Ochrona kodu Pythona

Python dzięki swojej wszechstronności oraz ogromnej ilości bibliotek jest często wybierany do tworzenia aplikacji desktopowych, webowych i embedded. Znajduje zastosowanie w rozwiązaniach tworzonych na potrzeby analizy danych i symulacji, uczenia maszynowego czy projektów IoT.

Jako że Python jest językiem skryptowym, kod aplikacji zapisany w plikach .py jest czytelny dla każdego, kto ma do nich dostęp. Kompilacja kodu do plików .pyc (bytecode) czy generowanie samodzielnych plików wykonywalnych sprawia, że jest mniej czytelny, ale nadal możliwy do wyekstrahowania.

Niniejsze wprowadzenie ma na celu przedstawienie, w jaki sposób prosto i skutecznie ochronić swój kod napisany w Pythonie, bez potrzeby stosowania dodatkowych zabiegów. Poprowadzi Cię przez proces zabezpieczania, w którym wykorzystasz pakiet AxProtector Python. Skupimy się na najczęściej wykorzystywanych opcjach, tak abyś mógł stworzyć własną, bezpieczną dystrybucję aplikacji w zaledwie kilka minut.

W celu zabezpieczenia oprogramowania napisanego w Pythonie potrzebne będą:

- CodeMeter AxProtector NC GUI
- Firm Security Box (FSB)
- CodeMeter License Editor
- Python

INSTALACJA ŚRODOWISKA CODEMETER

AxProtector NC GUI wchodzi w skład pakietu narzędziowego **CodeMeter Development Kit,** który można pobrać ze strony: https://www.wibu.com/pl/sdk.html

Pakiet do celów testowych jest darmowy. Po wypełnieniu formularza rejestracyjnego wygenerowana zostanie odpowiednia licencja testowa. Po zainstalowaniu pakietu mamy do dyspozycji wszystkie istotne komponenty środowiska CodeMeter, które pozwolą nam na ochronę kodu w Pythonie.

KONTENERY LICENCYJNE I FIRM SECURITY BOX (FSB)

AxProtector zabezpiecza kod wynikowy m.in. za pomocą szyfrowania. Wykorzystywane w tym celu klucze przechowywane są w bezpiecznych repozytoriach – kontenerach licencyjnych. W nich również zapisywane są licencje, które określają zakres użycia Twojej aplikacji.

Rozróżniamy następujące kontenery licencyjne:

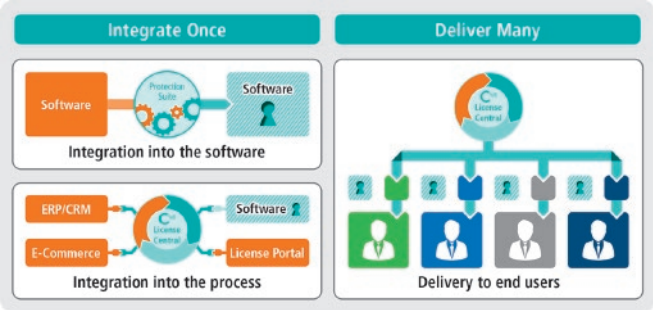
1. **CmDongle** – potocznie określany jako „klucz sprzętowy". Jest to karta mikroprocesorowa (np. w postaci niewielkiego pendriva), wyposażona w procesor kryptograficzny, który m.in. wykonuje operacje szyfrowania i deszyfrowania, podłączana do komputera, serwera bądź urządzenia IoT/embedded (np. poprzez port USB). Używane do szyfrowania klucze są w nim trwale zapisane.
2. **CmCloud** – to chmurowa wersja repozytorium kluczy. Jeśli Twoja aplikacja pracuje online i ma stały dostęp do Internetu (środowiska wirtualne, chmurowe, IoT/IIoT), CmCloud stanowić będzie interesującą opcję.
3. **CmActLicense** – to kontener w postaci pliku, unikalnego identyfikatora (odcisku) cyfrowego urządzenia docelowego (komputera, urządzenia IoT/embedded). Zmiana wybranych składników sprzętu wymaga ponownego wygenerowania pliku. CmActLicense zapisywany jest na urządzeniu docelowym. CmReady to wariant, który powiązany jest z kartą SD, znajdujący często zastosowanie w urządzeniach embedded i IoT.

Zabezpieczenie oprogramowanie za pomocą AxProtector wymaga kluczy, które zapisane są w kontenerze licencyjnym, tzw. **Firm Security Box (FSB),** i jednoznacznie identyfikują Cię jako producenta.

OCHRONA OPROGRAMOWANIA – PROCES

Zabezpieczenie kodu wynikowego składa się z dwóch kroków, które mogą być wykonane w dowolnej kolejności (Rysunek 1):

- Krok 1 – **Integracja** zabezpieczeń z aplikacją. Wykonywana jest jednokrotnie, dla każdego nowego wydania.
- Krok 2 – **Tworzenie licencji** – czyli zapisanie kluczy umożliwiających odszyfrowanie kodu aplikacji oraz zasad ich użycia (licencji) w jednym z trzech w/w rodzajów kontenerów.

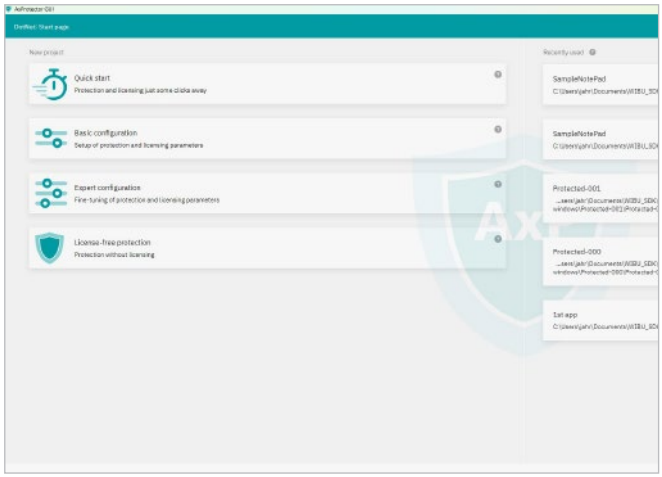


Rysunek 1. Sposób integracji zabezpieczeń z oprogramowaniem i tworzenie licencji

OCHRONA KODU PYTHONA PRZY UŻYCIU AXPROTECTOR

Środowisko AxProtector NC służy do ochrony kodu wynikowego. Mamy do wyboru 4 warianty, które przechodzimy za pomocą prostego konfiguratora:

- Quick start – zawiera standardowe ustawienia, pozwalające na szyfrowanie i licencjonowanie aplikacji.
- Basic configuration – daje możliwość zmiany parametrów konfiguracyjnych.
- Expert configuration – umożliwia określenie konkretnych części aplikacji (metod), do których można przypisać dany typ licencji.
- License-free protection – aplikacja zostaje zabezpieczona, z pominięciem licencjonowania.



Rysunek 2. Ekran startowy CodeMeter AxProtector NC GUI

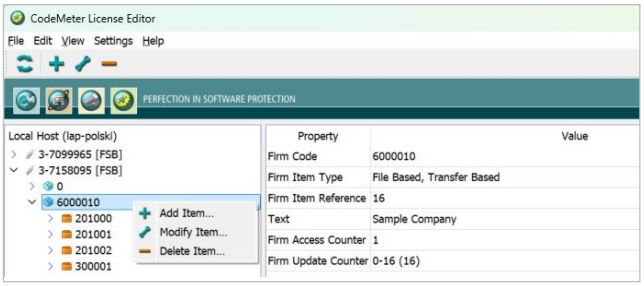
W każdym z wariantów należy skonfigurować parametry dot. licencji:

- Firm Code – to unikalny kod producenta. Wersja testowa ma domyślnie 6000010.
- Product Code (wariant 1-3) – to kod, który nadajesz danym licencjom Twojej aplikacji (funkcjonalności, cechy itp.), np. 201 000.

TWORZENIE LICENCJI

Za wyjątkiem wariantu License-free protection zabezpieczona aplikacja może być uruchomiona wyłącznie w przypadku, gdy jest dostępna odpowiednia licencja.

W zależności od docelowego środowiska wybieramy kontener licencyjny (CmDongle, Cloud lub CmActLicense) i zapisujemy w nim odpowiednią licencję (np. 201 000). Operację możemy wykonać na kilka sposobów, lecz na potrzeby testów oraz niedużej ilości licencji wystarczający jest CodeMeter License Editor.



Rysunek 3. Menu kontekstowe CodeMeter License Editor. Tworzenie licencji na kluczu sprzętowym 3-7158095, dla kodu producenta (testowy) 6000010

URUCHAMIANIE ZABEZPIECZONEJ APLIKACJI PYTHON

Na docelowym komputerze/urządzeniu instalowana jest natywna biblioteka *cprst.dll*, która może być dostarczana razem z CodeMeter Runtime lub osobno z aplikacją.

Biblioteka tworzy bezpieczne i odporne na manipulacje środowisko, w którym ma miejsce sprawdzanie licencji i odszyfrowywanie kodu aplikacji. W momencie uruchomienia biblioteka ładowana jest do pamięci i odszyfrowywany jest kod aplikacji, który następnie wykonywany jest przez interpreter.

DALSZA EKSPLORACJA

Omawiany scenariusz pozwala na zabezpieczenie i opcjonalne przypisanie licencji do aplikacji w kilku prostych krokach. Nie wyczerpuje wszystkich możliwości, jakie daje AxProtector w zakresie ochrony kodu aplikacji Python, ale jest dobrym punktem wyjścia do sprawdzenia jego możliwości. W kolejnych artykułach zostanie omówiona implementacja zaawansowanych mechanizmów licencjonowania oraz użycie AxProtector w ramach zautomatyzowanego systemu kompilacji.

JANUSZ HRYSZKIEWICZ

janusz.hryszkiewicz@wibu.com

Związany z IT od 25 lat, głównie w zakresie rozwiązań dedykowanych (Microsoft, Java, Python, SAP). Autor kilku rozwiązań dla biznesu, które stały się produktami. Obecnie zaangażowany w zwiększenie cyberodporności oprogramowania, AI oraz systemów wbudowanych.