**Отчет по ДДЗ «Эхо-сервер Kerberos (SSPI)**

*Теоретические сведения*

Security Support Provider Interface (SSPI) — программный интерфейс в Microsoft Windows между приложениями и провайдерами безопасности. SSPI используется для отделения протоколов уровня приложения от деталей реализации сетевых протоколов безопасности и обеспечивает уровень абстракции для поддержки множества механизмов аутентификации.

SSPI делает доступным для приложений динамически подключаемые библиотеки провайдеров поддержки безопасности (SSP), которые содержат одну или несколько схем аутентификации и криптографии, называемых пакетами безопасности. Каждый пакет безопасности отображает функции SSPI на фактические функции протокола безопасности. Функции SSPI делятся на следующие функциональные группы:

* Функции управления пакетами — перечисляют установленные пакеты безопасности SSP и запрашивают их атрибуты.
* Функции управления мандатами (пароли, билеты kerberos и т.д.).
* Функции управления контекстом безопасности.
* Функции поддержки сообщений — позволяют передавать шифрованные и подписанные сообщения.

## Программный код

Проект разбит на 5 частей:

* NSspi
* NsspiDemo
* TestProtocol
* TestClient
* TestServer

NSspi

Решение содержит DLL-сборку, которая предоставляет интерфейс C# / .Net для интегрированного API проверки подлинности Windows. Это позволяет пользовательской клиентской / серверной системе аутентифицировать пользователей, используя их существующие учетные данные для входа. Что позволяет разработчику обеспечить единый вход в своем приложении.

API предоставляет необработанный доступ к токенам аутентификации, так что аутентификация может быть легко интегрирована в любую сетевую систему – можно отправить токены через сокет, интерфейс удаленного взаимодействия или последовательный порт в виде байтов. Клиенты и серверы могут обмениваться зашифрованными и подписанными сообщениями.

Основное взаимодействие между клиентом и сервером осуществляется, благодаря классам, унаследованным от класса Credential, в которых задаются параметры для соединения с сервером, либо клиентом и генерируется токен для взаимодейстсвия. Credential взаимодействует с классом Context, содержащим начальные настройки конфигурации, настраивающие правила обмена между клиентом и сервером сообщениями, данный класс написан как обертка над API.

Возможные варианты настроек расположены в файле ContextAttrib.cs.

Для связи буферы с API используется класс SecureBuffer SecureBuffer, который помогает взаимодействовать с буферами для предоставления токенов и данных встроенному API SSPI.

*Описание взаимодействия*

Клиент получает некоторую форму учетных данных либо из контекста безопасности вошедшего в систему пользователя, либо путем получения имени пользователя и пароля. Аналогичным образом сервер получает учетные данные. Каждый из них использует учетные данные, чтобы идентифицировать себя друг с другом.

Клиент и сервер начинаются с неинициализированных контекстов безопасности. Они обмениваются маркерами согласования и проверки подлинности для выполнения проверки, и если это удается, они создают общий контекст безопасности в виде контекста клиента и контекста сервера. Каждый новый клиент, который проходит проверку подлинности на сервере, создает новый контекст безопасности, специфичный для этого сопряжения клиент-сервер.

Сервер инициализирует себя путем обмена маркерами проверки подлинности с клиентом.

NsspiDemo

В данном проекте приведены тестовые примеры для локального использования данного API. Клиент и сервер создаются на одной машине в рамках одного процесса, после чего каждым из них создается собственный контекст безопасности, происходит обмен маркерами, после чего клиентом шифруется сообщение и передается серверу, который дешифрует его. Проверяется корректность дешифрованного сообщения.

После чего сообщение подписывается клиентом и передается серверу, который проверяет подпись и читает сообщение.

TestProtocol

Данный проект является общей зависимостью для проектов TestClient и TestServer и содержит класс для описания передаваемых сообщений Message. А также классы, описывающие отправку сообщений клиентом CustomConnection и класс CustomServer, содержащий логику работы сервера по приему-отправке сообщений.

TestClient

Данный WinForms проект является графическим представлением клиентского приложения для подключения к серверу и отправки шифрованных, либо подписанных сообщений с использованием SSPI Context’а из DLL NSspi.

Внешний вид приложения выглядит следующим образом (см. Рисунок 1).

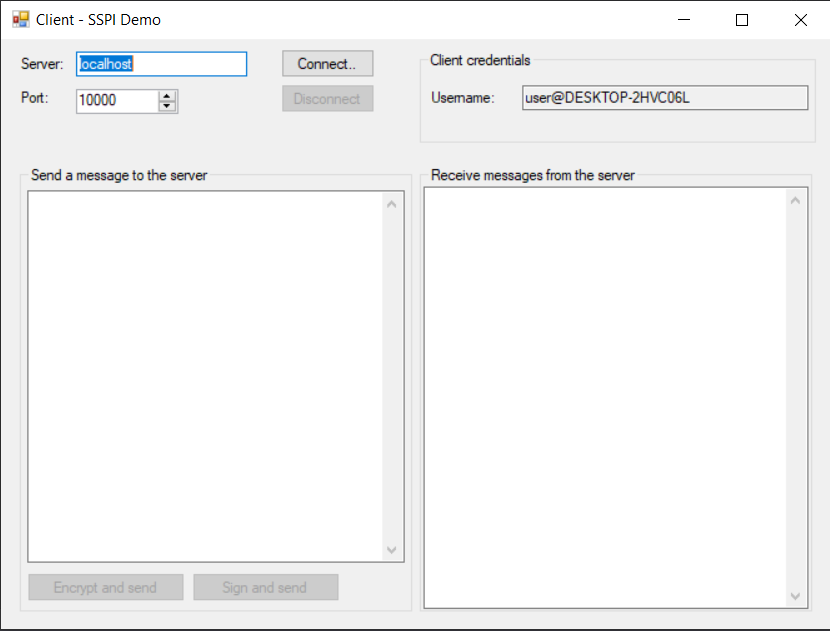


Рисунок 1 – Графическая часть клиентского предложения

Для подключения указывается адрес сервера и номер подключаемого порта. При нажатии на кнопку Connect, происходит отправка маркеров согласования контекста серверу и, если сервер одобряет соединение – оно устанавливается. После чего, клиент может отправлять на сервер либо шифрованные сообщения, либо подписанные сообщения, а сервер в свою очередь, используя контекст, может расшифровывать их (см. Рисунок 2).

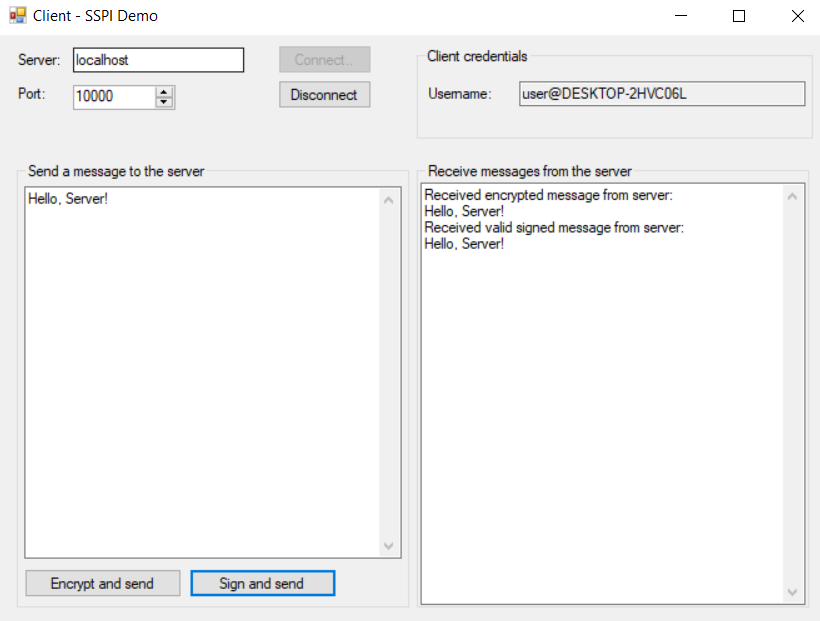


Рисунок 2 – Отправка клиентом сообщений на сервер

TestServer

Данный WinForms проект является графическим представлением серверной части приложения для подключения к нему клиентов и отправки шифрованных, либо подписанных сообщений с использованием SSPI Context’а из DLL NSspi, в ответ на принятые сообщения от клиентов.

Внешний вид приложения выглядит следующим образом (см. Рисунок 3).

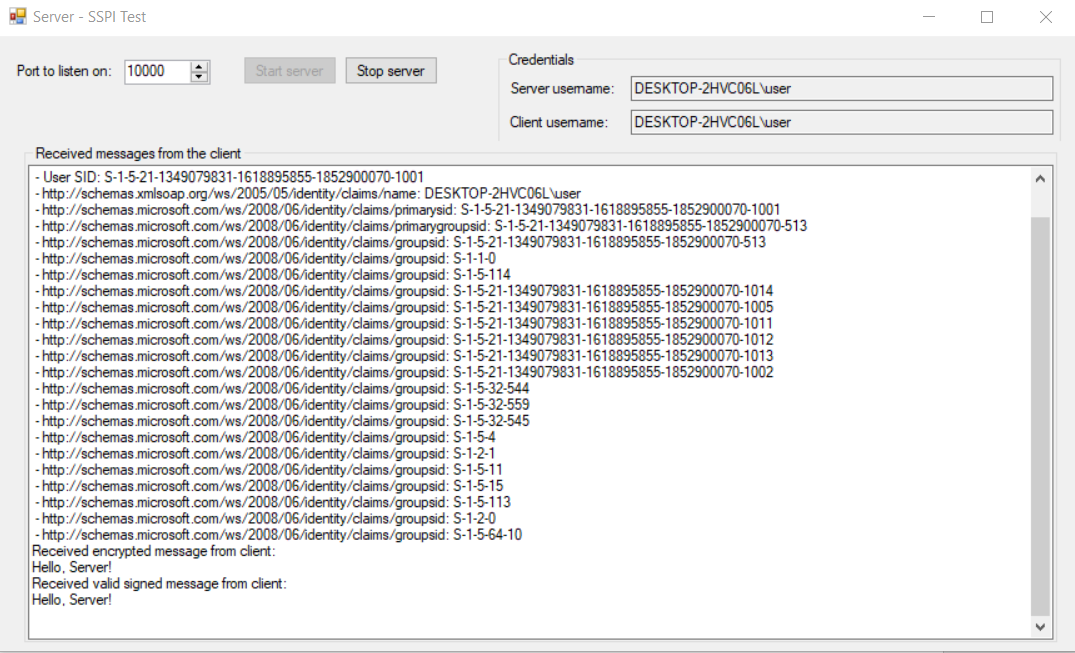


Рисунок 3 – Внешний вид серверной части приложения

При нажатии кнопки Start server, сервер запускает событие, которое ожидает входящие подключения от клиентов и в случае, если клиентский маркер согласования корректен, он подключается к серверу на передачу шифрованных, либо подписанных сообщений. В ответ на принятые сообщения сервер осуществляет эхо-ответ, пересылая клиенту его же сообщение используя тип сообщения клиента.

Вид отправляемых и принимаемых сообщений представлен на рисунках (см. Рисунок 2 и см. Рисунок 3).