CATEGORIA EM ANDAMENTO

1. RESUMO

Com a atual alta de ataques no meio tecnológico, vários sistemas são comprometidos diariamente e, consequentemente, os dados destes sistemas que incluem, mas não estão limitados a: dados de pessoas, empresas e tecnologias. Desta maneira, este projeto visa investigar um dos ataques utilizados para comprometer sistemas: o KeyLogger. Um KeyLogger é um tipo de software que captura, armazena e transmite tudo que é digitado através de um teclado, físico ou virtual, em um sistema e é uma das principais ferramentas usadas em ataques cibernéticos. Através do desenvolvimento de um keylogger na prática, visamos entender seu funcionamento, quais são seus pontos fortes e quais são suas fraquezas. Além disso, realizamos a adição de recursos importantes como uma requisição através de um servidor DNS, visando evitar medidas de segurança como antivírus e firewalls, demonstrando assim uma infecção em caso prático. A criação desse software malicioso deixa explícito que qualquer pessoa com um conhecimento de programação pode fazê-lo e utilizá-lo tanto para fins acadêmicos, como neste caso, ou para fins maliciosos. Portanto, entender como funcionam esses programas e esses tipos de invasões é a chave para entender como é possível se defender de ataques que o utilizem. Além disso, o conhecimento adquirido durante o desenvolvimento pode também ser utilizado para prevenção de outros ataques, uma vez que vários ataques utilizam várias das mesmas técnicas.

2. INTRODUÇÃO

Atualmente um dos bens mais valiosos da humanidade é a informação. Não somente o conhecimento mas também a informação pessoal. Essa última chega a ser tão valiosa que empresas privadas, públicas e até mesmo governos tentam comprar tal informação [1] e, em alguns casos, também tentam a adquirir de forma ilegal [2]. A aquisição desta informação, quando não comprada, é feita de várias maneiras, desde funcionários insatisfeitos com seu empregador que decidem roubar dados (ou são pagos para fazê-lo) até casos onde temos ataques diretos e exploração de vulnerabilidades dos sistemas de informação [3].

Neste trabalho em especial, vamos explorar um dos métodos de ataque utilizados para roubo de dados: o Keylogger. Um Keylogger é um software que, uma vez presente em um sistema, irá registrar tudo que é digitado através de um teclado, ou teclado virtual no caso de smartphones, e fará o posterior envio dessa informação para um local externo. Desta maneira é possível conseguir dados como: usuários e senhas, dados de documentos sigilosos, informações de sistemas e muitas outras.

3. OBJETIVOS

Desta maneira, temos por objetivo principal deste trabalho o desenvolvimento de um Keylogger, afim de entender seu funcionamento e, entendendo seu funcionamento, propor a realização da proteção contra ataques que o utilizam. Conscientizando a todos que utilizam o meio digital.

4. METODOLOGIA

A linguagem de programação Python [4] foi escolhida por ser uma linguagem de fácil aprendizado e uma das linguagens de programação mais utilizadas atualmente [5]. Por se tratar de um projeto relativamente simples, não são necessárias muitas outras tecnologias além da linguagem de programação, suas bibliotecas existentes e qualquer sistema que suporte o desenvolvimento através da linguagem escolhida.

5. DESENVOLVIMENTO

O primeiro passo para o desenvolvimento de um Keylogger é conseguir acesso ao teclado. Tal acesso normalmente é alcançado através de funções de entrada e saída como, por exemplo, a função *input* da linguagem de programação Python. O problema de usar funções deste tipo, que tratam entrada e saída, é o escopo. Funções planejadas para tratar entrada e saída desta maneira não têm acesso a entrada e saída de outros programas, somente do programa que está sendo executado. Para solucionar este problema foi necessário encontrar uma maneira de ter acesso ao teclado diretamente. A solução encontrada foi utilizada a biblioteca *pynput* que nos dá acesso ao pressionamento de teclas em um teclado através de eventos: podemos associar uma função arbitrária que será executada quando uma tecla é pressionada. A figura 1, abaixo, exemplifica o seu funcionamento básico.

Figura 1. Representação do Pynput em código

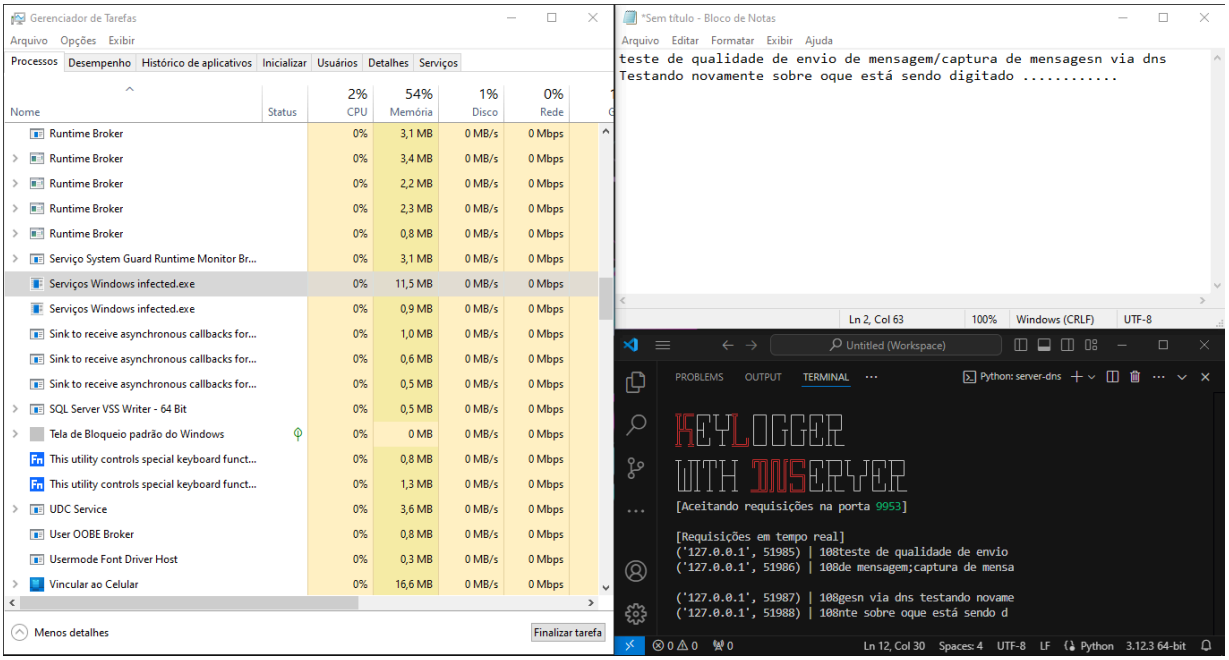
```
def on_release(key):
    global _KEYS
    if len(_KEYS) >= 31:
        send_data(_KEYS)
        _KEYS = []
    if key == keyboard.Key.esc:
        return False
    try:
        _KEYS.append(key.char)
    except AttributeError:
        if key == key.space or key == key.enter:
            try:
                if _KEYS[-1] == ' ':
                    pass
                else:
                    _KEYS.append(' ')
            except:
                pass
        elif key == key.backspace:
            _KEYS = _KEYS[:-1]
        else:
            print(key)

if __name__ == "__main__":
    lst = keyboard.Listener(on_release=on_release)
    lst.start()
    lst.join()
```

Fonte: Próprio Autor

Uma vez que temos acesso ao teclado, o próximo desafio foi encontrar uma maneira de executar o código sem que o usuário note a presença do mesmo. A resposta para este problema foi executar o código em forma de daemon: uma parte do processo original que é executada em segundo plano sem uma interface aparente. Para esse fim foi necessário utilizar algumas outras bibliotecas da linguagem de programação Python: *threading* e a *pyinstaller*. Podemos ver como isto foi feito na figura 2 abaixo.

Figura 2. Visualização do processo infectado.



Fonte: Próprio Autor.

Tendo uma maneira de adquirir toda a informação que fosse digitada através de um teclado sem que o usuário perceba, o próximo passo foi encontrar uma maneira de transferir esta informação do computador da vítima para um servidor seguro de maneira que fosse possível ter acesso à essa informação. A maneira mais simples de fazer essa transferência é através de sockets, usando a biblioteca *socket,* entretanto, essa maneira de solucionar o problema pode deixar a transmissão de dados muito aparente e, talvez, detectável por sistemas de proteção como antivírus. A solução encontrada foi fazer a transmissão dos dados através do protocolo DNS [7]. Deste modo os pacotes disfarçados como pacotes do protocolo DNS, que já são normalmente encontrados trafegando pela rede, serão menos suspeitos e não serão detidos por proteções como firewalls e antivírus tão facilmente. Obviamente é necessário controlar o servidor DNS que receberá as mensagens, pois um servidor de DNS honesto simplesmente descarta as mensagens uma vez que os nomes requisitados não existem. Resolvemos um problema ocultando a transmissão de dados desta maneira, mas criamos outro problema: pacotes do protocolo DNS têm um limite máximo de 63 bytes e não podemos usar caracteres especiais (caracteres com acento, por exemplo). Isto poderia ser resolvido com o processamento dos caracteres quando são capturados mas, como isto poderia criar problemas de desempenho do lado do cliente e alertar a vítima sobre algum problema do sistema,

resolvemos mandar os dados, codificados em bytes, usando hexadecimal, para o servidor e deixar o tratamento para ser feito do lado do servidor.

Do lado do servidor, uma vez que os dados são recebidos, fazemos o tratamento das mensagens e posterior armazenamento em arquivos de texto com identificação do cliente através do IP.

6. RESULTADOS PRELIMINARES

O software proposto está se mostrando estável nos testes de bancada e também nos testes em vários tipos de dispositivos espalhados pela rede, juntamente a isso, os dados estão sendo exfiltrados e tratados juntamente ao servidor com sucesso.

7. FONTES CONSULTADAS

[1] DIORIO, Rafael Fernando et al. Segurança da informação e de sistemas computacionais: Um estudo prático sobre ataques utilizando malwares. **Anais SULCOMP**, v. 9, 2018.

[2] DONNER, Christopher M. et al. Low self-control and cybercrime: Exploring the utility of the general theory of crime beyond digital piracy. **Computers in Human Behavior**, v. 34, p. 165-172, 2014..

[3] FAORO, ROBERTA RODRIGUES; DE JESUS, BETINA RIBEIRO; DE ABREU, MARCELO FAORO. UM ESTUDO SOBRE CRIMES DIGITAIS: DETECÇÃO E PREVENÇÃO. 2015.

[4] **Python**. Python, 2024. Disponível em: https://www.python.org/. Acesso em: 27/08/24.

[5] TIOBE. Tiobe Index for August 2024, 2024. Disponível em: https://www.tiobe.com/tiobe-index/. Acesso em: 02/09/2024.

[6] SEIXAS, Paulo Renato Lopes. Uma análise do protocolo DNS e suas extensões. **Cadernos de Educação, Tecnologia e Sociedade**, v. 1, n. 1, p. 161-171, 2008.