01

Bindery

UKRYWANIE WIRUSÓW W APLIKACJACH

hacking, cyberbezpieczeństwo i sztuczna inteligencja

opracowanie: Michał Suchoń

NEXT

Przypomnienie... z poprzedniego semestru!

I. W jakim pliku i lokalizacji są przechowywane hasła w systemie Kali Linux?

II. Jak nazywał się program do łamania haseł w Kali Linux?

III. Co to jest hash?

Przypomnienie... z poprzedniego semestru!

I. W jakim pliku i lokalizacji są przechowywane hasła w systemie Kali Linux?

Plik shadow w katalogu /etc

II. Jak nazywał się program do łamania haseł w Kali Linux?

III. Co to jest hash?

Przypomnienie... z poprzedniego semestru!

I. W jakim pliku i lokalizacji są przechowywane hasła w systemie Kali Linux?

Plik shadow w katalogu /etc

II. Jak nazywał się program do łamania haseł w Kali Linux?

John the Ripper (johnny)

III. Co to jest hash?

Przypomnienie... z poprzedniego semestru!

I. W jakim pliku i lokalizacji są przechowywane hasła w systemie Kali Linux?

Plik shadow w katalogu /etc

II. Jak nazywał się program do łamania haseł w Kali Linux?

John the Ripper (johnny)

III. Co to jest hash?

Quasi-przypadkowy ciąg znaków powstały w wyniku przetworzenia tekstu w formie jawnej przez funkcję haszującą (skrótu)

Czym jest wirus tzw. Trojan?
Co jest jego cechą szczególną?

Koń trojański jest przykładem bindera!

BINDERY

Są to programy (wirusy), które ukrywają malware jak keyloggery czy konie trojańskie w zwyktych programach poprzez łączenie ich w jeden plik wykonywalny (np. exe, com, bat...)

Po co stosować bindery?

Po co stosować bindery?

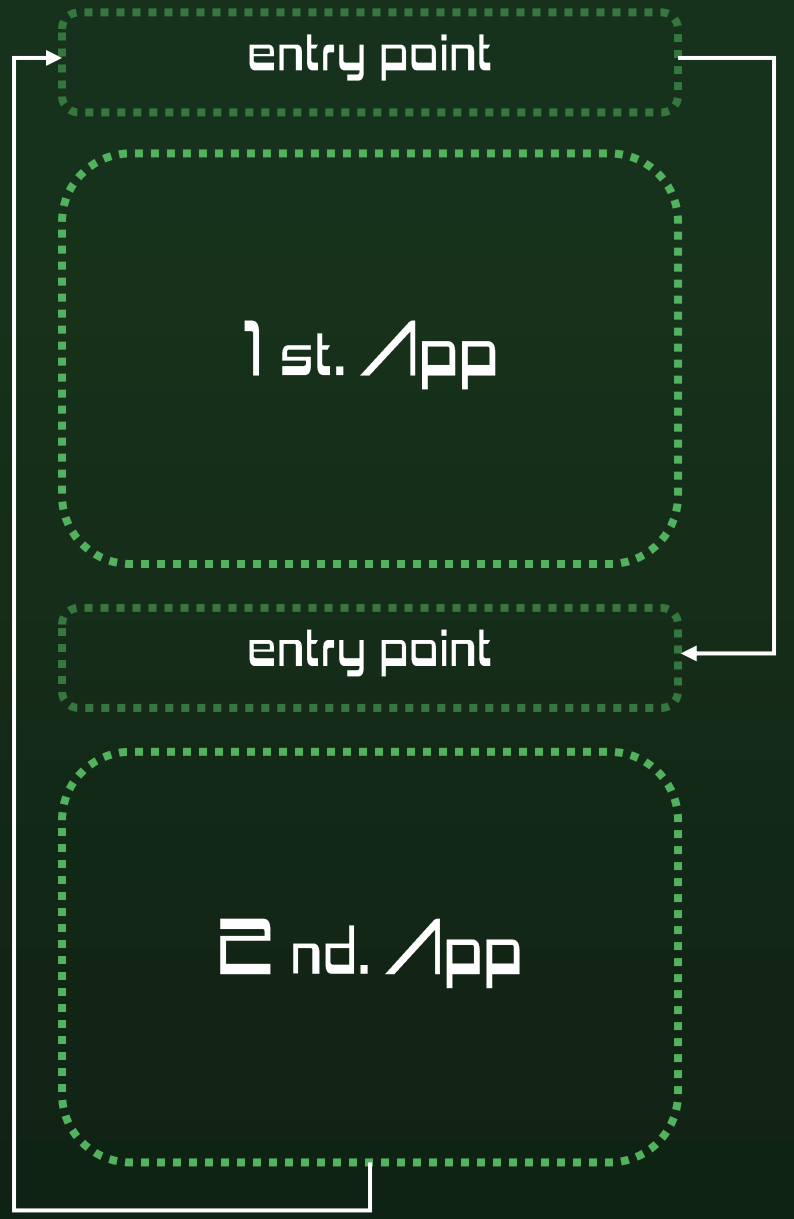
ABY NIE WZBUDZIĆ PODEJRZEŃ UŻYTKOWNIKA. INSTALUJĄC CZYTNIK PDF NIE ZAKŁADAMY, ŻE KTOŚ UZYSKA DOSTĘP DO NASZEGO KOMPUTERA, A JEDNOCZEŚNIE GDY SYSTEM ZAPYTA O DODATKOWE UPRAWNIENIA (TZW. SYSTEM UVAC) - ODRUCHOWO JE PRZYZNAMY, ABY CZYTNIK NA PEWNO ZADZIAŁAŁ

Działanie bindera technicznie:

Załóżmy, że plik .exe waży 800 kB. Dokładamy do tego plik z wirusem o wadze 50 kB, w rezultacie plik wynikowy będzie wynosił 850 kB. Jest to niezauważalna różnica, która sprawia, że użytkownik nie zwróci na to większej uwagi.

Plik ten dodatkowo ma modyfikowane punkty uruchomienia, przez co aplikacje (właściwa i wirus) uruchamiają się jednocześnie a nie sekwencyjnie.

entry point -> punkt początkowy, adres startowy, pod którym system zaczyna uruchamiać program



Co możemy „wrzucić" jako binder?

Co możemy „wrzucić" jako binder?

ZŁOŚLIWE OPROGRAMOWANIE DOŁĄCZANE DO NORMALNEGO PROGRAMU (BINDOWANIE) NAZYWAMY FACHOWO PAYLOADEM

1. RAT – *Remote Access Trojan; rodzaj konia trojańskiego, w którym atakujący próbuje uzyskać zdalny dostęp i przejąć kontrolę nad systemem ofiary*
2. Rootkity – wirus, który ukrywa programy bądź procesy w systemie operacyjnym, utrudniając ich wykrycie
3. Ransomware – wirus, szyfrujący pliki na komputerze ofiary z „gwarancją" przywrócenia ich za zapłatą okupu

Bindery - pros and cons

ZALETY (+)

- szkodliwy program jest ukrywany i trudny do wykrycia
- stosunkowo prosty do wykonania (scalanie plików binarnych + drobna modyfikacja)

Sygnatura -> ciągła sekwencja bajtów w bazie antywirusa, która jest utożsamiana z określoną próbką złośliwego programu

WADY (-)

- ryzyko wykrycia przez antywirusa, który jeśli posiada sygnaturę złośliwego programu w swojej bazie danych, to wirus zostanie zablokowany
- ochrona przez wykryciem bindera jest trudna i dość skomplikowana (konieczność tzw. zaciemniania kodu i wnikania w działanie antywirusa)

Jak chronić się przed binderem?

Jak chronić się przed binderem?

- Zainstalowanie antywirusa i jego ciągła aktualizacja
- Pobieranie oprogramowania wyłącznie przez oficjalne źródła (Softonic / dobreprogramy etc. = ryzyko bindera!)
- Skanowanie plików przez programy antywirusowe lub serwisy np. virustotal.com (choć z tym ostrożnie!)

Spróbujmy zbindować pliki...

Task 1

- Tworzymy 2 aplikacje konsolowe w C# (.Net Framework)
- Sprawdzamy, czy w folderze „Debug” po kompilacji pojawiły się pliki .exe
- Łączymy aplikacje ze sobą na jeden z wybranych sposobów:
  - Skrypt Visual Basic
  - Poprzez trzecią aplikację
  - Z wykorzystaniem pliku archiwum
- Testujemy wirusa

Task 1 - rozwiązanie

- PRZYGOTOWANIE APLIKACJI (WPISUJEMY POLECENIA W TERMINALU):

  TWORZENIE PROJEKTU Z APLIKACJĄ:

- *dotnet new console --use-program-main -n {nazwa projektu}*

- KOMPILACJA I URUCHOMIENIE PROJEKTU:

- *cd {nazwa projektu}*
- *dotnet run (po tym poleceniu możemy też uruchomić plik .exe z folderu Debug)*

- DODAJEMY PAKIET „NUGET” (system zarządzania pakietami):

- *dotnet nuget add source* https://api.nuget.org/v3/index.json -n nuget.org

- PRZYGOTOWUJEMY GOTOWEGO BUILDA W POSTACI JEDNEGO PLIKU .EXE (dzięki temu będziemy w stanie przenosić dowolnie plik wykonywalny) -> PO NAPISANIU APEK!

- *dotnet publish -c Release -r win-x64 -p:PublishSingleFile=true --self-contained true -o publish*

Aplikacja 1:

```
namespace aplikacja1;

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        Console.WriteLine("To jest aplikacja nr 1");
        Console.ReadKey();
    }
}
```

Aplikacja 2:

```
namespace aplikacja2;

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        Console.WriteLine("To jest aplikacja nr 2, która jest wirusem");
        Console.Beep(500, 700);
        Console.ReadKey();
    }
}
```

Sposób 1: SKRYPT VISUAL BASIC

ABY STWORZYĆ SKRYPT - TWORZYMY NOWY PLIK TEKSTOWY (.TXT), WRZUCAMY PONIŻSZE LINIJKI KODU, PO CZYM ZMIENIAMY NA ROZSZERZENIE .VBS

```
Set File1 = CreateObject("WScript.Shell")

File1.Run ("aplikacja1.exe")

Set File2 = CreateObject("WScript.Shell")

File2.Run ("aplikacja2.exe")
```

UWAGA! PLIK ZE SKRYPTEM MUSI BYĆ W TYM SAMYM FOLDERZE CO PLIKI .EXE APLIKACJI - BEZ TEGO NAM NIE ZADZIAŁA (wystarczy przekopiować i wrzucić pliki .exe z folderu *publish*, w którym powstanie nasz build projektu)

Sposób 2: TRZECIA APLIKACJA

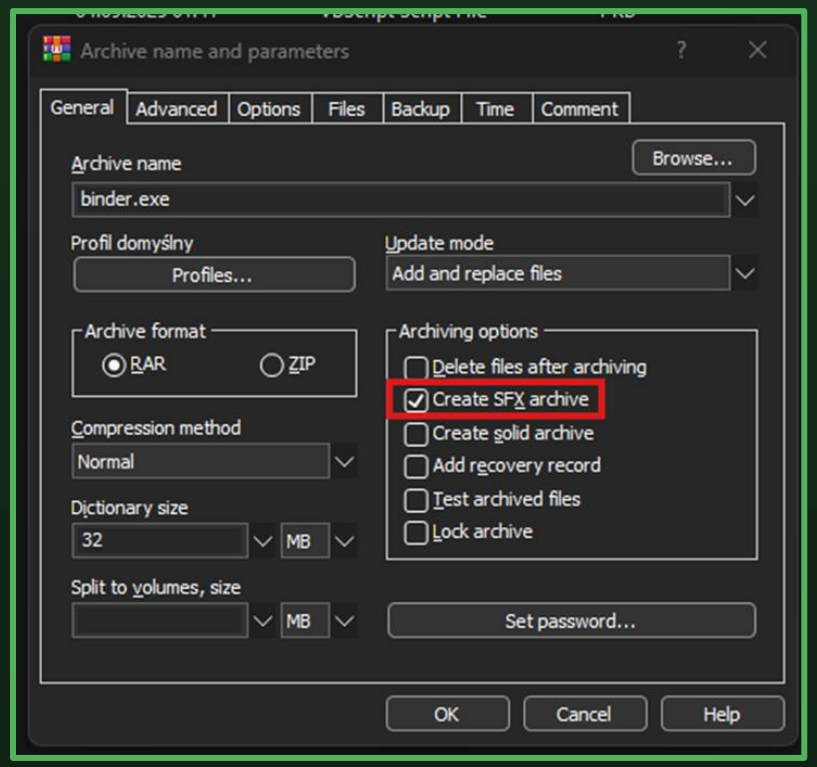
TWORZYMY TRZECI PROJEKT, WRZUCAMY KOD JAK PONIŻEJ, A NASTĘPNIE TWORZYMY BUILDA JAK W SPOSOBIE NR 1.

```
using System.Diagnostics;
namespace aplikacja3;

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        Console.WriteLine("Appka nadrzędna");
        Process.Start("aplikacja1.exe");
        Process.Start("aplikacja2.exe");

        Console.ReadKey();
    }
}
```

UWAGA! PLIK .EXE MUSI BYĆ W TYM SAMYM FOLDERZE CO PLIKI .EXE APLIKACJI - BEZ TEGO NAM NIE ZADZIAŁA (wystarczy przekopiować i wrzucić pliki .exe z folderu *publish*, w którym powstanie nasz build projektu)

Sposób 3: SAMOROZPAKOWUJĄCE SIĘ ARCHIWUM

W CELU WYKONANIA TEGO SPOSOBU POTRZEBUJEMY PROGRAMU WINRAR. ZAZNACZAMY OBA PLIKI .EXE APLIKACJI, A NASTĘPNIE WYBIERAMY „DODAJ DO ARCHIWUM". W okienku zaznaczamy opcję „Utwórz archiwum SFX", a następnie przechodzimy do sekcji Komentarz i wprowadzamy poniższy skrypt:




```
Silent=1    tzw. tryb cichy (niewidoczny)
Setup=aplikacja1.exe uruchomienie aplikacji 1
Setup=aplkacja2.exe uruchomienie aplikacji 2
Path=%tmp%\ ścieżka, gdzie zostanie wypakowane archiwum
```

UWAGA! PO KLIKNIĘCIU „OK" STWORZY SIĘ PLIK .EXE Z NAZWĄ PODANĄ W SEKCJI „NAZWA ARCHIWUM".

NALEŻY JEDNAK PAMIĘTAĆ, ŻE KOLEJNE URUCHOMIENIE MOŻE WYRZUCIĆ KOMUNIKAT O NADPISANIU PLIKÓW ZE ŚCIEŻKI TMP, CO MOŻE WZBUDZIĆ PODEJRZENIA! (dlatego najlepiej zmienić parametr Path na inny)

Podsumowanie

I. Co to jest binder?

II. Co dzieje się podczas procesu bindowania?

III. Jak sprawdzić, czy aplikacja nie zawiera payload'u?

EXIT