# Projekt na [TIN] - Zadanie 3.
# Klient oraz symulator urządzenia protokołu EAPoL/802.1x

**Grupa: Maciej Lotz, Szymon Mysiak, Andrzej Siadkowski, Patryk Wąsowski**

## 1. Środowisko

Program zostanie napisany w języku C++ z wykorzystaniem bibliotek standardowych:

- stdlib
- string
- iostream
- windows.h
- types.h
- wincrypt.h
- pcap.h

Środowisko graficzne:

- SFML
    - TGUI

Na systemach rodziny Windows 7 i nowszych, ze względów bezpieczeństwa, mocno ograniczone zostało API surowych gniazd . Z tego powodu, do obsługi karty sieciowej na poziomie warstwy drugiej użyjemy biblioteki **WinPcap**.

(źródło:https://msdn.microsoft.com/en-us/library/windows/desktop/ms740548%28v=vs.85%29.aspx)

## 2. Zakładana funkcjonalność programu klienta

Program główny ma służyć do uwierzytelnienia użytkownika w sieci Ethernet, do której jest podłączony przy pomocy przełącznika. Pakiety będą przesyłane na drugiej warstwie modelu ISO/OSI - łącza danych - pomiędzy komputerem klienta a przełącznikiem. Autoryzacja odbywa się na podstawie EAP-MD5. Suma kontrolna pakietów wyliczana jest na zasadzie CRC32.

## 3. Zakładana funkcjonalność symulatora serwera

Ponadto będzie jeszcze dołączony program z symulatorem NAS i RADIUS, który będzie w dużej mierze namiastką działania urządzeń i ma je zastąpić, jeśli nie będą one dostępne. Będzie on łączył się z programem klienta przy pomocy surowych gniazd. Jego funkcjonalność ogranicza się do odbierania ramek od programu klienta oraz wysyłania żądań (o podanie loginu, hasła) i decyzji.

## 4. Wielowątkowość

Program klienta działać będzie na dwóch wątkach: GUI i sieciowym. Symulator przełącznika (oraz serwera), który nie jest głównym tematem naszego projektu, dla uproszczenia będzie korzystał tylko z jednego wątku.

## 5. Sposób instalacji oraz uruchomienia programu

Instalacja oprogramowania będzie polegała na rozpakowaniu dostarczonego archiwum z plikami, w którym będą zawarte wszystkie niezbędne do działania programu pliki oraz pliki wykonywalne do uruchomienia aplikacji klienta i symulatora serwera. Wybór wszelkich plików konfiguracyjnych będzie się odbywał w samej aplikacji za pomocą przycisków.
Ponadto wymaga będzie instalacja sterownika WinPcap, pozwalającego wysyłać i przechwytywać pakiety.

## 6. Format logów

Zakładamy następującą strukturę: Data, godzina, adres MAC urządzenia uwierzytelniającego oraz klienta, wysyłane komunikaty. Dane te będą zapisywane do pliku tekstowego na komputerze z aplikacją kliencką.

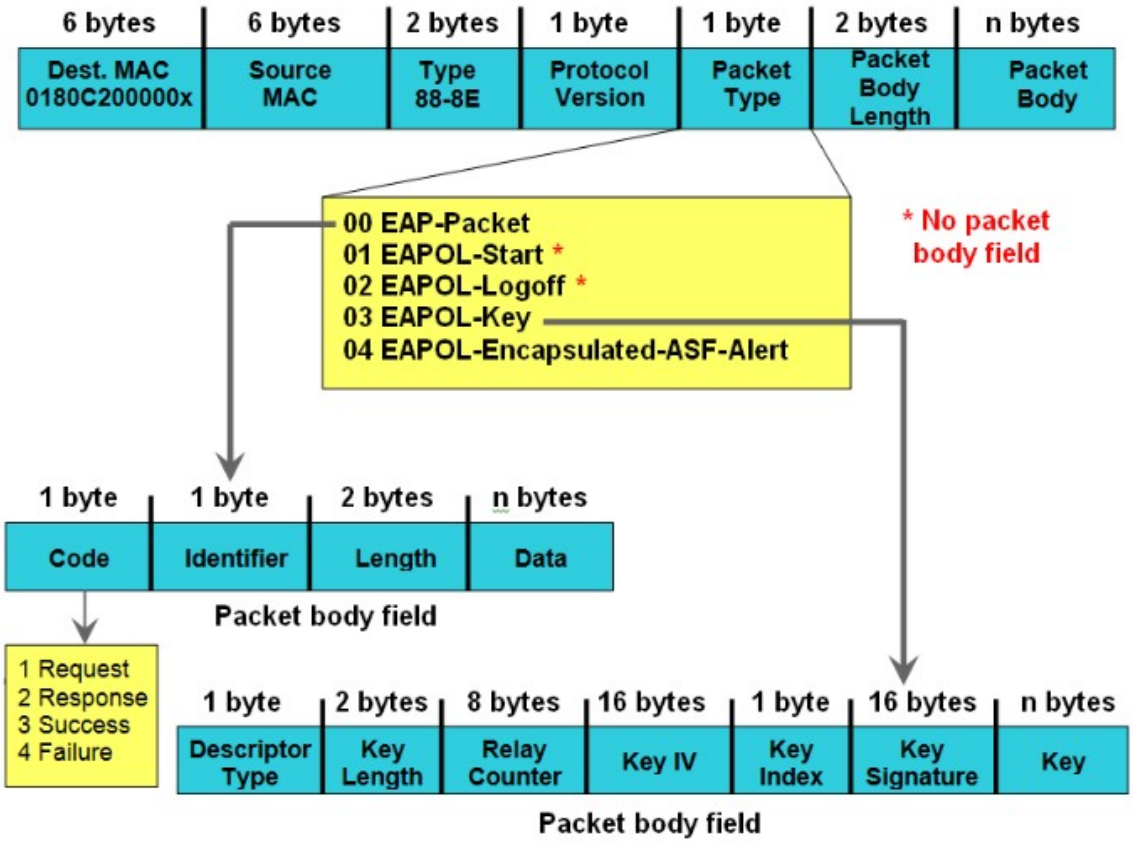
Przykładowe logi:
02-04-15 10:45:06 00:0A:E6:34:FA:E4 00:05:36:31:D1:B1 Client responses on login request: maciek
02-04-15 10:45:07 00:0A:E6:34:FA:E4 00:05:36:31:D1:B1 Server requests the password
02-04-15 10:45:12 00:0A:E6:34:FA:E4 00:05:36:31:D1:B1 Client responses on password request *(w logu nie jest zapisywane hasło, aby nie można było go później wykraść)*

## 7. Format ramek

Adres docelowy (6B) | adres źródłowy (6B) | typ Ethernetu (2B) | wersja EAPoL (1B) | rodzaj pakietu (1B) | wielkość pakietu EAPoL (2B) | dane pakietu EAPoL | suma kontrolna (4B)

Adresy docelowy oraz źródłowy są adresami MAC urządzeń interfejsu sieciowego klienta oraz przełącznika. Obie strony będą przesyłały ramki w podanym formacie.

CRC będzie sprzętowo wyliczane przez urządzenie interfejsu sieciowego(w przypadku braku wsparcia urządzenia w wyliczaniu, CRC będzie wyliczane programowo).




Schemat ramek EAPOL i EAP

(źródło: https://downloads.avaya.com/css/P8/documents/100123970)

*Ogólny format ramek zgodny z dokumentacją RFC 3748 (https://tools.ietf.org/html/rfc3748) - punkt 4. Format ramek request response - punkty 4.1, 5.*

## 8. Idea działania protokołu EAPoL

Protokół 802.1x odpowiada za uwierzytelnienie suplikanta przez serwer uwierzytelniający (najczęściej RADIUS)  za pośrednictwem urządzenia NAS. Suplikant wysyła na adres multicast ramkę EAPOL-Start i oczekuje na żądania identyfikacji przez dostępne urządzenia dostępowe. Klient wówczas wysyła swój login, który jest odebrany przez serwer RADIUS. Kolejno RADIUS wysyła żądanie hasła wraz z losowym łańcuchem. Klient dołącza do tego łańcucha swoje hasło i tworzy skrót przy pomocy algorytmu MD5. Następnie taką samą

operację przeprowadza RADIUS i porównuje skróty własny i otrzymany od klienta. Na tej podstawie serwer dopuszcza dostęp do sieci lub dalej go blokuje.

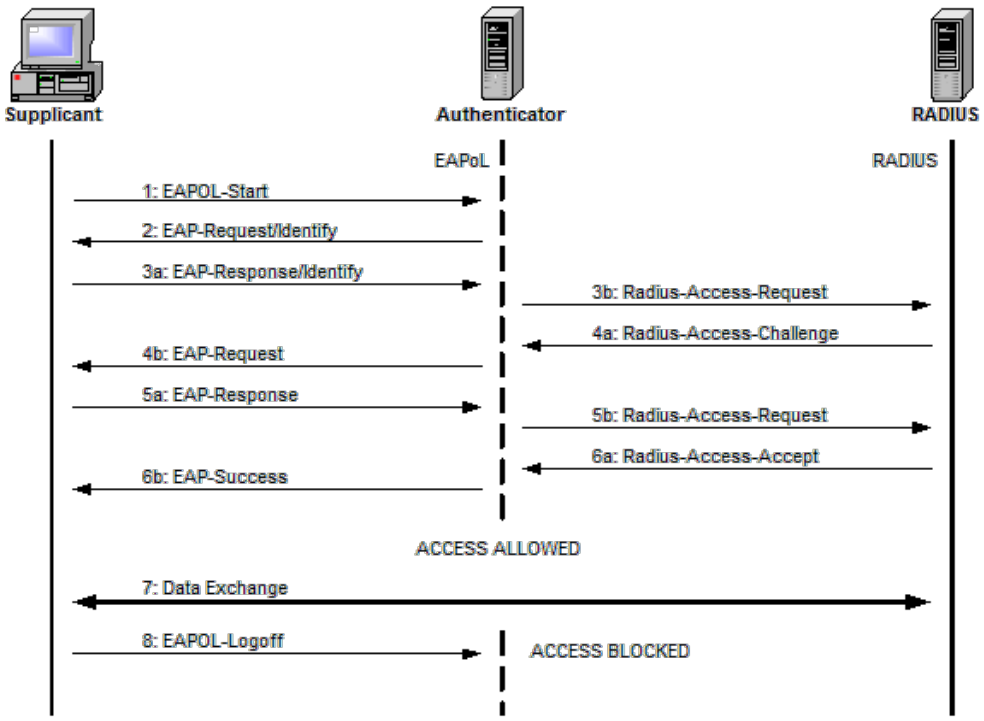



**Figure 2: Sample EAPoL Exchange**

Schemat współpracy suplikanta i przełącznika z serwerem RADIUS (źródło:http://www.vocal.com/secure-communication/eapol-extensible-authentication-protocol-over-lan/)

Nasz symulator będzie stanowił namiastkę przełącznika i serwera RADIUS.

## 9. Sposób zmiany parametrów działania programu

Zmiana parametrów wykonania programu będzie dostępna z poziomu okienka aplikacji i pozwoli na wybór odpowiedniego pliku konfiguracyjnego, podanie adresu serwera, adresu MAC przełącznika etc..

**Struktura pliku konfiguracyjnego:**

- każdy parametr musi być podany w oddzielnej linii
- linia zaczynająca się znakiem # jest komentarzem
- po nazwie parametru powinien być dwukropek, a następnie wartość parametru, niedozwolone białe znaki
- parametry będące ciągiem znaków nie mają kwalifikatora tekstu

- separator dziesiętny - ‘.’ (kropka)
- separator tysięczny - brak
- kodowanie - ANSI 1250

**Modyfikowalne parametry klienta:**

- connection_mode(0 - symulator; 1 - rzeczywiste urządzenie)
- NAS_MAC [adres MAC serwera NAS]

**Przykładowy plik konfiguracyjny dla klienta:**

*# to jest komentarz*
*connection_mode:1*

## 10. Reakcja i sposób obsługi sytuacji wyjątkowych

1. W przypadku każdej nietypowej sytuacji - umieszczenie stosownego komunikatu w logach oraz retransmisja odpowiedniego pakietu.
2. Za nietypowe sytuacje uważamy:
    - nieprawidłowa sekwencja odebrania pakietów (np. odebranie EAP-success po EAPOL-start)
    - zawartość pakietu niezgodna z sumą kontrolną
3. Serwer (symulator) 10 razy w 6-sekundowych interwałach będzie wysyłał pakiety z żądaniem odpowiedzi, jeśli do tego czasu klient nic nie napisze komunikacja zostaje zakończona (RFC 2284 2.2.1)