
Malware Research Team | LATAM
Spy.Banker.FN
Novo trojan bancário está sendo propagado no Brasil
Laboratório de Pesquisa e Investigação
ESET LATAM

Conteúdo

Através deste relatório, o Laboratório de Pesquisas e Investigação da ESET América Latina reporta um código malicioso, neste caso um trojan bancário, detectado por nossas soluções de segurança como ***MSIL /Spy.Banker.FN***, que afeta os sistemas operacionais Windows.

Este código malicioso foi descoberto pela propagação de uma ameaça que é detectada pelas soluções de segurança ESET como ***MSIL/TrojanDropper.Agent.FQC***, que afeta os sistemas operacionais Windows. Detectamos que a atividade desta ameaça sofreu um aumento desde o mês de setembro, com setembro e outubro sendo o pico da atividade maliciosa.

Foi possível identificar que praticamente todos os usuários alvo desse código malicioso são residentes no Brasil. Em particular, estamos falando de usuários pertencentes a setores como seguradoras, usuários domésticos e órgãos governamentais.

Além disso, conseguimos determinar a existência de mensagens predefinidas dentro do código malicioso, na língua portuguesa, bem como em diferentes sites de instituições bancárias no Brasil. Estes são os dois fatores importantes que servem para entender o motivo da forte presença desse código malicioso no Brasil. A utilidade dessas mensagens, assim como dos sites, será explicada mais adiante neste documento.

Por sua vez, vimos que este código malicioso ativo até o momento, em um período de dois meses sofreu várias modificações ou atualizações que serão mencionadas posteriormente neste relatório, demonstrando que possivelmente se trate de um código malicioso que está em estágio de desenvolvimento.

O gráfico a seguir mostra a distribuição por país para ambas as detecções, desde a segunda semana de setembro até a última semana de dezembro de 2022. Isso demonstra a forte presença dessas ameaças no Brasil, embora algumas detecções tenham sido observadas no Peru.

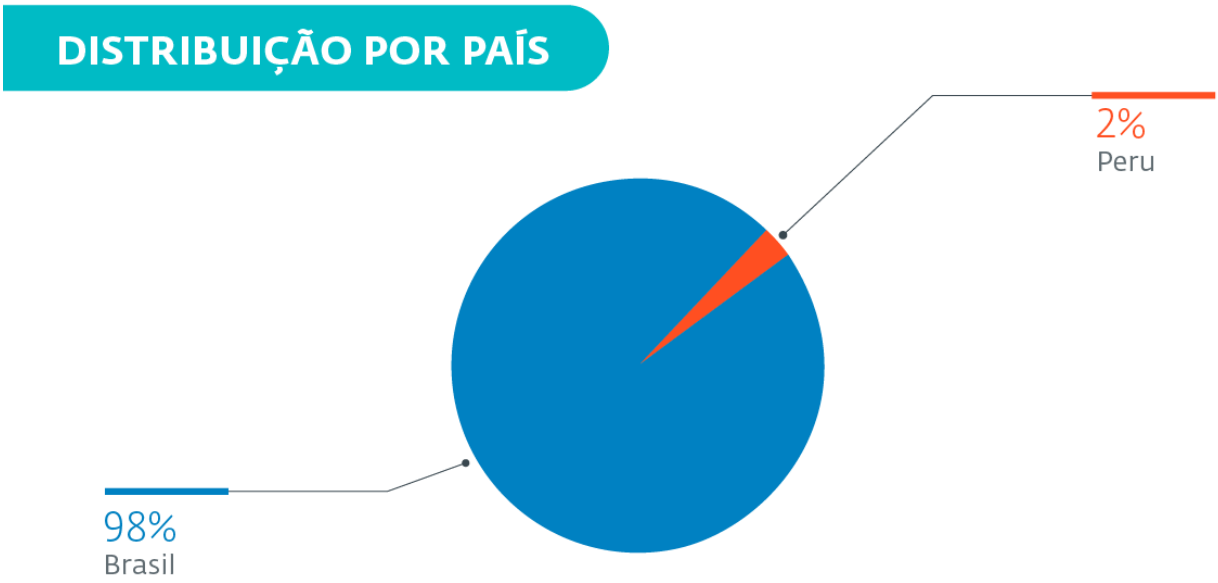

*Imagem 1. Distribuição por país com maior número de detecções na América Latina para as detecções **MSIL/TrojanDropper.Agent.FQC** e **MSIL/Spy.Banker.FN.***

No diagrama a seguir você pode ver como é o processo de infecção deste código malicioso, desde o recebimento de um e-mail que contém um arquivo malicioso compactado anexado, até chegar aos últimos

códigos maliciosos responsáveis por infectar e executar na máquina da vítima o payload final ***MSIL/Spy.Banker.FN***.

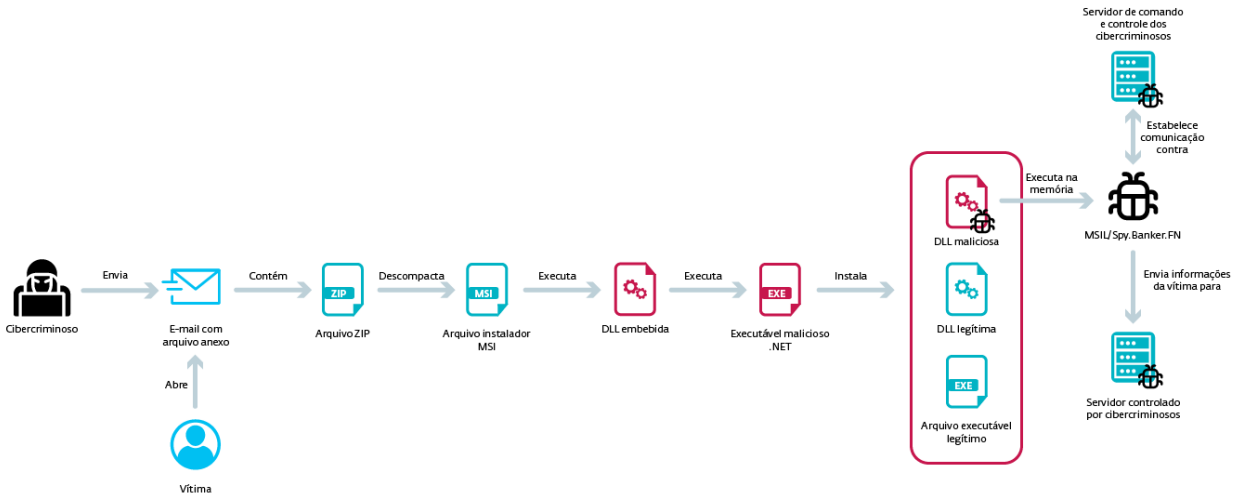

Imagem 2. Esquema de infecção.

Atividades Maliciosas

Esses códigos maliciosos têm características de dropper e trojan bancário, enquanto que a campanha que os propaga tem características de spear phishing. Por um lado, o primeiro objetivo é induzir as vítimas a baixar e executar um instalador MSI malicioso, que é compactado e anexado aos e-mails que recebem. Em seguida, este instalador malicioso começa a executar outro código malicioso responsável por "descartar" ou persistir e instalar o trojan bancário que é propagado por essa ameaça.

Em relação ao trojan bancário, ele possui características diferenciadas para roubar informações sigilosas da vítima. Alguns deles produzem uma captura de tela, registram as teclas que estão sendo pressionadas pela vítima, entre outras ações. Em seguida, estas informações são coletadas e enviadas para um servidor controlado por cibercriminosos.

Acesso inicial

Conforme mencionado anteriormente, essa ameaça chega por e-mail como um arquivo compactado malicioso anexado.

Durante a fase de análise da campanha, não foi possível obter uma amostra desses e-mails, mas conseguimos obter amostras do arquivo compactado malicioso anexado e do arquivo que ele contém.

Através de nossos sistemas internos de telemetria, conseguimos obter diferentes nomes usados nos assuntos desses e-mails. Aqui estão alguns exemplos encontrados:

- *Segue em anexo para conhecimento em | 9/14/2022 3:37:13 PM*
- *Segue em anexo documento para conhecimento em | 9/14/2022 2:01:01 PM*
- *BOLETO e NFS - 405396851302183 | 14/09/2022 14:54:23.*

Ao baixar o arquivo zip anexado, ele apresenta algo que parece ser um instalador da Microsoft, MSI, como pode ser visto na captura de tela a seguir.

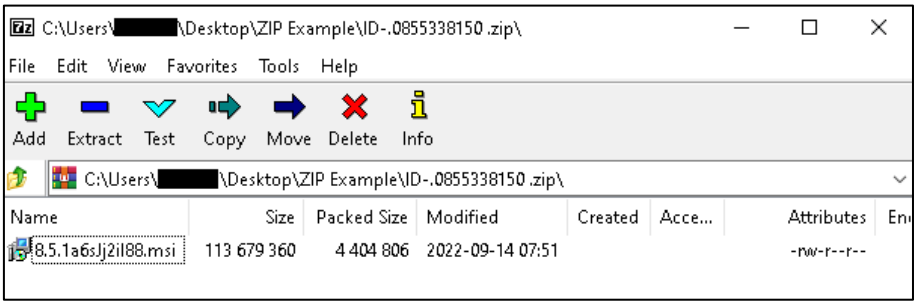

Imagem 3. Exemplo de um arquivo malicioso compactado que uma vítima pode receber.

Vimos que esses arquivos compactados não parecem ser protegidos por senha e a extensão deles pode ser "*.zip*" ou "*.rar*".

A seguir, explicaremos a atividade maliciosa realizada por este instalador MSI, quando executado por uma vítima.

Instalador MSI malicioso e seus componentes

O objetivo desta etapa é executar vários códigos maliciosos que serão instalados e executados na máquina da vítima, o trojan bancário ***Spy.Banker.FN***.

Para fazer isso, o instalador MSI malicioso possui em seus componentes uma *DLL* maliciosa que será invocada durante a execução deste instalador.

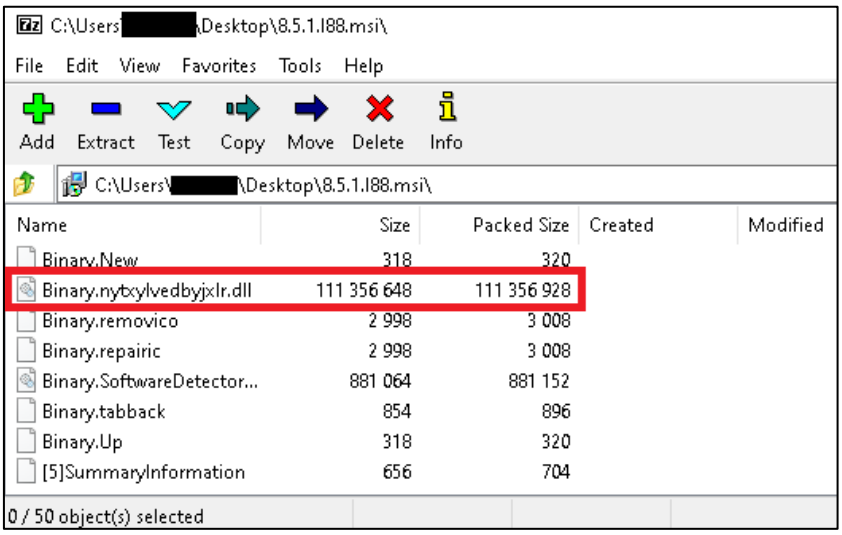

Imagem 4. DLL maliciosa incorporada no arquivo do instalador MSI.

Esta DLL maliciosa contém código malicioso desenvolvido com o *framework Microsoft .NET*, que é criptografado com o algoritmo XOR. Tanto a chave usada para descriptografar esse código malicioso quanto o próprio código malicioso serão carregados na memória pela DLL maliciosa durante sua execução.

Na captura de tela a seguir, é possível ver, como exemplo, o código malicioso carregado na memória e sua respectiva rotina de descriptografia.

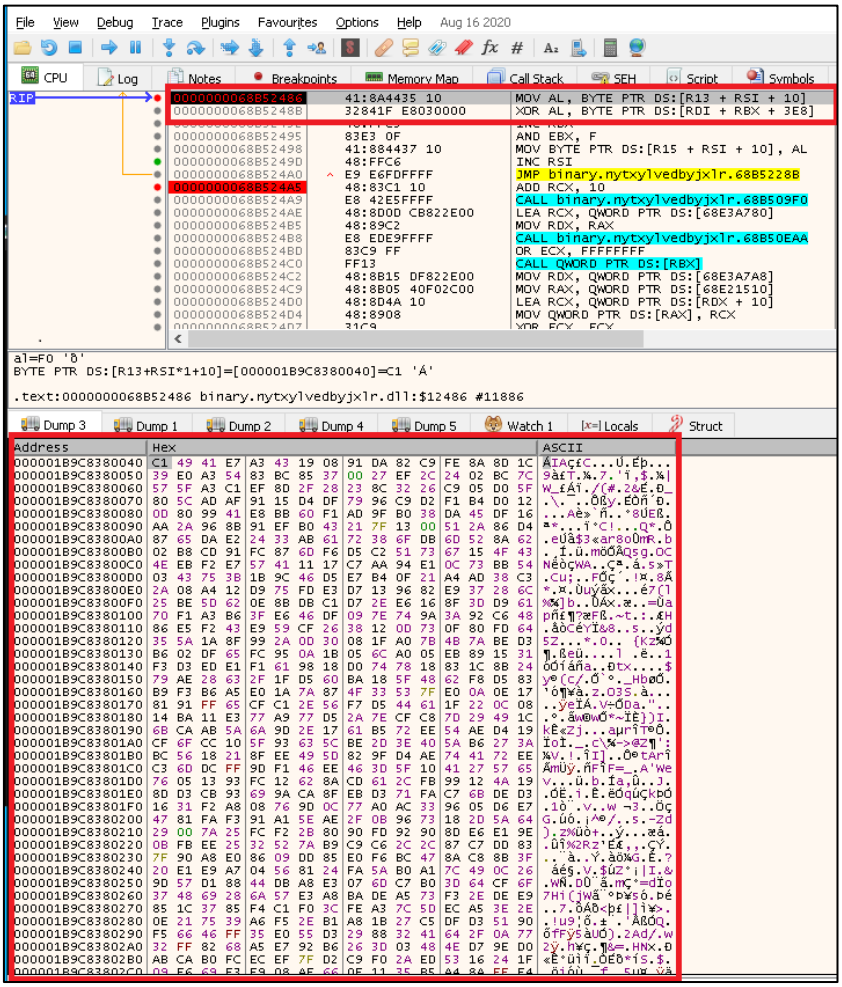

Imagem 5. Lógica usada para descriptografar o código malicioso carregado na memória.

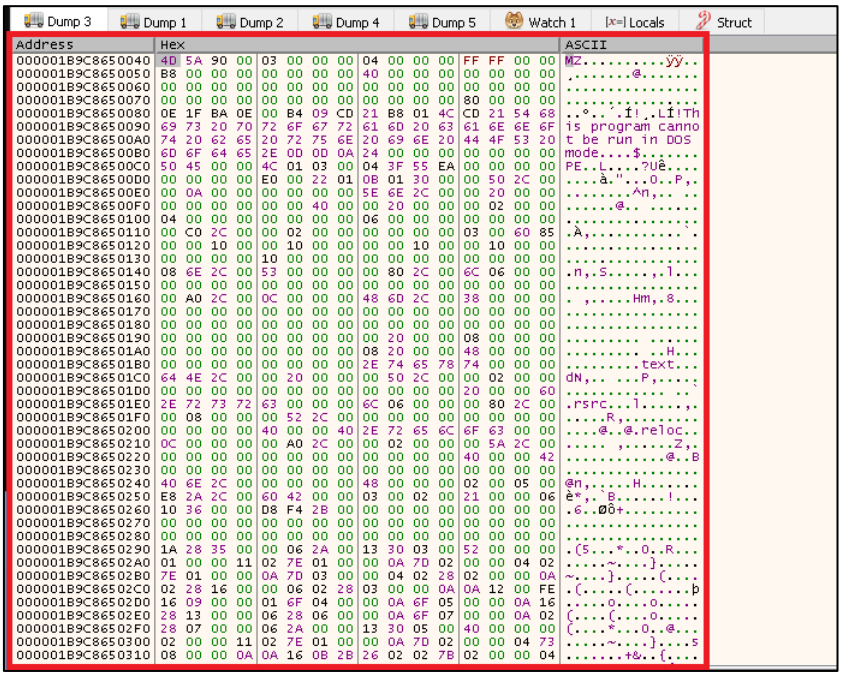

Imagem 6. Código malicioso quebrado.

Depois que a DLL termina de descriptografar o código malicioso, ela passa a executá-lo. A primeira coisa que este código malicioso faz é carregar uma janela pop-up que solicita que a vítima insira uma chave, localizada dentro de uma imagem, para abrir um suposto conteúdo em PDF.

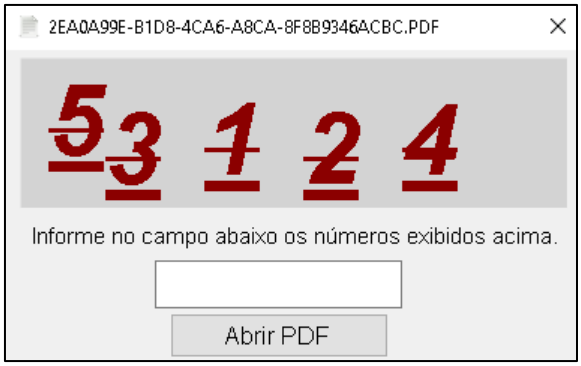

Imagem 7. Exemplo de uma janela pop-up gerada pelo código malicioso.

Uma vez que a vítima insere a chave, o código malicioso valida se realmente é a mesma mostrado na imagem. Se estiver correto, passa a executar uma função que se encarregará de instalar 3 arquivos que se encontram dentro dos recursos deste código malicioso.

Por sua vez, essa validação impede que o malware seja executado em um ambiente de sandbox automatizado, pois essa interação do usuário é necessária para ser executado.

Antes de prosseguir com a instalação, esse código malicioso verificará se a vítima está localizada no Brasil. Ele faz isso obtendo o código do país em formato de dois caracteres usando a classe RegionInfo, pertencente ao pacote de classes *System.Globalization* do *framework Microsoft .NET*, e comparando-o com um valor que tem.

Em seguida, ele cria uma pasta dentro do caminho *C:\Users\USER_NAME*, que terá um nome aleatório de 3 caracteres. Nessa pasta, o código malicioso instalará os três arquivos mencionados acima.

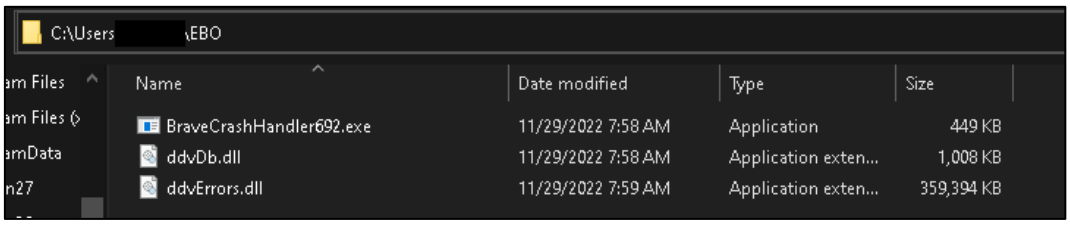

Imagem 8. Persistência na máquina da vítima.

Esses 3 arquivos consistem em:

- Um arquivo executável chamado *"BraveCrashHandler692.exe"*
- Duas DLLs chamadas *"ddvDb.dll"* e *"ddvErrors.dll"*

Após a instalação desses arquivos, o código malicioso continua invocando o arquivo executável persistente.

Por outro lado, verificou-se que muitas das cadeias de caracteres usadas por esse código malicioso, como os nomes usados para persistir os arquivos, são criptografadas com o algoritmo criptográfico *AES*. A chave usada para descriptografar suas respectivas sequências de caracteres é `75e83ed516a248f48a911a2666474d66`.

É importante mencionar que os arquivos "*BraveCrashHandle692.exe*" e "*ddvDb.dll*" são binários legítimos usados como um componente do Dell Support Center destinado a coletar informações de aplicativos desenvolvidos pela Dell. Ambos os binários são assinados digitalmente com certificados válidos emitidos pela empresa Dell.

Nesse caso, eles são usados criminalmente por cibercriminosos para executar o terceiro arquivo, "*ddvErrors.dll*", que contém o trojan bancário que será executado na máquina da vítima.

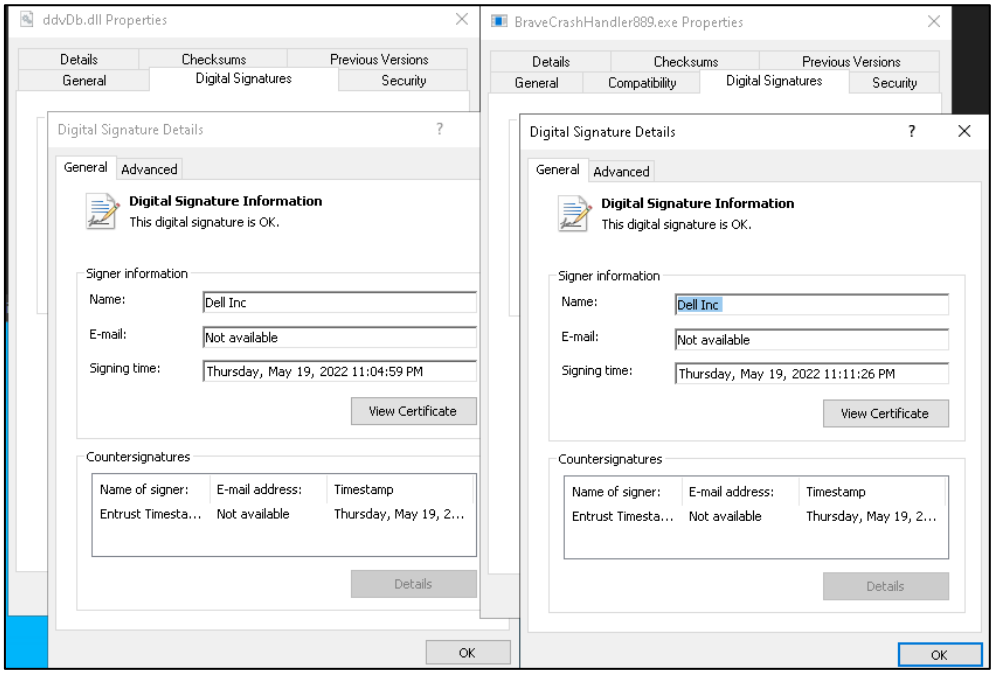

Imagem 9. Certificados digitais dos arquivos legítimos instalados pelo código malicioso.

Quando o código malicioso executa o arquivo legítimo, ele procura a presença dessas duas *DLLs* com base em seus nomes para executar, caso contrário, o arquivo exibe uma mensagem de erro.

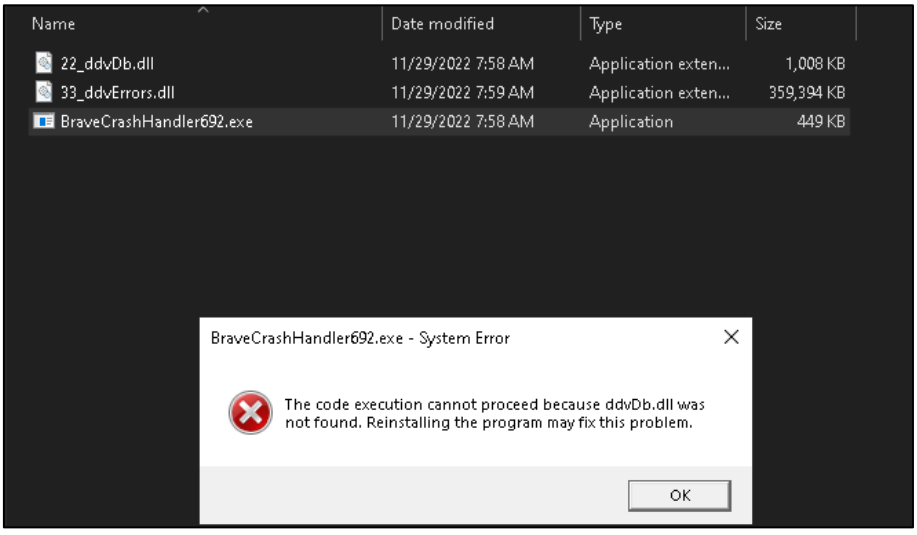

Imagem 10. Exemplo de mensagem de erro gerada pelo arquivo legítimo.

Dessa forma, os cibercriminosos delegam a execução de seu código malicioso a um arquivo legítimo, essa técnica também é conhecida como *DLL Side-Loading*.

Depois que a DLL maliciosa "*ddvErrors.dll*" é executada, ela descriptografa o payload *Spy.Banker.FN*, que será injetada no processo *explorer.exe* usando a técnica de injeção de processo.

Por outro lado, durante a fase de análise foi possível detectar que o instalador *MSI* malicioso tinha verificações para detectar se está sendo executado em uma máquina virtual.

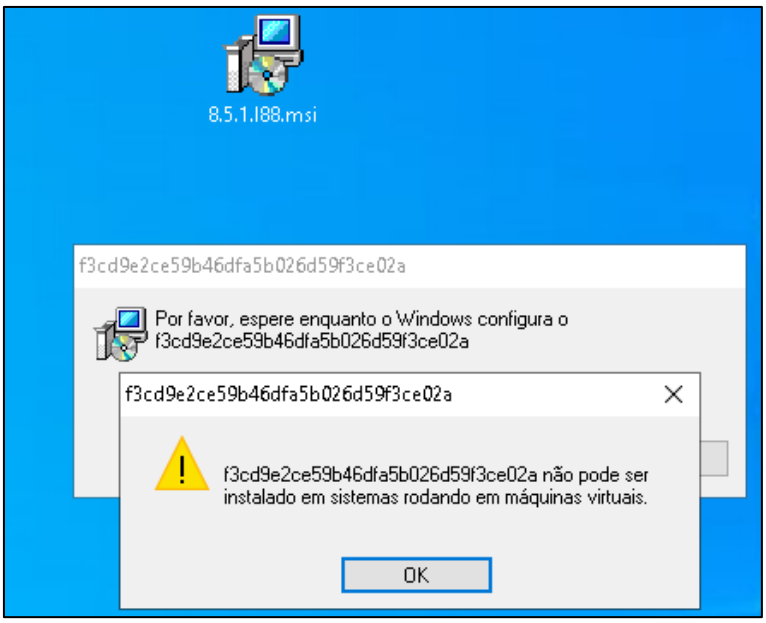

Imagem 11. Mensagem de alerta do instalador MSI malicioso detectando que está sendo executado em uma máquina virtual.

Spy.Banker.FN

O Spy.Banker.FN é um trojan bancário desenvolvido com o framework Microsoft .NET, que visa roubar informações financeiras confidenciais da vítima. Para isso, utiliza técnicas como capturas de tela ou a criação de falsas janelas pop-up, fazendo-se passar por diferentes instituições, algumas delas bancos, para induzir a vítima a inserir dados sensíveis, como senhas ou tokens de segurança, nessas janelas pop-up , que então enviam essas informações para um servidor controlado pelos cibercriminosos.

Esse código malicioso primeiro executa uma validação nos registros do Windows da máquina da vítima para garantir que o código malicioso seja executado assim que o sistema for inicializado. Para fazer isso, ele verificará a existência de uma entrada que tenha o caminho onde o código malicioso foi instalado no seguinte caminho no registro do Windows `HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run`.

Se esta entrada não existir nos registros do Windows, o código malicioso criará uma nova entrada cujo nome será caracteres aleatórios com comprimento variável entre 7 e 21 caracteres.

A captura de tela a seguir mostra um exemplo da chave criada pelo código malicioso nos registros do Windows.

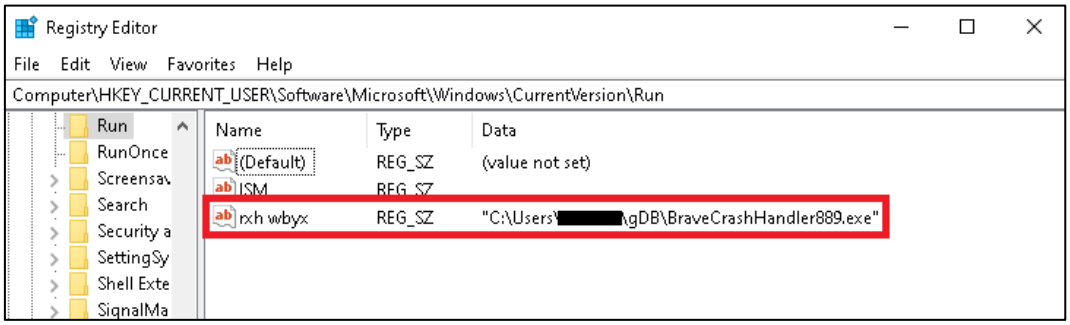

Imagem 12. Criação de chave no registro do Windows do Spy.Banker.FN.

Caso essa chave precise ser criada, o código malicioso enviará as seguintes informações para uma URL maliciosa controlada por cibercriminosos:

- Nome do computador e nome de usuário;
- Nome completo do sistema operacional;
- Versão do código malicioso;
- Lista de aplicativos antifraude instalados na máquina da vítima.

Para obter a lista de aplicativos antifraude, o código malicioso pesquisa entre os aplicativos em execução, bem como arquivos executáveis em caminhos específicos para determinar a presença de algum dos seguintes aplicativos:

- Aplicativo Itaú;
- Navegador Exclusivo Bradesco;
- Warsaw Bank Software;
- IBM Rapport Security Software.

Antes de serem enviadas, as informações são codificadas em base64 e colocadas dentro de um objeto JSON, que irá no corpo da requisição feita à URL.

A partir da amostra analisada, foi possível obter a seguinte URL maliciosa utilizada pelos cibercriminosos para enviar as informações descritas acima.

- http[:]//counter02.udurnfdsbferruwewrucont2.xyz/{Guid.NewGuid().ToString()}/{Guid.NewGuid().ToString()}/count

> *NOTA: A URL maliciosa utilizada varia de acordo com a amostra. Ao mesmo tempo, acreditamos que os cibercriminosos queriam colocar um identificador GUID na URL, mas devido a um problema de programação no código, a URL permaneceu como mostrada acima.*

A captura de tela a seguir mostra uma captura do tráfego gerado ao enviá-lo para os cibercriminosos.

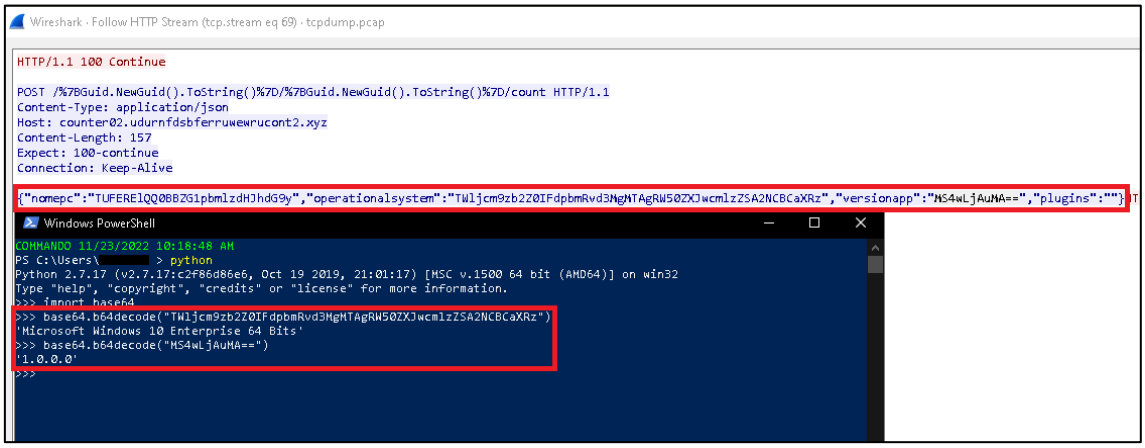

Imagem 13. Exemplo de informação enviada para uma URL maliciosa controlada pelos cibercriminosos.

Em seguida, esse código malicioso cria uma Tarefa encarregada de monitorar a atividade da vítima e se comunicar com o servidor controlado pelos cibercriminosos.

Para fazer isso, ele cria um loop infinito onde obtém a janela ativa que a vítima está usando naquele momento e tenta obter o endereço URL, se existir. Dessa forma, se a vítima estiver visitando um site, o código malicioso pode obter a URL do site em questão.

Depois de obter a URL, ele compara seu domínio contra vários domínios que possuem o código malicioso. Esta série de domínios está relacionada a entidades bancárias ou sites de Exchange de criptomoedas usados no Brasil. Alguns exemplos estão listados abaixo:

- bancobrasil.com.br;
- gerenciador.caixa.gov.br;
- mercadobitcoin.com.br;
- itau.com.br.

> *NOTA: Para obter informações mais detalhadas sobre todos os domínios que este código malicioso possui, confira a Lista completa de domínios.*

Caso tenha encontrado um domínio de seu interesse, ele salvará em uma variável um número que pode ser entre 0 ou 9 usado para determinar o domínio que é visitado pela vítima. O código malicioso se comunicará com o servidor de comando e controle gerando uma comunicação TCP usando uma biblioteca de código aberto conhecida como *WatsonTCP*. Para estabelecer a comunicação, o código malicioso utilizará um certificado SSL, que é criado com base em um array de bytes que possui nos recursos e uma senha que é a seguinte: `5ed5f495c80d567d0e8823603e579b16`.

As informações enviadas ao servidor de comando e controle são as seguintes:

- Nome da máquina e nome de usuário da vítima;
- Nome completo do sistema operacional e sua arquitetura;
- Número da versão do código malicioso;
- Domínio do servidor de comando e controle utilizado;
- Número que identifica o tipo de domínio visitado pela vítima.

A captura de tela a seguir mostra parte da lógica usada pelo código malicioso para atribuir o valor à variável de acordo com o domínio visitado pela vítima.

```
StringBuilder stringBuilder = new StringBuilder(255);
Program.GetWindowText(foregroundWindow, stringBuilder, 255);
if (stringBuilder.Length != 0)
{
    string text = stringBuilder.ToString();
    if (!string.IsNullOrEmpty(text))
    {
        string text2 = Program.get_url_from_app(foregroundWindow, text.ToLower());
        if (!string.IsNullOrEmpty(text2))
        {
            string a = Regex.Match(text2, Program.decrypt_n_get_string(25)).Groups[1].Value.ToLower();
            bool flag = false;
            if (a == "bancobrasil.com.br"){
                Program.bank_url_type_ID = 0;
                flag = true;
            }else if (a == "autoatendimento.bb.com.br"){
                Program.bank_url_type_ID = 0;
                flag = true;
            }else if (a == "www2.bancobrasil.com.br"){
                Program.bank_url_type_ID = 0;
                flag = true;
            }else if (a == "internetbanking.caixa.gov.br"){
                Program.bank_url_type_ID = 1;
                flag = true;
            }else if (a == "gerenciador.caixa.gov.br"){
                Program.bank_url_type_ID = 1;
                flag = true;
            }else if (a == "loginx.caixa.gov.br"){
                Program.bank_url_type_ID = 1;
                flag = true;
            }
            else if (a == "banco.bradesco"){
                Program.bank_url_type_ID = 2;
                flag = true;
            }else if (a == "cidadetran.bradesco"){
                Program.bank_url_type_ID = 2;
                flag = true;
            }else if (a == "ne12.bradesconetempresa.b.br"){
                Program.bank_url_type_ID = 2;
                flag = true;
            }else if (a == "binance.com"){
                Program.bank_url_type_ID = 3;
                flag = true;
            }else if (a == "mercadobitcoin.com.br"){
                Program.bank_url_type_ID = 3;
                flag = true;
            }else if (a == "bitcointrade.com.br"){
```

Imagem 14. Lógica usada pelo código malicioso para atribuir um número com base no site visitado pela vítima.

Uma vez realizada a comunicação com o servidor, este devolverá ao código malicioso uma cadeia de caracteres que determina a ação maliciosa a realizar. Esta string de caracteres é separada em uma lista por meio de uma expressão regular que usa 9 dígitos consecutivos que aparecem na string como separador. Veja a seguir um exemplo de como o código malicioso pode receber uma string de caracteres:

- `FakeWindowsImpersonationSSS999555444Locked999555444ITA`

Depois que a string de caracteres é transformada em uma lista, o comprimento do primeiro elemento dessa lista determina a ação que o código malicioso executará na vítima.

Na tabela a seguir, você pode ver todas as funcionalidades que o código malicioso é capaz de fazer com base no comprimento do primeiro elemento da lista.

Longo	Descrição
10	Desconectar do servidor de comando e controle
11	Termine sua corrida
12	Faça uma captura de tela da janela ativa e envie-a para o servidor de comando e controle
13	Envie informações sobre quais janelas estão visíveis para o servidor de comando e controle
14	Restaurar e maximizar uma janela de entrada, usando as APIs SetForeGroundWindow e ShowWindow Windows
15	Minimizar uma janela usando a API ShowWindow Windows
16	Restaurar uma janela de entrada, usando as APIs SetForeGroundWindow e ShowWindow Windows
17	Encerre um processo usando o valor do parâmetro MainWindowHandler
18	Alterar a posição de uma janela
19	Salve o ponteiro relacionado a uma janela em uma variável
20	Esvazie a variável que armazena o ponteiro
21	Faça logoff da vítima executando o comando “shutdown -l”
22	Ativando a composição do Desktop Window Manager por meio dos registros do Windows
23	Alterar a posição do cursor
24	Envie uma string para a janela ativa usando a API SendWait
25	Semelhante ao anterior, mas valida se a string de caracteres a enviar está em maiúsculas ou minúsculas
26	Desative a composição do Desktop Window Manager usando a API do Windows DwnEnableComposition
27	Capaz de realizar dois tipos de representação: 1. Crie uma tela falsa de atualização no Windows Update ou em um banco 2. Representar um aplicativo específico para roubar uma senha ou um token de segurança da vítima
28	Limpe variáveis e destrua janelas criadas por códigos maliciosos
30	Alterar a opacidade de uma janela
31	Keylogging

Tabela 1. Funcionalidades que o código malicioso possui.

Como pode ser visto na tabela acima, este código malicioso possui um grande número de funcionalidades específicas para geração de janelas pop-up. Algumas dessas janelas têm a capacidade de bloquear o uso da máquina da vítima até que a atividade maliciosa termine ou os cibercriminosos o indiquem enviando um comando.

Abaixo está uma parte do código usado para a criação de um dos falsos pop-ups.

```
}
else if (app_to_impersonate == "ITA")
{
    LinearGradientBrush linearGradientBrush = new LinearGradientBrush();
    linearGradientBrush.StartPoint = new System.Windows.Point(0.5, 0.0);
    linearGradientBrush.EndPoint = new System.Windows.Point(0.5, 1.0);
    GradientStop gradientStop = new GradientStop();
    gradientStop.Color = new SolidColorBrush(
        (System.Windows.Media.Color)System.Windows.Media.ColorConverter.ConvertFromString(Program.decrypt_n_get_string(122))).Color;
    gradientStop.Offset = 0.9;
    linearGradientBrush.GradientStops.Add(gradientStop);
    GradientStop gradientStop2 = new GradientStop();
    gradientStop2.Color = new SolidColorBrush(
        (System.Windows.Media.Color)System.Windows.Media.ColorConverter.ConvertFromString(Program.decrypt_n_get_string(122))).Color;
    gradientStop2.Offset = 1.0;
    linearGradientBrush.GradientStops.Add(gradientStop2);
    grid.Background = linearGradientBrush;
    streamSource2 = new MemoryStream(Convert.FromBase64String(Program.decrypt_n_get_string(123)));
    rectangle.Height = 50.0;
    rectangle.Width = 50.0;
    rectangle.Margin = new Thickness(30.0, 12.0, 0.0, 0.0);
}
else if (app_to_impersonate == Program.decrypt_n_get_string(124))
{
    LinearGradientBrush linearGradientBrush = new LinearGradientBrush();
    linearGradientBrush.StartPoint = new System.Windows.Point(0.5, 0.0);
    linearGradientBrush.EndPoint = new System.Windows.Point(0.5, 1.0);
    GradientStop gradientStop = new GradientStop();
    gradientStop.Color = new SolidColorBrush((System.Windows.Media.Color)System.Windows.Media.ColorConverter.ConvertFromString(Program.decry
    gradientStop.Offset = 0.9;
```

Imagem 15. Lógica usada para criar um pop-up falso.

Aqui estão dois exemplos de pop-ups falsos que o código é capaz de criar.

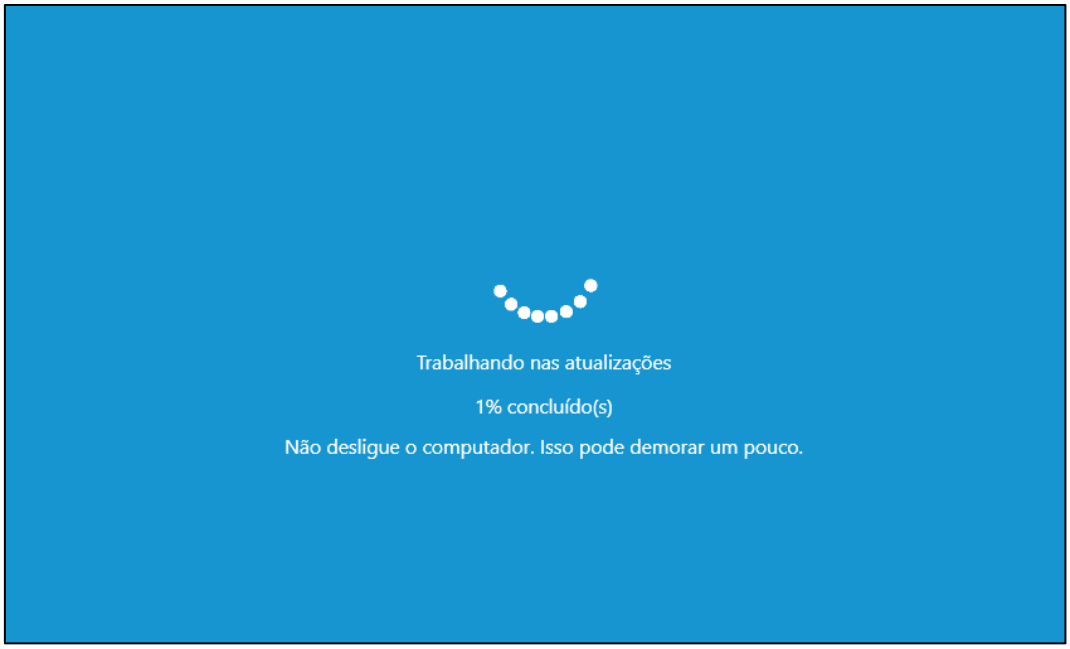

Imagem 16. Exemplo de pop-up falso simulando uma atualização do Windows.

Imagem 17. Exemplo de uma janela pop-up falsa simulando uma janela de aplicativo bancário.

NOTA: Para ver mais exemplos dos diferentes tipos de janelas pop-up que este código malicioso pode criar, confira o anexo Exemplos de telas falsas.

Com base na análise realizada em uma das amostras desse trojan bancário, observou-se que o código malicioso é capaz de se passar pelas seguintes entidades:

- Banco BRADESCO;
- Banco ITAU;
- Banco Brasil;
- Banco Santander;
- Banco do Nordeste;
- Banco Sicredi;
- Institución financiera CAIXA;
- Aplicación Trusteer Raptor;
- MercagoPago;
- Binance.

Um fato importante a ser mencionado é que muitas das strings de caracteres usadas pelo código malicioso são criptografadas com o algoritmo criptográfico AES usando a seguinte chave `4ca756ab6f9b4b2ebe21c42ffdbdcc88`.

Em seguida, mencionamos diferentes tipos de cadeias de caracteres que são criptografadas:

- Senha para o certificado x509;
- Caminho da chave de registro usado para persistência de ameaças;
- Aplicativos antifraude para busca na máquina da vítima;
- Domínio e porta usados para se comunicar com o servidor de comando e controle;
- Domínios de entidades bancárias e serviços de câmbio para pesquisa na janela utilizada pela vítima;
- Expressões regulares para manipular a resposta do servidor controlado pelos atacantes;
- Imagens, cores e textos usados para a criação de pop-ups falsos.

Evolução do Spy.Banker.FN

Durante a etapa de análise, foi possível observar que o trojan bancário teve várias mudanças em seu comportamento durante o período de setembro ao final de novembro de 2022.

A linha do tempo a seguir mostra as versões com as alterações mais significativas e a data em que foram detectadas por nossas soluções de segurança.

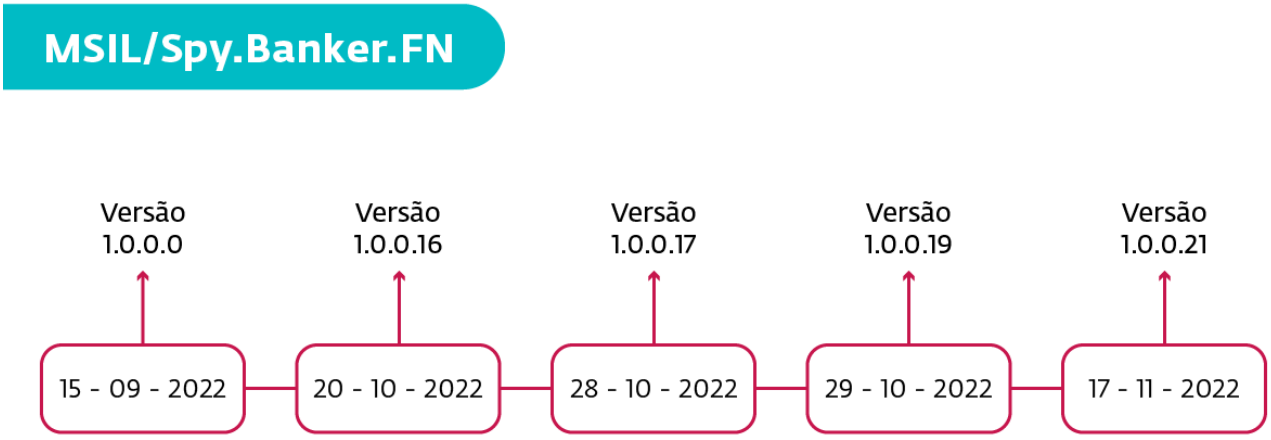

Imagem 18. Linha do tempo mostrando a evolução do Spy.Banker.FN

Como pode ser visto no gráfico acima, este código malicioso sofreu muitas modificações em um curto período de tempo, o que nos dá indícios de que os cibercriminosos ainda estão trabalhando nele e ainda não possuem uma versão final.

A tabela a seguir, ordenada por versão, mostra as mudanças mais significativas que o código malicioso sofreu em cada versão.

Versión	Descrição
1.0.0.0	Comportamento descrito neste documento.
1.0.0.16	• Adicionado novas sequências de caracteres, como domínios para detectar na máquina da vítima. • É definido com base nos processos em execução na vítima, se o código malicioso já estiver em execução. • A função número 12, responsável por fazer capturas de tela, agora tem toda a sua lógica encapsulada em uma tarefa. • Valida se o Aplicativo ITAU está rodando na máquina da vítima e, caso esteja, fecha-o. • Agora o código malicioso persiste em uma DLL chamada *libiomp5md.dll*. Esse nome é usado ao validar a entrada nos logs do Windows para persistência. • Na funcionalidade principal do código malicioso, a seguinte linha *Console.WriteLine(ex)* foi adicionada em vários lugares. É provável que tenha sido usado em estágios de teste de malware, pois imprime o valor de uma exceção.
1.0.0.17	• Adiciona-se uma nova funcionalidade número 32: modifica o valor de uma variável booleana chamada *DateChanged*. Se definido como verdadeiro, fecha a janela ativa da vítima usando a *API SendMesage.*

1.0.0.19	• A alteração feita na versão 1.0.0.17 foi removida.
	• Novos recursos são adicionados:
	  ○ Funcionalidade 33: Enviar chaves específicas (UP, DOWN, LEFT, RIGHT) para a janela ativa da vítima através da API SendWat
	  ○ Funcionalidade 34: Modifica a resolução da tela da vítima para 1920x1080 e 2560x1440 usando a API ChangeDisplaySettings

1.0.0.21	• Removida a linha Console.WriteLine(ex) do código malicioso • A funcionalidade 34 é modificada, a resolução a ser modificada pela API *ChangeDisplaySettings* é obtida usando a seguinte consulta: *SELECT * FROM CIM_VideoControllerResolution.*

Tabela 2. Evolução do Spy.Banker.FN e suas mudanças mais significativas.

Conclusão

*O **Spy.Banker.FN*** é um trojan bancário que está surgindo no Brasil. Considerando o número de versões diferentes que vimos em tão pouco tempo, podemos perceber que os cibercriminosos ainda não contam com uma versão final da ameaça, mas continuam tentando obter um código malicioso final que sirva para realizar suas atividades criminosas.

Assim como o trojan bancário documentado em 2021 *Janeleiro*, ***Spy.Banker.FN*** também foi desenvolvido sob a mesma estrutura do *Microsoft .NET.* Essa é uma clara diferença em relação a outros trojans bancários vistos no Brasil, onde geralmente são desenvolvidos na linguagem de programação *Delphi*, como é o caso do *Mekotio*.

Apesar do *Janeleiro* e do ***Spy.Banker.FN*** contarem com certas semelhança no uso de APIs do Windows para criar janelas pop-up falsas ou no uso do algoritmo de criptografia AES para descriptografar strings de caracteres. Nenhuma semelhança ou relação significativa foi encontrada para levar à conclusão de que pode haver uma ligação direta entre os dois ou que o ***Spy.Banker.FN*** pode se tornar uma nova versão do Janeleiro.

Indicadores de comprometimento

Registros

As chaves de registro do Windows manipuladas pelos códigos maliciosos descritos neste relatório estão listadas abaixo:

- HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
    - <NOME_ALEATORIO> – Rota onde a ameaça persiste

Persistência

A seguir estão os caminhos usados pelo código malicioso para persistir na máquina da vítima:

- C:\Users\Public\<NOME_ALEATORIO>
- C:\Users\NOME_USUARIO\<NOME_ALEATORIO>

> *NOTA:<NOME_ALEATORIO> é composto por uma string de caracteres escolhidos aleatoriamente com um comprimento que pode variar entre 3 a 21 caracteres.*

Hashes, URLs e C&C

Hashes de amostras analisadas:

- 7BB2E25286C3B41D6F7C20D60CDA076AD87CEA77 – MSIL/TrojanDropper.Agent.FQC
- C59D0EDCC7E4CACA877074978B5F0CABC7631500 – MSIL/TrojanDropper.Agent.FQC
- EF0AF6134A4A3B76ED33CF60444907B66322B829 – MSIL/TrojanDropper.Agent.FQC
- 4611351CE5FEBC948E5801C9B91109F5FE75FF3C – Win64/ShellcodeRunner.DG
- ACB65446DB497ECC6E29279D67EE888E2722D821 – MSIL/Spy.Banker.FN
- F3665697AC2D53E1E8F20282F0C64CF4FA23C2F0 – MSIL/Spy.Banker.FN
- 594B89D088A275339238203ABAD7E4D7375F00F6 – MSIL/Spy.Banker.FN
- 67FB2845B3129AC07468125ED778DFB35D8E995F – MSIL/Spy.Banker.FN
- 2AB73029397815B5102E883F09797CE958F06688 – MSIL/Spy.Banker.FN

Domínios e IPs detectados nas amostras analisadas:

- counter02[.]udurnfdsbferruwewrucont2.xyz
- atacadao-oleo[.]com.br
- intimidadesegura[.]com.br
- serenatadenoticias[.]online
- carvalhomeloefrancoeireli[.]com
- moraesnegocios[.]com
- hotbarclube[.]online
- mundodostrecos[.]online
- postodeciolima[.]com
- sociomarcos[.]online
- postodemolasve[.]online
- 206[.]72.192.90
- 66[.]42.103.250
- 20[.]21.108.67
- 192[.]124.216.211
- 34[.]70.159.185
- 200[.]98.113.122

- 74[.]50.81.133
- 45[.]76.35.50
- 70[.]34.199.17

Dicas para se proteger

Como essa ameaça é distribuída por e-mails que contêm arquivos compactados maliciosos anexados, listamos várias recomendações a serem lembradas para evitar se tornar uma vítima em potencial:

- Verifique o e-mail, preste atenção a:
  - O endereço de onde vem.
  - O nome da pessoa que o enviou.
  - Observe o conteúdo da mensagem, por exemplo, se há erros de ortografia.
- Não abra nenhum e-mail se houver motivos para duvidar do conteúdo ou de quem o enviou.
- Não baixe anexos de e-mail se tiver dúvidas sobre o recebimento ou qualquer outro detalhe.
- Observe as extensões dos arquivos, por exemplo, se um arquivo terminar com "*.pdf.exe*" a última extensão é a que determina o tipo de arquivo, neste caso seria "*.exe*" um executável.
- Se um e-mail tiver um link e caso você considere a página de redirecionamento suspeita, não a abra.
- Seja cauteloso ao baixar e extrair arquivos *.zip/.bz2* de fontes não confiáveis, pois eles costumam ser utilizados para ocultar códigos maliciosos e burlar certos mecanismos de segurança.
- Atualize os seus dispositivos e aplicativos com a versão mais recente.
- Mantenha as soluções de segurança instaladas no dispositivo atualizadas.
- Observe com bastante atenção qualquer atividade incomum nos aplicativos de instituições bancárias que você usa. Se houver qualquer atividade suspeita, encerre imediatamente a execução do aplicativo e análise o dispositivo com um produto de segurança.

Anexo

Exemplos de telas falsas

As capturas de tela a seguir mostram diferentes exemplos de pop-ups falsos que o ***Spy.Banker.FN*** é capaz de criar.

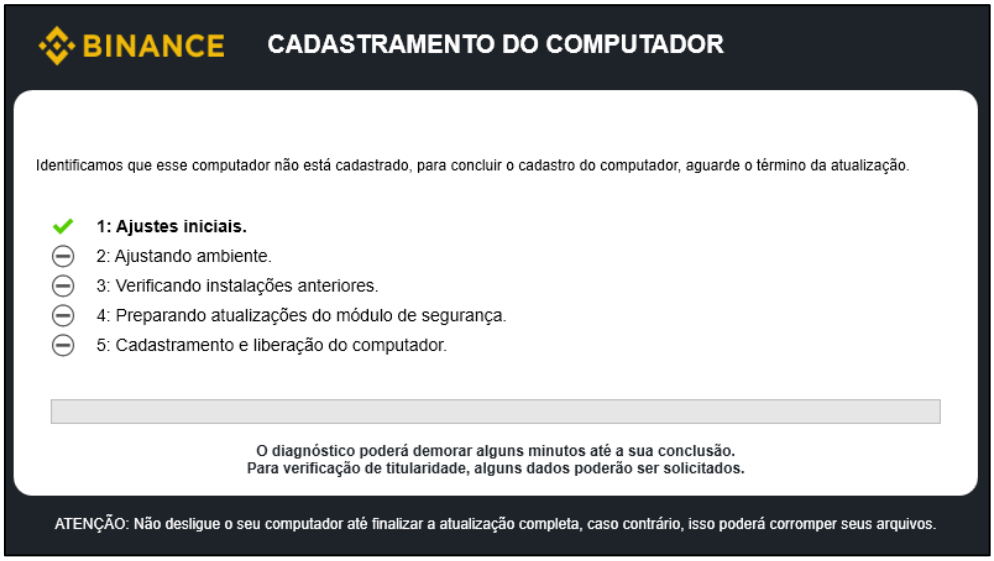

Imagem 19. Exemplo de um pop-up falso exibindo uma suposta atualização representando um site de Exchange de criptomoedas.

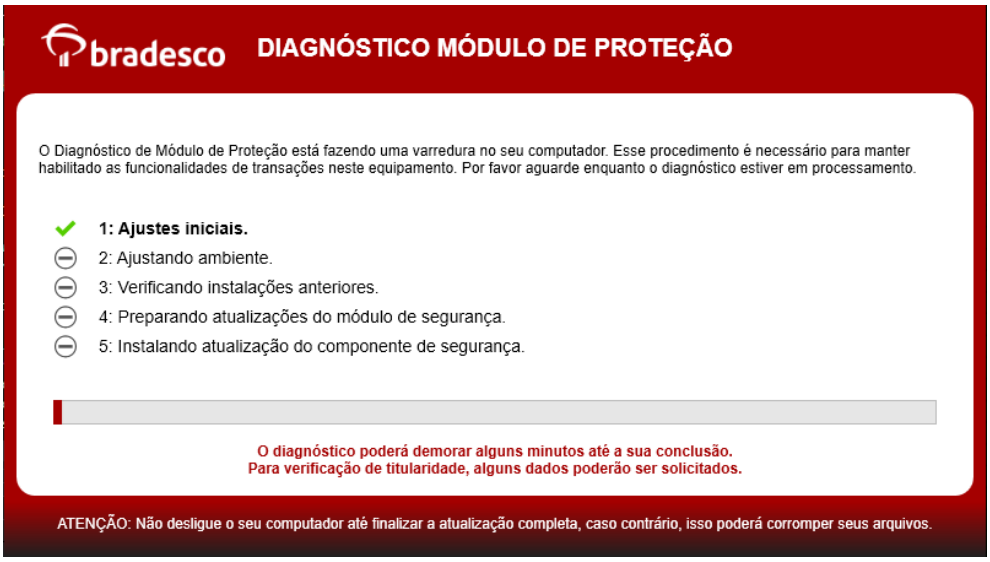

Imagem 20. Exemplo de um pop-up falso exibindo uma suposta atualização se passando pelo Banco Bradesco.

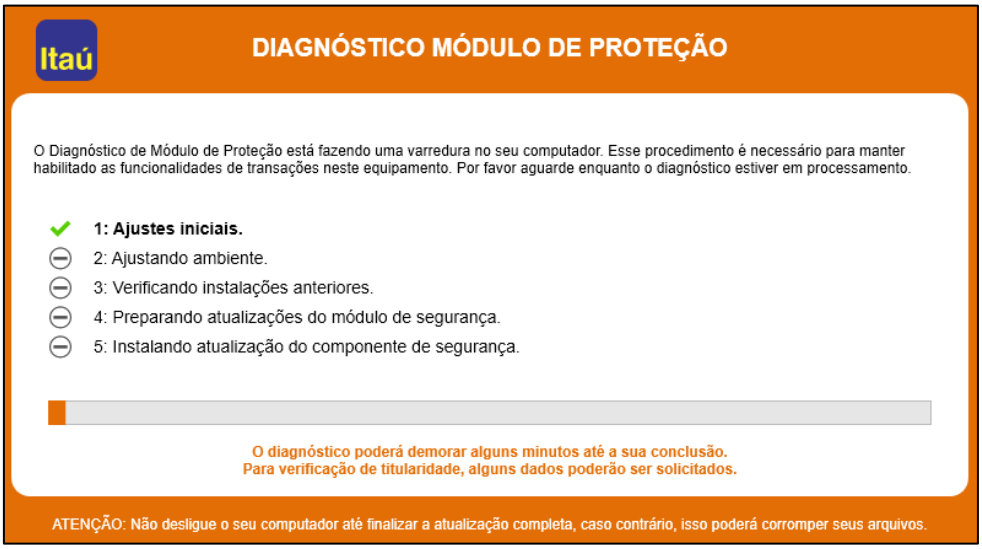

Imagem 21. Exemplo de pop-up falso mostrando uma suposta atualização se passando pelo Banco ITAU.

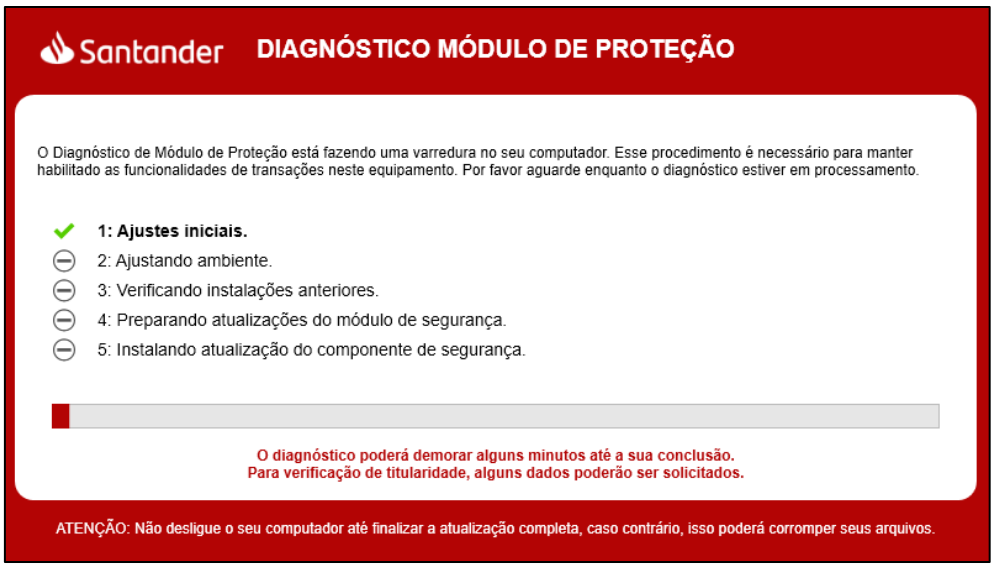

Imagem 22. Exemplo de um pop-up falso exibindo uma suposta atualização se passando pelo Banco Santander.

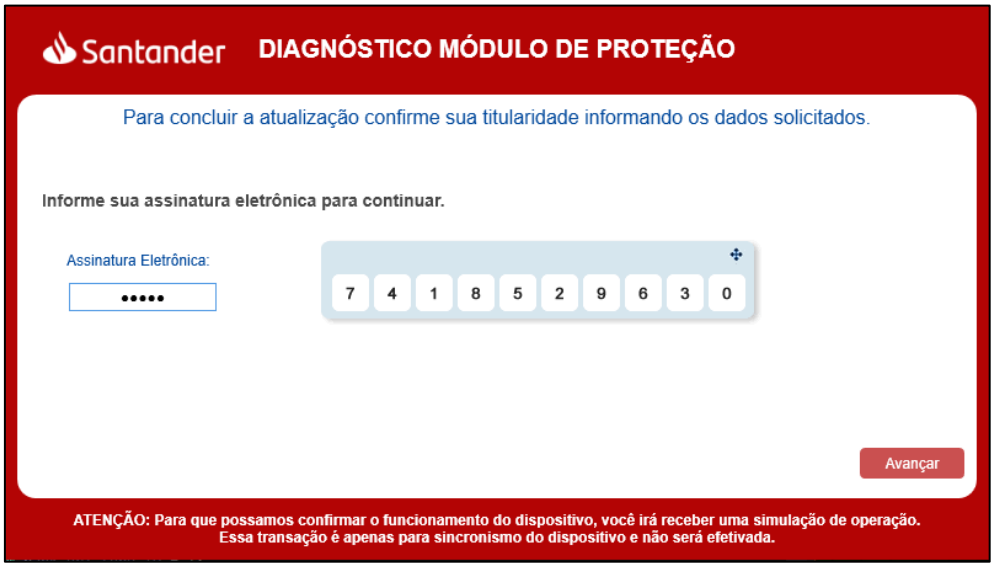

Imagem 23. Exemplo de um pop-up falso fazendo-se passar pelo Banco Santander pedindo uma assinatura eletrônica.

Imagem 24. Exemplo de um pop-up falso se fazendo passar pelo Mercado Pago e pedindo que a vítima digite o código do token de segurança física.

Imagem 25. Exemplo de pop-up falso passando pelo Mercado Pago pedindo a entrada do código QR.

Imagem 26. Exemplo de um pop-up falso se fazendo passar pelo Mercado Pago e pedindo que a vítima digite o código do token de segurança física.

Imagem 27. Exemplo de um pop-up falso se fazendo passar pelo Mercado Pago e pedindo que a vítima digite o código de segurança do seu celular.

Imagem 28. Exemplo de um pop-up falso se fazendo passar pelo Banco do Brasil e pedindo que a vítima digite o código de verificação que foi enviado para o celular.

Imagem 29. Exemplo de um pop-up falso se fazendo passar pelo Banco do Brasil e pedindo que a vítima digite o código de segurança do token físico.

Imagem 30. Exemplo de um pop-up falso se fazendo passar pelo Banco de Nordeste e pedindo que a vítima digite o código QR.

Imagem 31. Exemplo de um pop-up falso se fazendo passar pelo Banco Sicredi e pedindo que a vítima digite o código de verificação que foi enviado para o celular.

Imagem 32. Exemplo de um pop-up falso se fazendo passar pelo Trusteer Rapport e pedindo que a vítima digite o código de verificação que foi enviado para o celular.

Lista completa de domínios

Todos os domínios que *Spy.Banker.FN* está procurando estão listados abaixo:

bancobrasil.com.br	autoatendimento.bb.com.br	www2.bancobrasil.com.br
gerenciador.caixa.gov.br	loginx.caixa.gov.br	banco.bradesco
ne12.bradesconetempresa.b.br	binance.com	mercadobitcoin.com.br
electrum	foxbit.com.br	blockchain.com
pf.santandernet.com.br	pj.santandernetibe.com.br	itau.com.br
internetbanking.bancointer.com.br	contadigital.bancointer.com.br	meu.original.com.br
ibpj.original.com.br	banrisul.com.br	internetbanking.banpara.b.br
www2s.bancoamazonia.com.br	ecode.daycoval.com.br	mercantildobrasil.com.br
bancopan.com.br	unicred.com.br	safra.com.br
ib.brde.com.br	banese.com.br	bancobmg.com.br
internetbanking.confesol.com.br	tribanco.com.br	credisisbank.com.br
bancobs2.com.br	bancofibra.com.br	uniprimebr.com.br
bancotopazio.com.br	btgmais.com	citidirect.com
zeitbank.com.br	cora.com.br	sofisa.com.br
sicredi.com.br	nel.bnb.gov.br	mercadopago.com.br
banestes.com.br	wwws.uniprimedobrasil.com.br	www.rendimento.com.br
contaonline.viacredi.coop.br	internetbanking.caixa.gov.br	ib.banpara.b.br
accounts.binance.com	cidadetran.bradesco	stone.com.br
contadigitalpj.bancointer.com.br	bitcointrade.com.br	safraempresas.com.br
empresas.original.com.br	banestes.b.br	brbbanknet.brb.com.br
credisan.com.br	sofisadireto.com.br	www.banestes.com.br
uniprime.com.br		rendimento.com.br

Técnicas MITRE ATT&CK

As técnicas MITRE ATT&CK observadas nas amostras analisadas estão listadas abaixo:

Tática	Técnica (ID)	Nome
Initial Access	T1566.001	Phishing: Spearphishing Attachment
Execution	T1204.002	User Execution: Malicious File
	T1106	Native API
	T1047	Windows Management Instrumentation
Persistence	T1547.001	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder
Defense Evasion	T1055	Process Injection
	T1027	Obfuscated Files or Information
	T1497	Virtualization/Sandbox Evasion
	T1622	Debugger Evasion
	T1574.002	Hijack Execution Flow: DLL Side-Loading

Tática	Técnica (ID)	Nome
Discovery	T1012	Query Registry
	T1614.001	System Location Discovery: System Language Discovery
	T1033	System Owner/User Discovery
	T1057	Process Discovery
	T1083	File and Directory Discovery
	T1082	System Information Discovery
	T1010	Application Window Discovery
Collection	T1113	Screen Capture
	T1056.001	Input Capture: Keylogging
	T1056.002	Input Capture: GUI Input Capture
Command and Control	T1573.002	Encrypted Channel: Asymmetric Cryptography
	T1132.001	Data Encoding: Standard Encoding
	T1132.002	Data Encoding: Non-Standard Encoding
Exfiltration	T1041	Exfiltration Over C2 Channel
	T1048.003	Exfiltration Over Alternative Protocol: Exfiltration Over Unencrypted Non-C2 Protocol

Créditos

Autor

Fernando Tavella

Malware Researcher