BOLETIM DE SEGURANÇA

Novo malware macOS TodoSwift ligado a hackers Norte-Coreanos

heimdall
security research
A DIVISION OF ISH

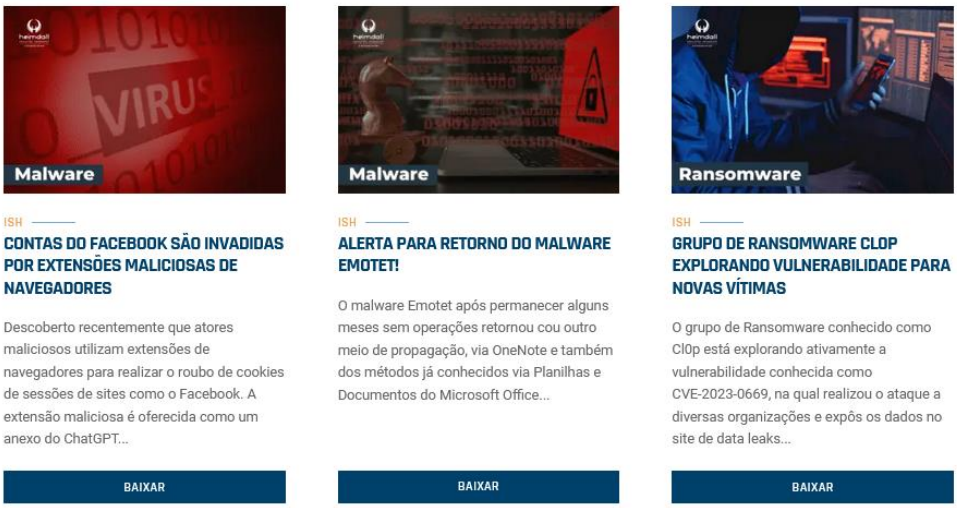
TLP: CLEAR

Receba alertas e informações sobre segurança cibernética e ameaças rapidamente, por meio do nosso **X**.

[Heimdall Security Research]

Acesse boletins diários sobre agentes de ameaças, *malwares*, indicadores de comprometimentos, TTPs e outras informações no *site* da ISH.

[Boletins de Segurança – Heimdall]



SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

1 SUMÁRIO EXECUTIVO

Pesquisadores de segurança cibernética identificaram uma nova variante de malware para macOS, denominada **TodoSwift**. Este malware tem apresentado características semelhantes a outros softwares maliciosos previamente atribuídos a grupos de hackers norte-coreanos.

2 INFORMAÇÕES SOBRE A AMEAÇA

Um arquivo assinado chamado TodoTasks foi enviado ao VirusTotal em 24 de julho de 2024. Este aplicativo exibe comportamentos semelhantes aos malwares originários da Coreia do Norte (RPDC), especificamente do grupo de hackers conhecido como BlueNoroff, como KandyKorn e RustBucket. Devido a essas semelhanças, acredita-se que este novo malware, que está sendo chamado de TodoSwift, provavelmente tem a mesma origem.

Na análise compartilhada pelos pesquisadores, focou-se especialmente no dropper do malware, um aplicativo GUI escrito em Swift/SwiftUI. Sob o pretexto de baixar e apresentar um PDF ao usuário, ele simultaneamente baixa e executa um binário malicioso de segunda etapa.

Ele começa com uma chamada para *makeWindowControllers*, pois isso configura o comportamento malicioso do aplicativo. Segundo a Apple, este método "*cria os objetos controladores de janela que o documento usa para exibir seu conteúdo*". Neste caso, o aplicativo envia este método para um objeto *NSDocument* personalizado chamado *TodoTaskDocument*.

```
100006464    void -[_TtC9TodoTasks8Document makeWindowControllers](struct _TtC9TodoTasks8Document* self, SEL sel)
100006470        id x0 = _objc_retain(obj: self)
100006478        GoogleDocANDBuy2xURLs(x0)
100006488        return _objc_release(obj: x0) __tailcall
```

Figura 1 – TodoTaskDocuments personalizado.

Este método invoca uma sub-rotina (que foi renomeada para GoogleDocANDBuy2xURLs()), a qual aceita o objeto TodoTaskDocument como argumento. Esta função é crucial para o que o usuário veja quando o windowController for carregado pelo aplicativo.

O espaço alocado contendo as strings de URL é passado para esta função através do registrador x20, uma convenção de chamada específica do Swift. Sabemos que existem duas strings no buffer passado para esta função, então as nomearemos de acordo com sua ordem, cada uma é utilizada de maneira diferente por esta função, e é importante entender essa diferença.

Há uma chamada para função callToCurl que aceita a primeira string para o Google Drive, além de outras duas que veremos a seguir.

```
100005000    int32_t success = callToCurl(firstSwiftString_pt1: *(AllocatedSpaceWithURLS + 0x10), firstSwiftString_pt2: *(AllocatedSpaceWithURLS + 0x18), StringStruct_1_ouputPath:
0xd000000000000018, StringStruct_2_ouputPath: 0x8000000100019080, strStruct(pw|gc) p1: 'gc', strStruct(pw|gc) p2: 0xe200000000000000)
```

Figura 2 – Chamada da função CallToCurl.

Observando adiante, pode-se perceber que esta função configura um objeto NSTask para executar um comando curl. O mais interessante é que esta função possui várias declarações condicionais que determinam qual NSTask criar, com base na flag que é passada para ela.

Como foi visto anteriormente, na primeira vez que isso é chamado, a string gc é passada como a flag. Agora pode-se observar como isso é utilizado.

Primeiro, uma chamada para inicializar o objeto NSTask:

```
1000053a8    struct objc_object* task = -[_TtC9TodoTasks8Document init](self: _objc_allocWithZone(cls: _OBJC_CLASS_$_NSTask, zone), sel: "init")
```

Figura 3 – Chamada da função CallToCurl.

Em seguida, é observada uma comparação:

```
1000053c4    if (strStruct(pw|gc) p1 != 'pw' || strStruct(pw|gc) Size != 0xe200000000000000)
1000053dc        stringMatch = _stringCompareWithSmolCheck(_:_:expecting:)(strStruct(pw|gc) p1, strStruct(pw|gc) Size, 'pw', 0xe200000000000000, 0)
```

Figura 4 – Comparação da função.

Agora, é realizada a chamada buildCurlCommand() para continuar a execução. A chamada para callToCurl() retorna um valor Booleano renomeado como success, que é verificado antes de prosseguir para a próxima parte da função. Vamos retornar ao chamador buildCurlCommand() para dar continuidade à execução. A função callToCurl() devolve um valor Booleano, que é renomeado para success e verificado antes de seguir para a próxima etapa da função.

```
100005004    if ((success & 1) != 0)
100005010        // /tmp/GoogleMsgStatus.pdf
100005010        openPDFSetup(strstructFilePath_/tmp/GoogleMsgStatus.pdf: 0xd000000000000018, strstructFilePath2_/tmp/GoogleMsgStatus.pdf: 0x8000000100019080)
```

Figura 5 – Chamada buildCurlCommand().

Agora que o PDF foi baixado pelo comando curl, executado pelo objeto NSTask, esta função gerencia a apresentação do PDF ao usuário. Lembre-se de que toda essa atividade é resultado do método makeWindowControllers para lidar com um NSDocument; neste caso, isso resultará na aplicação exibindo o PDF baixado. Veja como isso funciona.

Com o PDF já baixado pelo comando curl, executado pelo objeto NSTask, esta função é responsável por apresentar o PDF ao usuário. É importante lembrar que toda essa operação é consequência do método makeWindowControllers para manipular um NSDocument, nesse cenário, a aplicação exibirá o PDF baixado. Aqui está o processo.

```
1000056dc        __builtin_strcpy(dest: &file://, src: "file://")
1000056dc        int64_t sizeOfString = 0xe700000000000000
1000056ec        String.append(_:)(strstructFilePath_/tmp/GoogleMsgStatus.pdf, strstructFilePath2_/tmp/GoogleMsgStatus.pdf)
1000056fc        // initialize the file://pathToGoogle.pdf
1000056fc        URL.init(string:)(file://, sizeOfString)
```

Figura 6 – Download do arquivo .pdf.

3 RECOMENDAÇÕES

Além dos indicadores de comprometimento elencados abaixo pela ISH, poderão ser adotadas medidas visando a mitigação da infecção do referido *malware*, como por exemplo:

Utilize antivírus atualizado

- Mantenha seu software antivírus sempre atualizado para detectar e remover novas ameaças de malware.

Atualize sistemas e aplicativos

- Instale regularmente atualizações de segurança para corrigir vulnerabilidades que podem ser exploradas por malwares.

Cuidado com links e anexos

- Não clique em links ou abra anexos de e-mails suspeitos, mesmo que pareçam vir de fontes confiáveis.

Baixe aplicativos de fontes oficiais

- Instale aplicativos apenas de lojas oficiais para evitar softwares maliciosos disfarçados.

Use verificação em duas etapas

- Ative a autenticação de dois fatores para adicionar uma camada extra de segurança às suas contas.

Faça backups regulares

- Mantenha backups atualizados de seus dados importantes para se proteger contra perda de dados causada por malwares.

Monitore atividades suspeitas

- Fique atento a comportamentos estranhos no seu dispositivo, como lentidão ou pop-ups inesperados, que podem indicar a presença de malware.

4 INDICADORES DE COMPROMISSOS

A ISH Tecnologia realiza o tratamento de diversos indicadores de compromissos coletados por meio de fontes abertas, fechadas e também de análises realizadas pela equipe de segurança Heimdall. Diante disto, abaixo listamos todos os Indicadores de Compromissos (IOCs) relacionadas a análise do(s) artefato(s) deste relatório.

Indicadores de compromisso do artefato	
md5:	c64143eeb1a62c6d1ac0742263c692d8
sha1:	67b2a61c7114c7110140d934399372338345f356
sha256:	f1b3ce96462027644f9caa314d3da745dab139ee1cb14fe508234e76bd686f93
File name:	MasaMatsu.TodoTasks

Indicadores de compromisso do artefato	
md5:	73e7de61c40241cc39ce84934c9e65f3
sha1:	c1a80bcd7283195eca0190dc3645b73482467ad4
sha256:	9623c98f7338d56b07b35cd379e31e685e32a9c5317d7bc4af5276916cef4ed3
File name:	TodoTasks

Indicadores de compromisso do artefato	
md5:	c64143eeb1a62c6d1ac0742263c692d8
sha1:	67b2a61c7114c7110140d934399372338345f356
sha256:	f1b3ce96462027644f9caa314d3da745dab139ee1cb14fe508234e76bd686f93
File name:	MasaMatsu.TodoTasks

Indicadores de compromisso do artefato	
md5:	7dcbf2bb4a6620a82bc2c4ecb03504bc
sha1:	a72af7beb12a4357190fdee89b7dff3d18863906
sha256:	e09d2277a19dddd751edb164bde064682a6acc41a7ee178a2dacd4f9ac357fc7
File name:	GoogleMsgStatus.pdf

Tabela 1 – Indicadores de Compromissos de artefatos

5 REFERÊNCIAS

- **Heimdall *by* ISH Tecnologia**
- Kandji
- Thehackernews

6 AUTORES

- **Leonardo Oliveira Silva**

heimdall
security research
A DIVISION OF ISH