin active.

Request Parameter	Value
req_argc	1
req_argv	Application Id
service_name	TSC_DBUS_SERVER_PLUGIN_CHANNEL
method_name	TPCSSetActivePlugin

Reply Parameter	Value
req_argc	0 – on failure with sending message. 1 – on success/failure.
req_argv	rep_argv[0] – Status code "0" on success, "1" on failure

Code Example

```
IpcClientInfo *info = IpcClientOpen();
// Request args.
int req_argc = 0;
char *req_argv = NULL;

// Response args.
int rep_argc = 0;
char **rep_argv = NULL;

if (TSCSendMessageN(info, TSC_DBUS_SERVER_PLUGIN_CHANNEL,
                 "TPCSSetActivePlugin",req_argc,
                  req_argv, &rep_argc, &rep_argv, -1))
{
   printf("client_stub: send message error");
}
else
{
   if (rep_argc == 1)
   printf("Status code", rep_argv[0]);
}
```

TPCSUninstallPlugin Method

This uninstalls the plugin.

Request Parameter	Value
req_argc	1
req_argv	Application Id
service_name	TSC_DBUS_SERVER_PLUGIN_CHANNEL
method_name	TPCSUninstallPlugin

Reply Parameter	Value
req_argc	0 – on failure with sending message. 1 – on success/failure.
req_argv	rep_argv[0] – Status code "0" on success, "1" on failure

Code Example

```
IpcClientInfo *info = IpcClientOpen();
// Request args.
int req_argc = 0;
char *req_argv = NULL;

// Response args.
int rep_argc = 0;
char **rep_argv = NULL;

if (TSCSendMessageN(info, TSC_DBUS_SERVER_PLUGIN_CHANNEL,
                 "TPCSUninstallPlugin",req_argc, req_argv,
                  &rep_argc, &rep_argv, -1))
{
   printf("client_stub: send message error");
}
else
{
   if (rep_argc == 1)
   printf("Status code", rep_argv[0]);
}
```