

TeleToken – wprowadzenie
Szybki start - podręcznik programisty

TELEVOX®

Słowem wstępu

Projektując TeleToken nie chcieliśmy narzucać ograniczeń na programistów. Dzięki temu narzędzie to jest bardzo uniwersalne, a jego wykorzystanie może różnić się od projektu do projektu. Uzyskano to dzięki wyposażeniu TeleTokenu przede wszystkim w funkcje kryptograficzne, czyli szyfrowanie oraz deszyfrowanie, ale także dodatkowo w liczniki i pamięć dostępną dla użytkownika, zarówno ulotną jak i nieulotną. Programista stykając się pierwszy raz z TeleTokenem może poczuć się zagubiony, szczególnie ilością możliwości konfiguracyjnych TeleTokenu. Ta instrukcja powstała, aby przybliżyć programiście zarówno korzystanie z samego TeleTokenu, jak i użycie go w konkretnych zastosowaniach.

O czym należy pamiętać

Generowanie liczb losowych

Do wykonywania różnych operacji kryptograficznych, a więc także przy korzystaniu z TeleTokenu, istnieje potrzeba generowania danych (bloków) losowych. Będą one wykorzystywane, np. jako „klucze sesji”, jednorazowe hasła, czy w końcu, jako same klucze szyfrowania, po uprzednim zapisaniu.

W API TeleTokenu udostępniliśmy generator pseudolosowy, jest on dostępny funkcją `TVTT_Rand`. Korzystając z niego trzeba jednak pamiętać o dwóch rzeczach. Po pierwsze zawsze przed skorzystaniem z tego generatora należy go jednorazowo w aplikacji zainicjalizować. Służy do tego opcja `TVTT_RandInit`. Drugą niezwykle istotną rzeczą jest dostarczanie zdarzeń losowych do ziarna, dzięki temu generator nie będzie przewidywalny. Służy do tego funkcja `TVTT_Randomize` (*dokładny opis w/w funkcji można znaleźć w dokumentacji TeleTokenu - „teletoken.pdf”*).

Polecamy korzystanie z generatora z TeleToken API, zamiast z rozwiązań wbudowanych w biblioteki popularnych języków programowania, ponieważ są one zbyt słabe (zbyt przewidywalne) dla naszych zastosowań.

Projektowanie zabezpieczeń

Zabezpieczenie jest tak silne, jak silne jest jego najslabsze ogniwo. Nawet jeśli użyjemy zaawansowanego kryptograficznie urządzenia, całą komunikację z nim będziemy szyfrować za pomocą uzgodnionego (wylosowanego) klucza sesji, dane będziemy szyfrować tajnym kluczem, a całe nasze bezpieczeństwo oprzemy na jednym, prostym warunku „if” to osoba atakująca w ogóle nie będzie zawracać sobie głowy łamaniem naszych kluczy. Do osiągnięcia celu wystarczy jej ominięcie/odpowiednie spreparowanie wyniku tego porównania. Dlatego też projektując zabezpieczenie warto to robić już na etapie projektowania samego systemu/aplikacji, a nie starać się doklejać je do gotowego produktu. Pomoże to na pewno w zaprojektowaniu spójnego, prostszego a zarazem bezpieczniejszego rozwiązania.

Jakkolwiek każdy system trzeba rozpatrywać indywidualnie, możemy zaproponować kilka naszych wskazówek:

- Zawsze lepszym pomysłem niż bezpośrednio porównywanie jest wpalenie otrzymanej wartości (przeszyfowanego bloku, odczytanej pamięci) w jakiś mechanizm aplikacji/systemu.
- Nie opierajmy zabezpieczenia na odczytywaniu wartości zapisanych wprost w pamięci wew. TeleTokenu. Pamięć ta powinna być pomocą, nie podłożem zabezpieczenia. Jeśli składujemy w niej istotne dane, przeszyfrujmy je kluczem z TeleTokenu, generowanym dla klienta.
- Jeśli zabezpieczamy aplikację nie sprawdzamy obecności TeleTokenu tylko w jednym miejscu. Odwołujmy się do niego cyklicznie, nie pozwalając na korzystanie z aplikacji po jego odłączeniu.
- Kluczy użytych do komunikacji nie używajmy do szyfrowania/desyfrowania.
- Przemyślmy konfigurację TeleTokenu. Klucze, liczniki powinny mieć włączone tylko te właściwości (bity konfiguracyjne), które są im niezbędne.

Konfigurowanie TeleTokenu

Na skonfigurowanie TeleTokenu pod kątem wykorzystania w swoim projekcie trzeba poświęcić chwilę czasu. Dobra, przemyślana konfiguracja jest istotna dla późniejszej prawidłowej pracy urządzenia i całego środowiska, w którym będzie wykorzystywany.

Do konfiguracji TeleTokenu można posłużyć się funkcjami API, lub edytorem dostarczonym w pakiecie programistycznym (TeleTokenEdit).

Każdy TeleToken (API v.2) wyposażony jest w możliwość zapisania do niego 8 kluczy 128-bitowych, które mogą być wykorzystane bezpośrednio przez TeleToken do szyfrowania bądź deszyfrowania, lub do procesu komunikacji. W dalszej części rozdzieliliśmy te dwie funkcje od siebie, co wydaje się nam dobrym pomysłem, także do zastosowania w konkretnych rozwiązaniach.

Kluczy zapisanych do TeleTokenu nie da się z niego odczytać. Jest to jedno z założeń, które uwzględniliśmy przy projektowaniu TeleTokenu.

Klucze komunikacyjne

Aby wykonać jakąkolwiek operację za pomocą TeleTokenu należy rozpocząć sesję. W ramach danej sesji można wykonywać różne operacje, należy jednak pamiętać, że sesja może trwać maksymalnie 5 sek., po upływie tego czasu jest automatycznie kończona przez TeleToken. W ramach otwarcia sesji tworzony jest tzw. „klucz sesji”, czyli hasło wykorzystywane w obrębie danej sesji do komunikowania się z TeleTokenem (szyfrowania i deszyfrowania komunikacji). Do stworzenia „klucza sesji” używana jest para kluczy komunikacyjnych, tzw. „klucz aplikacji” (nazywany dalej kluczem PC) oraz „klucz tokenu”.

Aby wykorzystać dany klucz jako „klucz tokenu” należy ustawić mu bit TVTTPC0_TOKEN. Od tego momentu klucz ten może zostać zaproponowany przez TeleToken do komunikacji (czyli zwrócony numer klucza w TeleTokenie) w procesie uzgadniania klucza sesji i musimy być na to przygotowani. Innymi słowy aplikacja komunikująca się z TeleTokenem musi przechowywać także wszystkie te klucze. Dobrym pomysłem jest przeznaczenie jednego z kluczy tylko i wyłącznie na cel „klucza tokenu”, czyli ustawienie mu tylko i wyłącznie bitu TVTTPC0_TOKEN.

Konfigurowanie klucza pod kątem użycia jako „klucz aplikacji” jest nieco bardziej skomplikowane. Musimy tutaj uwzględnić operacje, które będą mogły być przeprowadzane w sesji otwartej przy użyciu tego konkretnego klucza. Nie musi oznaczać to, że klucz ten posłuży do wykonania tej operacji, może on służyć tylko do otwarcia sesji (i ze względów bezpieczeństwa powinien). Do wyboru mamy następujące opcje (wybrane bity konfiguracyjne bajta konfiguracji „0”):

- TVTTPC0_PASS_WRITE – możliwość zapisu hasła (klucza) do TeleTokenu
- TVTTPC0_PASS_CONFIG – możliwość zapisu konfiguracji haseł (kluczy) TeleTokenu
- TVTTPC0_MEM_READ – możliwość odczytu pamięci TeleTokenu
- TVTTPC0_MEM_WRITE – możliwość zapisu pamięci TeleTokenu
- TVTTPC0_ENCRYPT – możliwość użycia szyfrowania TeleTokenem

- TVTTPC0_DECRYPT – możliwość użycia deszyfrowania TeleTokenem
- TVTTPC0_DECRYPT_WRITE – możliwość użycia zapisu pamięci TeleTokenu z uprzednim odszyfrowaniem danych do zapisu. Opcja dostępna od wersji „2” API TeleTokenu i wersji „1.03” biblioteki.

Oraz dodatkowo część bitów bajta „1”:

- TVTTPC1_CNT_READ – możliwość odczytu licznika
- TVTTPC1_CNT_WRITE – możliwość zapisu licznika
- TVTTPC1_CNT_CONFIG – możliwość konfiguracji licznika

Klucze użytkowe (szyfrujące/deszyfrujące)

Mianem tych kluczy określamy klucze, które nie są skonfigurowane pod kątem udziału w procesie komunikacji, a pod kątem użycia ich bezpośrednio lub pośrednio w procesie szyfrowania/deszyfrowania TeleTokenem. W odróżnieniu od kluczy komunikacyjnych, w konfiguracji tych kluczy główną rolę odgrywa bajt konfiguracyjny „1”, a konkretnie bity:

- TVTTPC1_ENCRYPT_PASS – możliwość użycia tego hasła (klucza) do szyfrowania
- TVTTPC1_DECRYPT_PASS - możliwość użycia tego hasła (klucza) do deszyfrowania
- TVTTPC1_CNT_ENCRYPT_PASS - możliwość użycia tego hasła (klucza) do zaszyfrowania odczytanej wartości licznika

Jeszcze o kluczach

Dodatkowe bity konfiguracyjne kluczy:

- TVTTPC1_WRITE_ENABLED – wyczyszczenie tego bitu spowoduje zablokowanie możliwości nadpisania klucza, ponowny zapis będzie możliwy tylko po sformatowaniu TeleTokenu
- TVTTPC1_CONFIG_ENABLED - jeśli zapiszemy bajt konfiguracyjny ze „zgaszonym” tym bitem, utracimy możliwość konfiguracji klucza. Nie oznacza to utraty możliwości zapisu samego klucza, do tego służy opcja opisana wyżej. Odzyskanie możliwości konfiguracji jest możliwe tylko po sformatowaniu TeleTokenu.

Klucze adresowane są poprzez ich numer, zaczynając od zera.

W obecnej wersji TeleTokenu (API v.2) jest ich osiem, liczbę tą można odczytać z tabeli informacyjnej TeleTokenu, automatycznie wypełnianej po otwarciu połączenia.

Dodatkowe właściwości TeleTokenu

Poza podstawowym (i najważniejszym) wykorzystaniem TeleTokenu do szyfrowania i deszyfrowania oferuje on dodatkową funkcjonalność, pamięć wewnętrzną dostępną dla programisty, oraz liczniki uniwersalne. Nie należy polegać na nich jako na właściwych zabezpieczeniach, natomiast ich wykorzystanie może pomóc w zaprojektowaniu dobrego zabezpieczenia.

Pamięć wewnętrzna TeleTokenu

TeleToken jest wyposażony w pamięć ulotną i nieulotną dostępną dla programisty.

Pamięć ulotna w obecnej wersji urządzenia (API v.2) ma rozmiar jednego sektora (czyli 16 bajtów). Zawartość tej pamięci jest tracona przy restarcie oraz odłączeniu zasilania TeleTokenu. Natomiast podczas startu do sektora pamięci ulotnej przepisywana jest zawartość sektora pamięci nieulotnej o odpowiadającym mu numerze. Pamięć ta jest przeznaczona do chwilowego przechowywania informacji. Jest także wykorzystywana przy rozkazach zapisu do pamięci nieulotnej z uprzednim rozszyfrowaniem danych do zapisu.

Pamięć nieulotna ma w obecnym TeleTokenie (API v.2) rozmiar 16 sektorów. Sektor to 16 bajtów (czyli 128 bitów, jest to zgodne z przyjętym rozmiarem bloku szyfrowania). Pamięć adresowana jest po numerach sektorów, zaczynając od zera. Sektory pamięci mogą zostać zabezpieczone przed zapisem, jest to operacja nieodwracalna. Jediną możliwością odblokowania zapisu jest sformatowanie TeleTokenu.

Rozmiar pamięci może różnić się w zależności od wersji TeleTokenu, dlatego została udostępniona opcja jego odczytu poprzez tabelę informacyjną.

Bezpieczny zapis pamięci

Od wersji „2” API TeleTokenu i wersji „1.03” biblioteki, została wprowadzona nowa funkcja zapisu do pamięci TeleTokenu. Polega ona na zapisaniu bloku pamięci z uprzednim automatycznym odszyfrowaniem za pomocą wybranego klucza. Jeśli zapisujemy w ten sposób więcej sektorów pamięci TeleTokenu, możemy skorzystać z trybu szyfru blokowego, wspomaganego przez TeleToken.

Funkcjonalność taka pozwala np. na zaprojektowanie zdalnego zapisu do TeleTokenu, przesyłana wiadomość będzie w postaci zaszyfrowanej, a więc bezpiecznej.

Liczniki uniwersalne

TeleToken jest wyposażony w 16-bitowe liczniki, których zawartość nie ulega utraceniu przy restarcie lub wyłączeniu zasilania urządzenia. Liczniki, w zależności od ich konfiguracji, oferują możliwość:

- Zapisu – jeśli licznik ma ustawiony bit TVTTCC0_WRITE
- Odczytu - jeśli licznik ma ustawiony bit TVTTCC0_READ

- Zwiększenia wartości i odczytu - jeśli licznik ma ustawiony bit TVTTCC0_INC_READ
- Zmniejszenia wartości i odczytu - jeśli licznik ma ustawiony bit TVTTCC0_DEC_READ
- Pracy w trybie samoczynnego generowania kolejnych kodów uwierzytelniających - jeśli licznik ma ustawiony bit TVTTCC0_INC_ENCRYPT_READ
Tryb ten polega na zwiększeniu wartości licznika o jeden, pobraniu jej, zaszyfrowaniu wybranym kluczem i odesłaniu do aplikacji

Prosty licznik wykonanych operacji.

Licznik może zostać skonfigurowany w ten sposób, że przy osiągnięciu wartości „0” nie pozwoli na modyfikację zawartości, przy odliczaniu w górę bądź w dół. Odpowiada za to ustawienie bitu TVTTCC0_STOP_AT_ZERO.

W połączeniu z ustawieniem licznika na wartość początkową i używaniem funkcji zmniejsz i odczytaj, pozwala na proste pilnowanie ilości możliwych uruchomień np. aplikacji, bądź jakiejś jej funkcjonalności.

Bardziej zaawansowany automatyczny licznik wykonanych operacji.

Liczniki oferują jeszcze jedną, ciekawą funkcjonalność. Mogą pracować jako kontrolery ilości odczytów, zapisów powiązanego sektora pamięci i/lub ilości operacji szyfrowania, deszyfrowania (oprócz użycia przy komunikacji) powiązanego klucza (hasła). Do konfiguracji tej funkcjonalności służą bity:

- TVTTCC0_NVMEM_READ – kontrolowanie ilości odczytów powiązanego sektora pamięci nieulotnej
- TVTTCC0_NVMEM_WRITE – kontrolowanie ilości zapisów powiązanego sektora pamięci nieulotnej
- TVTTCC1_PASS – kontrolowanie ilości wykonanych operacji szyfrowania/deszyfrowania za pomocą powiązanego klucza
- TVTTCC1_NVMEM_DECRYPT_WRITE – kontrolowanie ilości przeprowadzonych operacji zapisu z uprzednim odszyfrowaniem do powiązanego sektora pamięci nieulotnej. Opcja dostępna od wersji „2” API TeleTokenu i wersji „1.03” biblioteki.

Taką funkcjonalność również można wykorzystać do pilnowania ilości uruchomień jakiejś funkcjonalności/modułu. Wyobraźmy sobie następującą sytuację (załóżmy wykorzystanie klucza i licznika numer 5):

- Licznik uniwersalny numer 5 został początkowo zapisany wartością 100.
- Licznik numer 5 jest skonfigurowany dla celów kontrolowania ilości szyfrowania powiązanym kluczem (TVTTCC1_PASS), oraz do zatrzymania się po osiągnięciu wartości zero (TVTTCC0_STOP_AT_ZERO).
- W aplikacji zaszyty jest blok 16-bajtów.
- Blok ten szyfrujemy TeleTokenem, kluczem o numerze 5.
- W zależności od zawartości licznika 5 przed operacją szyfrowania, operacja ta powiedzie się, lub nie. W buforze mamy więc blok odpowiednio: zaszyfrowany, bądź nie zmieniony w stosunku do początkowego (zaszytego w aplikacji).
- Otrzymany blok można teraz np. wpleść w mechanizm autoryzacji, czyli hasło użytkownika przed zweryfikowaniem może być przeszyfrowane naszym blokiem.

Sposoby użycia TeleTokenu - propozycje

Sposobów na wykorzystanie TeleTokenu we własnym projekcie jest bardzo dużo, postaramy się tutaj tylko przybliżyć wybrane rozwiązania, które mogą posłużyć jako źródło pomysłów, baza, do tworzenia własnego.

Siła TeleTokenu: kryptografia

Najważniejszą funkcją TeleTokenu jest możliwość szyfrowania/desyfrowania algorytmem AES (Rijndael), za pomocą wewnętrznych kluczy o długości 128 bitów. Konstrukcja TeleTokenu gwarantuje że klucze te nigdy go nie opuszczą, istnieje tylko możliwość ich zapisania przez użytkownika (programistę). Operacja szyfrowania/desyfrowania przebiega w całości wewnątrz urządzenia. Dzięki temu mamy gwarancję, że praktycznie nie ma możliwości odszyfrowania danych zaszyfrowanych TeleTokenem innej, jak fizyczne zdobycie i użycie konkretnego tokenu.

Propozycja pierwsza: TeleToken jako narzędzie szyfrujące/desyfrujące

Domyślnym zastosowaniem TeleTokenu będzie więc szyfrowanie i deszyfrowanie danych. Zasyfrowanie pojedynczych bloków o długości do 16 bajtów (czyli 128 bitów) nie nastręczy trudności, funkcja szyfrowania TeleTokenu jako argument przyjmuje bowiem blok o takiej długości. Co jednak zrobić z dłuższymi danymi? A dokładniej, jak je przeszyfrować dobrze? Można oczywiście po prostu podzielić nasze dane na 16 bajtowe bloki i przeszyfrować je każdy oddzielnie za pomocą TeleTokenu. Nie jest to jednak dobre rozwiązanie. Po pierwsze wymaga użycia TeleTokenu (dane/16) razy, a każda taka operacja wymaga odwołania do sprzętu. Trzeba również pamiętać o maksymalnym czasie trwania sesji, który wynosi 5 sek., jeśli będziemy potrzebować więcej czasu, będziemy musieli ponownie rozpocząć sesję. Wszystko to razem daje nam czas, który może się okazać nie do zaakceptowania. Po drugie zaś, szyfrowanie każdego bloku osobno powoduje, że w przypadku identycznych bloków wejściowych, otrzymujemy identyczne bloki wyjściowe. To w jakiś sposób może potencjalnemu atakującemu dać pewne informacje, a tego wolelibyśmy uniknąć. Proponujemy zatem użycie trybów szyfru blokowego. Tryby te pozwalają ukryć identyczne bloki danych wejściowych. Dodatkowo będziemy wykorzystywać TeleToken tylko do szyfrowania „klucza sesji”, co zdecydowanie przyspieszy całą operację. Wadą natomiast tego rozwiązania jest konieczność przechowywania dodatkowych informacji z zaszyfrowanymi danymi (konkretnie zaszyfrowanych: „klucza sesji” oraz „wektora początkowego” IV, co daje razem 32 bajty) Po dokładniejszy opis trybów blokowych odsyłamy do [1], natomiast tutaj przybliżymy nasze ulubione rozwiązania.

Generowanie „klucza sesji szyfrowania”

W obu poniższych trybach użyjemy „klucza sesji szyfrowania” do szyfrowania danych, ze względu na szybkość działania. Można to zrobić w sposób następujący:

- Używając generatora liczb pseudolosowych generujemy blok 16 bajtów
- Tworzymy kopię wygenerowanego bloku

- Wygenerowany blok (oryginał) szyfrujemy TeleTokenem, za pomocą wybranego klucza
- Otrzymany blok stosujemy do szyfrowania trybem blokowym (to jest nasz „klucz sesji szyfrowania”)
- Kopię bloku (z przed zaszyfrowania) dołączamy do zaszyfrowanych danych, posłuży nam do ich odszyfrowania

Tryb łańcuchowego szyfru blokowego (CBC)

Tryb ten pozwala na ukrycie identycznych bloków wejściowych, poprzez xorowanie każdego następnego bloku danych przed ich zaszyfrowaniem, z zaszyfrowanym już poprzedzającym.

Algorytm tego trybu wygląda następująco:

- Uzupełniamy nasze dane do wielokrotności 16 bajtów
- Generujemy „klucz sesji szyfrowania” (wg opisu wyżej)
- Losujemy tzw. „wektor początkowy” (dalej będziemy go nazywać IV) – w postaci bloku 16 bajtów
- Tworzymy kopię IV (dołączymy ją do zaszyfrowanych danych)
- Szyfrujemy IV TeleTokenem (np. tym samym kluczem, którym wcześniej przeszyfrowaliśmy „klucz sesji szyfrowania”)
- Wykonujemy operację XOR pierwszego bloku (16 bajtów) naszych danych z zaszyfrowanym blokiem IV
- Szyfrujemy otrzymany blok algorytmem AES (Rijndael), przy pomocy „klucza sesji”
- Dla każdego kolejnego bloku danych powtarzamy operację XOR z poprzedzającym zaszyfrowanym blokiem, a następnie szyfrujemy „kluczem sesji”

Aby rozszyfrować dane należy:

- Przygotować „klucz sesji szyfrowania” oraz IV, odzyskując odpowiednie bloki dołączone do zaszyfrowanych danych i szyfrując je TeleTokenem
- Odszyfrować ostatni blok algorytmem AES (Rijndael), przy pomocy „klucza sesji”
- Wykonać XOR odszyfrowanego bloku z blokiem poprzedzającym (zaszyfrowanym)
- Powtarzać operację deszyfrowania i xorowania dla poprzednich bloków
- Ostatni w kolejności (czyli pierwszy) blok danych należy zxorować z odzyskanym IV

Tryb licznika (CTR)

W odróżnieniu od poprzednika jest to szyfr strumieniowy, a więc nie wymusza konieczności dopełniania danych do wielokrotności długości pojedynczego bloku. Dodatkowo będziemy korzystać jedynie z funkcji szyfrującej, zarówno przy szyfrowaniu jak i odszyfrowywaniu danych. Natomiast trzeba zadbać tutaj o niepowtarzalność numeru jednorazowego, używanego jako „baza” szyfru blokowego.

Algorytm szyfrowania:

- Generujemy „klucz sesji szyfrowania” (wg opisu wyżej, identycznie jak w poprzednim trybie)
- Pobieramy unikalny, jednorazowy numer i łączymy go z licznikiem
- Szyfrujemy otrzymany blok „kluczem sesji”
- Wykonujemy operację XOR naszych danych z otrzymanym blokiem szyfru
- Jeśli przekroczyliśmy rozmiar bloku (16 bajtów), zwiększamy licznik, łączymy go z numerem jednorazowym, szyfrujemy, używamy do operacji XOR

Deszyfrowanie przebiega w tym trybie tak samo jak szyfrowanie, od momentu odzyskania jednorazowego numeru i „klucza sesji szyfrowania”.

Propozycja druga: Wykorzystanie TeleTokenu do autoryzacji

Przypuśćmy, że w projektowanym systemie będzie podział na użytkowników, a dostęp będzie wymagał logowania. Zwykle hasło to jednak dla nas za mało. Można zaprojektować system w ten sposób, aby hasło użytkownika po wprowadzeniu, było przesyfrowywane TeleTokenem, a dopiero następnie weryfikowane. Uzyskujemy więc 2 poziomy zapewnienia bezpieczeństwa hasło + TeleToken. Dodatkowo w prosty sposób można teraz przygotować TeleToken dla konkretnego użytkownika systemu w ten sposób, aby mógł on się w nim zalogować określoną ilość razy. Można to uzyskać konfigurując licznik powiązany z kluczem TeleTokenu używanym do autoryzacji, jako kontroler ilości szyfrowań (*Patrz: „Dodatkowe właściwości TeleTokenu/Liczniki uniwersalne”*).

Propozycja trzecia: TeleToken ze zbiornikiem danych

Możemy chcieć w wewnętrznej pamięci TeleTokenu przechować jakieś informacje, np. dotyczące możliwości/ilości uruchomienia jakichś modułów projektowanego przez nas systemu na komputerze klienta, do którego podłączony jest konkretny TeleToken. Jakkolwiek można korzystać z pamięci TeleTokenu jako ze zwykłego pojemnika na dane, proponujemy przesyfrowanie istotnych danych przed ich zapisem. Można sobie wyobrazić np. taką sytuację:

- Dla konkretnego klienta (tokenu) generowany jest klucz 128 bitowy
- Klucz jest zapisywany w centralnym repozytorium kluczy
- Przy generowaniu tokenu dla konkretnego odbiorcy wpisywany jest do niego klucz z repozytorium, oraz zawartość pamięci zaszyfrowana tym kluczem
- Aplikacja działająca u klienta przy odczycie danych z TeleTokenu, rozszyfrowuje je wpisanym w TeleToken kluczem
- Jeśli zajdzie konieczność zmiany zawartości pamięci w TeleTokenie klienta, należy skorzystać z klucza z repozytorium do jej zaszyfrowania

Niezależnie, od wersji „2” API TeleTokenu i wersji „1.03” biblioteki istnieje możliwość zapisu danych zaszyfrowanych, które zostaną podczas zapisu automatycznie odszyfrowane określonym kluczem wewnątrz TeleTokenu. Pozwala to na zorganizowanie przesyłu informacji do zapisania np. przez sieć.

Propozycja czwarta: Liczniki uniwersalne

Liczniki, jak to już było wspomniane, można wykorzystać np. do zliczania ilości uruchomień aplikacji/modułu/funkcjonalności. Można zawartość licznika odczytywać wprost, lub, jak to było podane w propozycji drugiej, skonfigurować jako kontrolery ilości operacji szyfrowania/deszyfrowania za pomocą powiązanego klucza w TeleTokenie. Na tej samej zasadzie można licznik skonfigurować jako kontroler ilości odczytów i/ lub zapisów do powiązanego sektora pamięci nieulotnej. W ten sposób również można ograniczyć jakąś

funkcjonalność pod względem ilości wykorzystania. Liczniki można również potraktować jako miejsce do zapisania 2-bajtowych wartości liczbowych, albo 16 bitów konfiguracji.

Podsumowanie

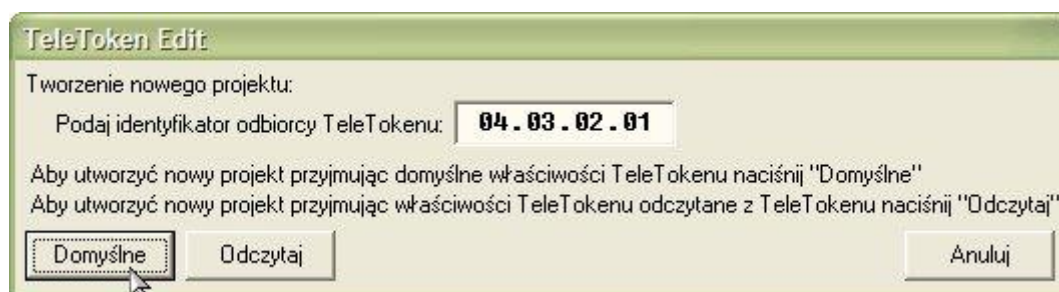
Przedstawione wyżej propozycje mają zachęcić do opracowania własnych rozwiązań, podpowiadając drogi które można wybrać. Oczywiście w/w propozycje można, a nawet wskazane jest, łączyć ze sobą, w celu uzyskania większej funkcjonalności/bezpieczeństwa. W następnym dziale można znaleźć podstawy konfiguracji i komunikacji z TeleTokenem, czyli tzw. „szybki start”.

Szybki start

Spróbujemy przygotować TeleToken do jakiegoś konkretnego działania. Załóżmy, że chcemy mieć możliwość zapisu/odczytu pamięci wewnętrznej. Dodatkowo chcemy mieć jeden klucz z możliwością szyfrowania, będziemy go używać w aplikacji do szyfrowania i deszyfrowania blokowego w trybie licznika. Załóżmy jeszcze, że jeden z sektorów pamięci będzie szyfrowany, wykorzystamy do tego jeszcze jeden klucz, będzie więc w nim potrzebna możliwość zarówno szyfrowania jak i deszyfrowania.

Konfigurowanie TeleTokenu

Aby stworzyć projekt w programie TeleTokenEdit naciśnij przycisk „Nowy”. Pojawi się okienko, w którym należy podać ID TeleTokenu. Posłużymy się tutaj TeleTokenem w wersji demo.

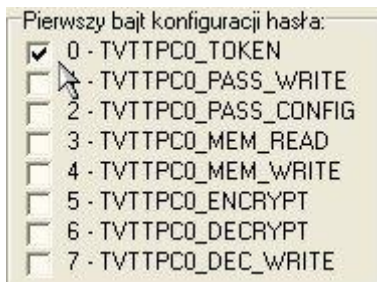


Po podaniu ID naciśnij przycisk „Domyślne”. Spowoduje to stworzenie nowego, pustego projektu dla TeleTokenu o podanym ID.

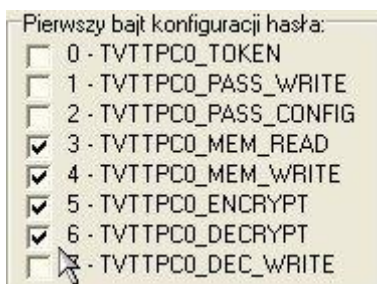
Od teraz w każdym momencie możesz skorzystać z opcji „Zapisz” w celu zapisania obecnego stanu projektu na dysku. Opcjonalnie możesz skorzystać z szyfrowania pliku przy zapisie (podając hasło dostępu do pliku projektu). Zapisany projekt można wczytać do edytora TeleTokenEdit opcją „Wczytaj”. Pliki projektu mają rozszerzenie „.ttp”.

Skupimy się na zakładce „Hasła”, ponieważ tutaj znajdują się wszystkie opcje potrzebne do zrealizowania naszych założeń. W górnej części okna widnieją opcje (bity konfiguracyjne) hasel, odnoszą się one do aktualnie wybranego (zaznaczonego) na liście w dolnej części okna hasła (klucza). To samo dotyczy się edycji samego hasła, pole edycyjne znajduje się w środkowej części okna. Na samym dole mamy jeszcze dodatkowo możliwość przetestowania naszego klucza, poprzez zaszyfrowanie i odszyfrowanie bloku o długości 16 bajtów.

Na pewno potrzebne nam będą klucze komunikacyjne, stwórzmy więc jedną parę kluczy z odpowiednią konfiguracją (pamiętając o tym co chcemy robić za pomocą TeleTokenu). Pierwszy klucz przeznaczymy na „klucz tokenu”. Czyli jego konfiguracja wyglądać będzie następująco:

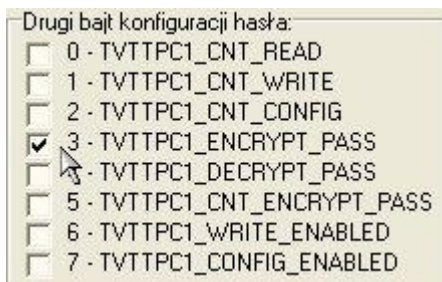


Drugi klucz będzie „kluczem aplikacji”, poniżej jego konfiguracja, uwzględniająca zastosowanie:

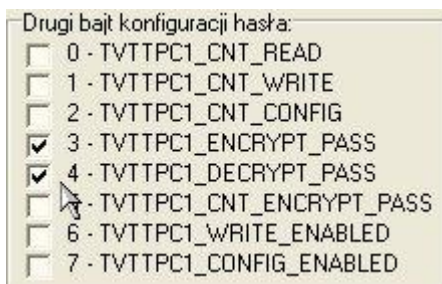


Jak widać po otwarciu sesji tym kluczem, aplikacja będzie mogła odczytać pamięć, zapisać pamięć, wykonać operację szyfrowania, oraz deszyfrowania.

Dla spełnienia naszych założeń musimy stworzyć jeszcze 2 klucze. Oto konfiguracja klucza nr 2, który będzie użyty do szyfrowania:



Oraz klucz nr 3, przeznaczony do szyfrowania i deszyfrowania:



Tak to może wyglądać na liście kluczy:

Tablica haseł TeleTokenu:

Nr.	1-bajt konf.	2-bajt konf.	Hasło
0	0-----	-----	F3 AB C5 80 42 8C B6 8D ED E2 C7 08 9D 57 C5 15
1	---3456--	-----	42 F9 1D 25 29 63 D9 0F 18 3C 04 9E AE F7 2D FA
2	-----	---3---	95 59 8C 48 D1 A0 58 01 36 2A 5B E8 FF 58 72 14
3	-----	---34---	F6 15 DC 90 B9 FE 8C BB 82 92 32 DE A6 AC 29 00
4	-----	-----	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
5	-----	-----	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
6	-----	-----	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Stworzyliśmy projekt spełniający nasze założenia. Możemy teraz zaprogramować TeleToken używając przycisku „Programuj”. Proszę zwrócić uwagę na fakt, że ponieważ nie pozostawiliśmy możliwości ani zmiany konfiguracji kluczy, ani nadpisania samych kluczy, przed następnym programowaniem będziemy zmuszeni sformatować TeleToken (przycisk „Formatuj”).

Komunikacja z TeleTokenem

Szczegółowy opis poleceń biblioteki „teletoken.dll” znajduje się w dokumentacji TeleTokenu („teletoken.pdf”), po szczegóły dotyczące implementacji komunikacji kierujemy do przykładów dołączonych do pakietu programisty.

Po pierwsze trzeba wykryć podłączony TeleToken w systemie:

```
#define TOKEN_ID 0x04030201
#define TOKEN_ID_MASK 0xff
TVTT_FindToken(&TokenH,10,TOKEN_ID,TOKEN_ID_MASK,0,&TokenInfoTblSize,TokenInfoTbl)
```

TokenInfoTbl to tabela informacyjna TeleTokenu, jest ona automatycznie wypełniana po poprawnym otwarciu urządzenia. Można znaleźć tu informacje dotyczące konkretnego tokenu, takie jak: rozmiar pamięci, ilość dostępnych kluczy, liczników, itp. Szczegóły znaleźć można w dokumentacji TeleTokenu.

Jak widać szukamy tokenu z konkretnym ID (tu: wersja demo). Wybrana maska powoduje odnalezienie TeleTokenu tylko jeśli zajdzie dokładne dopasowanie. Funkcja ustawia uchwyt na TeleToken, możemy z niego korzystać dopóki nie zwolnimy tokenu, lub urządzenie nie zostanie odłączone od komputera. Po ponownym podłączeniu uchwyt nie jest już prawidłowy.

W celu wykonania jakiegokolwiek operacji musimy rozpocząć sesję.

Do bezpośredniej komunikacji z TeleTokenem zawsze używamy funkcji TVTT_Talk. Rozkaz (który ma zostać wykonany przez urządzenie) oraz parametry przekazujemy poprzez drugą tabelę wysyłaną tym rozkazem do TeleTokenu.

```
// podstawienie numeru rozkazu, tutaj jest to otwarcie sesji
OutDataB[TVTTD_CMD]=TVTTC_OPEN;
// numer klucza proponowanego przez nas, czyli aplikację (musi być zgodny z konfiguracją tokenu)
OutDataB[TVTTD_PARAM1]=1;
```

Po wykonaniu funkcji TVTT_Talk TeleToken zwraca numer proponowanego przez siebie klucza (w naszym przypadku zawsze będzie to „0”), oraz wylosowaną tablicę, która posłuży do budowania klucza sesji.

Następnie w wyniku użycia rozkazu TVTT_CalcSessionKey zostaje wyliczony **klucz sesji**, którym szyfrowana będzie komunikacja w obrębie otwartej sesji. Rozkaz przyjmuje jako parametry klucze komunikacyjne (aplikacji, w naszym przypadku jest to klucz o numerze „1” i TeleTokenu – „0”, dlatego aplikacja musi znać oba), oraz bloki losowe, jeden wylosowany przez aplikację, drugi zaproponowany przez token (odesłany przez funkcję TVTT_Talk).

Po poprawnym rozpoczęciu sesji możemy wykonać np. odczyt pamięci TeleTokenu, operację szyfrowania/desyfrowania, pamiętając jednak o maksymalnym czasie trwania sesji, który wynosi 5 sek. Jeśli przekroczyliśmy ten czas, lub z innych powodów sesja zostanie zamknięta, będziemy musieli otworzyć ją na nowo. Nie istnieje potrzeba wykrywania TeleTokenu na nowo, dopóki mamy prawidłowy uchwyt na urządzenie.

Po wykonaniu operacji należy zamknąć sesję i ew. zwolnić uchwyt na połączenie z TeleTokenem, jeśli nie mamy zamiaru więcej z niego korzystać.

Literatura

[1] Niels Ferguson, Bruce Schneier. *Kryptografia w praktyce*. Wydawnictwo HELION, 2004. ISBN 83-7361-211-4.