Integrating Generative AI into Honeypots for Capturing Botnets

Cleber Reis[1], **Eduardo Lourenço**[1], **Reginaldo Santos**[1], **André Riker**[1,2]

[1] Federal University of Pará (UFPA) – Belém – PA – Brasil
{cleber, regicsf, ariker@}@.ufpa.br
eduardo.lourenco@itec.ufpa.br

[2]Arizona State University (ASU) – Tempe – AZ –, USA.
riker.a@asu.edu

Abstract. *A honeypot is a defense technique that complements defense systems, which are generally composed of firewalls and intrusion detection systems. A honeypot consists of a "bait" system that simulates attractive and vulnerable targets for potential attackers. In this article, we address the problem of the lack of dynamism in traditional honeypots, which are based on static scripts, with a special focus on botnet-type threats. To tackle this problem, the proposed solution integrates a large language model into a honeypot, capable of interacting with the attacker by dynamically imitating a target device. The open-source tool Cowrie was used as the traditional reference honeypot to receive functionalities from the large language model, including command response generation, log data analysis, and the emission of alerts and actions. Regarding the evaluation, tests were conducted to measure the accuracy of the command responses produced by the proposed honeypot, the time required for response generation, and the number of tokens. The obtained results reveal the viability of the proposed integration, showing that a large language model is a promising alternative for honeypots to achieve a higher level in deceiving attackers, especially botnets.*

Resumo. *Honeypot é uma técnica que complementa os sistemas de defesa em cibersegurança. Um honeypot consiste em um sistema "isca" que simula alvos atraentes e vulneráveis para potenciais atacantes. Neste artigo, abordamos a falta de capacidade de honeypots baseados em scripts para simular um dispositivo real para atacantes botnet. Além disso, outro problema abordado é a falta de funcionalidades automatizadas de geração de dados relacionados ao contexto, análise de dados de log para criação de relatórios e emissão de alertas e ações de mitigação. Para atacar estes problemas, propomos um honeypot que integra IA generativa, um modelo grande de linguagem capaz de melhorar a interação com o atacante e prover ações automatizadas de: (i) adaptação de respostas de comando; (ii) geração de informação de contexto; (iii) geração de relatórios, execução de ações de defesa e emissão de alertas. Além disso, este artigo apresenta e avalia uma prova de conceito da funcionalidade responsável pelas respostas de linha de comando, usando o honeypot Cowrie como referência. Foram feitos testes para medir a precisão das respostas de linha de comandos produzidos pelo honeypot proposto, o tempo necessário para geração das respostas e o número de Tokens. Os resultados obtidos revelam a viabilidade da integração proposta, mostrando que modelo grande de linguagem é uma alternativa promissora para que honeypots atinjam um maior nível no engano de atacantes, em especial botnets.*

1. Introdução

Entre as diversas ameaças existentes para sistemas de computação, *botnet* tem apresentado um grande potencial nocivo a cibersegurança. *Botnet* pode ser definida como uma

grande rede de dispositivos invadidos, chamados de *bots*, que são controlados por um atacante central. Esses *bots*, muitos são dispositivos IoT, podem ser usados para lançar várias ameaças online, como ataques distribuído de negação de serviço, espalhar e-mails de *spam* e enganar pessoas em fraudes [Winkler e Gomes 2016]. Por exemplo, em outubro de 2016, o *malware* Mirai infectou mais de 100.000 dispositivos IoT de pequenos escritórios e escritórios domésticos [Bertino e Islam 2017], sendo o responsável por diversos ataques cibernéticos.

As *botnets* realizam "Busca e Infiltração", que é um período no qual há a busca por vulnerabilidades em um conjunto de alvos. Uma vez que um dispositivo vulnerável é identificado, um *bot* atacante tenta acessar o dispositivo vulnerável remotamente usando ataques de dicionário ou explorando vulnerabilidades conhecidas. Após conseguir o acesso remoto da vítima, o *bot* invasor lança um ataque de escalada de privilégio para ter permissão suficiente e infectá-lo. Se bem-sucedido, uma versão compatível do *malware* é instalada e ativada, assim a rede de *bots* é expandida.

Koroniotis et al. [Koroniotis et al. 2019] mostram algumas abordagens eficazes a serem usadas contra *botnets* para impedir a expansão. Para este fim, destaca-se sistemas de Detecção de Intrusão (IDS), análise de fluxo de rede, inspeção profunda de pacotes e *honeypots*. Por definição, os *honeypots* são projetados para parecerem um alvo atraente e explorável para um atacante. Os objetivos de um *honeypot* são: (i) identificar atacantes; (ii) mitigar ataques em andamento, distrair os atacantes de dispositivos reais e coletar dados de tentativas de invasão. Ao analisar os dados coletados pelo *honeypot*, é possível reunir informações sobre redes *botnet* que estão ativas e seu comportamento. Por meio dessas informações é possível acionar medidas automatizadas baseadas em políticas e contramedidas de segurança eficazes [Bagui e Li 2021].

Ao revisar a literatura, é possível encontrar muitas soluções projetadas para fornecer defesa a ataques *botnets*. Neste conjunto de soluções, existe uma linha de trabalhos que foca em usar *honeypots*. Um grande número de soluções deixa os dispositivos "isca" em máquinas virtuais que são instanciadas de maneira dinâmica visando aumentar a capacidade de lidar uma grande escala de *botnets*. Este tipo de solução produz uma interação com o atacante muito realista, mas consome muitos recursos computacionais. Outras soluções de *honeypots*, baseadas em *scripts*, conseguem lidar um maior número de atacantes demandando poucos recursos, porém produzem falhas ao interagirem com o atacante. Essas falhas os deixam passíveis de serem identificados por softwares como um dispositivo isca. Desta forma, é um problema ter de *honeypots* que não conseguem se adaptar às interações remotas demandadas por um atacante e produzir respostas realísticas. Ao notar que um sistema não produz respostas compatíveis, um atacante interrompe a sessão de ataque e o *honeypot* não consegue capturar informações valiosas sobre o atacante, tais como: o tipo e versão do *malware*, dados que identifiquem a rede de *bots*, o endereço IP da central de comando da *botnet*. Adicionalmente a este problema, existe o desafio de ter soluções automatizada que suporte à geração e análise de dados armazenados no log e dispare relatórios, alertas e execute ações de defesa.

Para enfrentar esses problemas, este artigo apresenta:

- Um modelo conceitual de *honeypot* que integra uma IA generativa, especificamente um Modelo Grande de Linguagem (*Large Language Model* - LLM). A LLM integrada ao *honeypot* proposto confere à solução as seguintes funcionalidades automatizadas: (i) interação com o atacante e geração adaptativa de respostas de linha de comando; (ii) geração de dados de contexto; (iii) Produção de relatórios, emissão de alertas e ações de defesa.
- Uma prova de conceito da funcionalidade (i), considerando o *Cowrie*, de código aberto, o qual foi implantado e exposto a atacantes reais. A avaliação realizada

visou verificar a precisão, tempo de resposta e número de tokens da LLM ao submetê-la a tarefa de supervisionar a interação com o atacante e alterar partes da resposta do *honeypot* quando necessário.

Este artigo está organizado da seguinte forma: A Seção 2 introduz os trabalhos relacionados. A Seção 3 detalha a solução proposta. A Seção 4 descreve uma prova de conceito referente ao uso de LLM pelo *honeypot* para melhorar a interação com o atacante. A Seção 5 apresenta as conclusões e trabalhos futuros.

2. Trabalhos Relacionados

Existe muitos trabalhos voltados ao desenvolvimento de *honeypots* visando o aumento de segurança em ambientes cibernéticos. No entanto, propostas de integração de LLM a *honeypots* têm sido pouco exploradas. Ao considerar a versatilidade do modelo LLM, um sistema de defesa com *honeypot* é capaz de lidar com as características dinâmicas do atacante e fazer análise e geração de dados. Neste sentido, o modelo é capaz de gerar respostas de forma contextualizada às consultas de um usuário malicioso, tornando o *honeypot* mais robusto [Christli et al. 2024]. Deste modo, esta seção busca apresentar trabalhos com a seguinte classificação: i) *Honeypots* para mitigar ataques *botnets*; ii) *Honeypots* Baseados em LLM para mitigar ataques *botnet*. A metodologia de busca considerou as seguintes palavras-chave: *Honeypot*, *Internet of Things*, *Botnets*, *Large Language Models*, *Secure Shell*, *Cybersecurity*. As seguintes listas de bases foram consideradas para o presente trabalho: SBC OpenLib, IEEE Xplore, ACM Digital Library, ScienceDirect, Springer Nature e Wiley.

2.1. Honeypots para Mitigar Ataques Botnet

Os autores [Memos e Psannis 2022] introduzem uma infraestrutura de Virtualização de Funções de Rede (NFV) que utiliza tecnologias emergentes para aprimorar a defesa contra diversas ameaças de rede orquestradas por *botnets* especializados. Especificamente adaptada para Inteligência Artificial das Coisas, a arquitetura proposta tem o objetivo de fornecer um ambiente e resiliente ao incorporar recursos oriundos da computação em nuvem, fornecendo tanto defesa proativa quanto detecção precisa de potenciais atividades maliciosas. No entanto, apesar do trabalho enfatizar os problemas relacionados a *botnets*, a solução não apresenta testes e resultados que comprovem a viabilidade e eficácia da arquitetura.

O trabalho apresentado por [Trajanovski e Zhang 2021] propõe uma arquitetura dividida em dois elementos: o Bloco de Captura de *Botnet* (BCB) e o Bloco de Análise de *Botnet* (BAB). O BCB implanta *honeypots* para simular vulnerabilidades, capturando dados dos *botnets*. Esses *honeypots* interagem com os atacantes, registrando tentativas em um banco de dados. As amostras capturadas passam por análise do BAB por meio de Interfaces de Programação de Aplicativos (APIs), *parsers*, *sandboxes* e vários analisadores, por exemplo, estático, comportamental, de rede, antivírus e classificação de malware. O *framework* inclui camadas de simulação, extração de *payload* e relatórios. A análise do BAB abrange o envio de amostras, *parsing*, execução em *sandbox* e extração de dados para detectar indicadores de comprometimento e ataque. Contudo, diferente de [Trajanovski e Zhang 2021], nosso trabalho não carece de uma infraestrutura complexa para que o *honeypot* interaja de forma contextual ao atacante.

[Ma et al. 2024] propõem uma arquitetura de rede de *honeypot* dinâmica e adaptável. O trabalho considera um sistema de defesa especificamente para redes corporativas. A estratégia recai no redirecionamento do tráfego suspeito para a rede de *honeypot* a fim de fornecer maior segurança à rede corporativa. Adicionalmente, este artigo modela os comportamentos de ataque e defesa usando teoria dos jogos onde o jogo é não cooperativo

e de informação incompleta. Ao modelar e analisar o processo de jogo de ataque-defesa de estágio único, o artigo apresenta os benefícios de migração do tráfego suspeito, buscando encontrar uma configuração de rede de *honeypot* e uma estratégia de migração de tráfego eficiente. Embora o artigo considere um custo de migração atrelado aos recursos do sistema, a determinação precisa deste custo em um ambiente real é uma tarefa complexa.

[Omar et al. 2024] introduzem uma nova abordagem que utiliza *honeypots* de hardware como uma camada defensiva adicional contra *Trojans* de *Hardware* (HTs). O sistema proposto foi implementado em um Raspberry Pi e testado em um circuito emulado usando um *Field-Programmable Gate Array* (FPGA). Diferente dos artigos vistos anteriormente, este trabalho comprova a viabilidade da solução no uso de dispositivos de baixo poder computacional. Contudo, o artigo aborda a necessidade de adição de algoritmos de Aprendizado de Máquina para o sistema de defesa, limitando a análise aprofundada sobre o comportamento do atacante.

Os autores [Kristyanto e Louk 2024] propõem combinar *honeypots* com algoritmos de aprendizado por reforço, de modo que os *honeypots* se tornem *honeypots* adaptativos que aprendam com o comportamento do atacante. Este artigo compara dois algoritmos de Aprendizado por Reforço baseados em Q-learning, especificamente *Double Deep Q-Network* (DQN) e *Double Deep Q-Network* (DDQN). Os resultados apresentados mostram que o algoritmo DDQN é mais otimizado na determinação de ações quando comparado ao algoritmo DQN. Assim como nosso trabalho, os autores fornecem uma solução de *honeypot* dinâmico. Contudo, o nosso enfoque de defesa considera um ambiente IoT, viabilizando o uso de LLM ao *honeypot*.

De maneira semelhante ao visto no artigo de [Kristyanto e Louk 2024], os autores [Guan et al. 2023] também adotam o uso de *honeypots* integrados com Aprendizado por Reforço para atrair atacantes. No entanto, este último aborda o uso de Cadeia de Markov e estatísticas reais coletadas de dispositivos IoT reais. No entanto, o processo de rastreamento do ataque analisado no artigo demanda altos recursos computacionais e tempo.

2.2. Honeypots Baseados em LLM para Mitigar Ataques Botnet

O artigo de [Guan et al. 2024] apresenta uma abordagem inovadora para a criação de *honeypots*, utilizando LLMs para simular um ambiente de *shell* realista. Neste artigo, os autores propõem o uso de LLMs comerciais, treinados com registros reais de comandos de *shell* e respostas correspondentes, para gerar respostas completas durante as sessões de ataque. Além disso, técnicas como *in-context learning* e *chain-of-thought* são empregadas para aprimorar a precisão e a consistência das respostas do modelo. Diferentemente da solução proposta por [Guan et al. 2024], a qual gera todo conteúdo da resposta usando LLM, nós apresentamos uma solução que usa LLM para alterar parcialmente as respostas de terminal de comando. Acredita-se que a geração parcial das respostas trará ganhos de precisão e tempo de resposta.

A solução de [Christli et al. 2024] também combina o uso de *honeypot* com LLM a fim de gerar respostas adaptativas. O objetivo dos autores é fornecer maior segurança em ambientes *shell* usando o modelo LLaMA-3 via API. Na arquitetura, o sistema processa as entradas do atacante através de um servidor SSH Paramiko, o qual interage com o modelo de LLM para gerar respostas contextuais. Para medir o grau de similaridade entre as respostas do *honeypot* e as respostas reais, os autores utilizaram a distância de Levenshtein. Diferente de [Christli et al. 2024], que se limita ao LLaMA-3, nossa solução considera a análise de diversos modelos LLM em cenários IoT.

2.3. Análise e Desafios

Conforme observado, a literatura propõe diversas estratégias para melhorar o sistema de defesa com *honeypots*. Algumas abordagens incorporam em suas arquiteturas o conceito de instanciamento dinâminco de VMs, o que garantem um desempenho realista do *honeypot*, no entanto, a dificuldade recai na escalabilidade de tais soluções. Outro grupo de soluções, sob o objetivo de consumir menos recursos computacionais e atingir maior escalabilidade, utilizam-se de *honeypots* baseados em *scripts*. Porém, poucas dessas integram IA generativa para melhorar as funcionalidades do *honeypot*.

A Tabela 1 faz uma análise comparativa da nossa solução com o estado da arte.

Tabela 1. Tabela de Trabalhos Relacionados.

Trabalhos	Baseado em *script*	Foco em Botnet	Integra IA Generativa	Teste Real de Viabilidade
[Memos e Psannis 2022]	✓	✓	×	×
[Trajanovski e Zhang 2021]	×	✓	–	✓
[Ma et al. 2024]	×	×	–	✓
[Omar et al. 2024]	✓	✓	–	✓
[Kristyanto e Louk 2024]	×	×	×	✓
[Guan et al. 2023]	✓	✓	×	✓
[Guan et al. 2024]	×	×	✓	✓
[Christli et al. 2024]	×	×	✓	✓
Nosso trabalho	✓	✓	✓	✓

3. Modelo Conceitual de um Honeypot Baseado em LLM para Ataques Botnet

Honeypots são armadilhas projetadas para atrair atacantes, enganá-los e coletar informações sobre suas táticas, funcionamento, ferramentas, métodos e motivos. Um *honeypot* age como uma isca, imitando um sistema ou serviço real e atrai os atacantes para desperdiçarem seus esforços, onde suas ações são monitoradas [Zhang et al. 2019]. Esta seção propõe a arquitetura de um *honeypot* que integra uma *Large Language Model* (LLM). A Seção 3.1 apresenta a arquitetura geral, mostrando o seu funcionamento em um cenário definido. A Seção 3.2 apresenta detalhes específicos do *honeypot* proposto.

3.1. Visão Geral

O objetivo do *honeypot* proposto é atrair e enganar atacantes, assim como coletar, gerar e analisar dados sobre os ataques de maneira automatizada. A Figura 1 apresenta a visão geral da arquitetura do *honeypot* proposto, onde há uma central de comando *botnet* (1), coordenando as ações dos bots infectados. Conforme ilustrado na Figura 1, há um atacante bot (2) que escaneia as vulnerabilidades dos dispositivos próximos, localizados no ambiente alvo (3). O ambiente alvo é formado por dispositivos como câmeras, fechaduras inteligentes, smartphones, tomadas inteligentes, smart televisores e outros dispositivos IoT.

Como ilustrado na Figura 1, o *honeypot* proposto é executado em um dispositivo localizado no ambiente alvo e se passa por um dispositivo vulnerável. Como é possível observar, o *honeypot* integra funcionalidades de uma LLM em três módulos: (A) Módulo de interação com o atacante e geração adaptativa de respostas de linha de comando; (B) geração de dados de contexto; (C) Produção de relatórios, emissão de alertas e ações de defesa. As próximas seções detalham de cada módulo proposto que compõe a arquitetura.

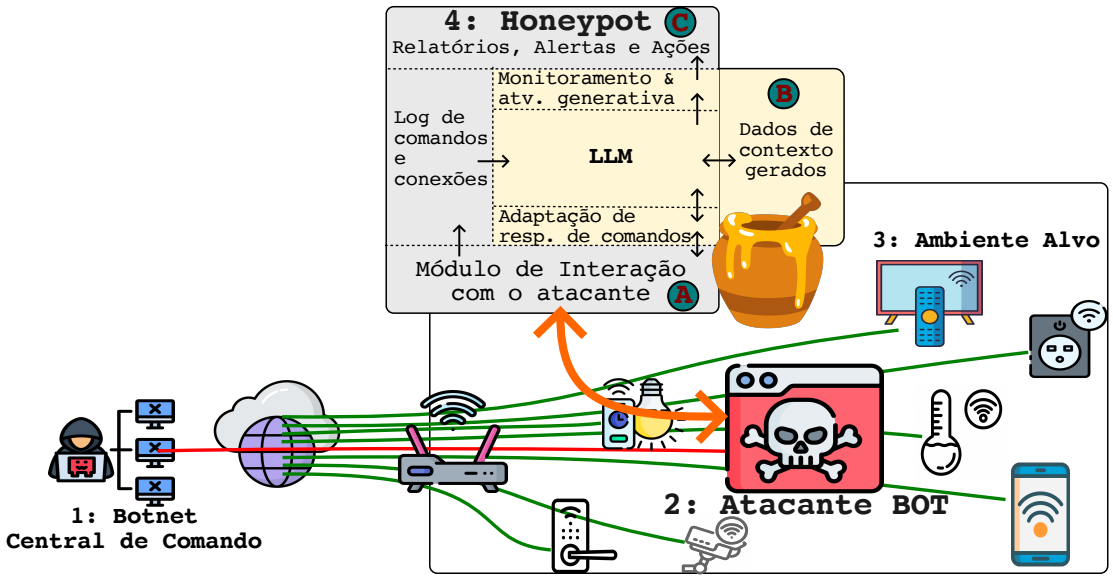


Figura 1. Visão geral: a arquitetura proposta em um cenário com atacante *botnet*

3.2. Módulo de interação com o atacante e geração adaptativa de respostas de linha de comando

Neste trabalho, assumimos que um *bot* que compõe a *botnet* já possui acesso remoto ao dispositivo alvo. É razoável levar em conta essa possibilidade, uma vez que a maior parte dos alvos infectados por *botnets* são aqueles que possuem senhas fracas do tipo `admin`, `admin123` ou `aaaa1234`. Ou seja, os infectados por *botnets*, em geral, são aqueles dispositivos vulneráveis à ataques de dicionário.

De maneira sequencial, as interações entre o atacante *botnet* e o *honeypot* proposto ocorrem da seguinte forma:

1. O atacante envia para o Centro de Comando e Controle (C&C Server) da *Botnet*, a instrução: "Escalar privilégio". Nesta fase, o *botnet* já possuía acesso remoto ao terminal do dispositivo alvo e tem como objetivo executar uma técnica de escalada de privilégio para instalar um *malware* no alvo, de modo a transformá-lo em membro da sua *botnet*.
2. Para atingir o objetivo de escalar o privilégio no dispositivo alvo, o C&C dispara uma série de comandos $C = 1, ..., R$ via SSH.
3. O *Honeypot* recebe estes comandos e repassa-os a um Agente.
4. O Agente tem a responsabilidade de gerar uma resposta CLI de referência para o comando e, em seguida, alterar essa referência usando LLM.
5. Por fim, o Agente repassa ao *Honeypot* a resposta alterada pela LLM, que será devolvida para o C&C.

A LLM deve alterar os dados sensíveis presentes no CLI de referência. Por exemplo, alterar os nomes de usuário, endereço IP e MAC, versão do sistema operacional, arquivos de configuração e outros dados em arquivos. Ao ter que apenas alterar os dados sensíveis, a LLM diminui a sua tarefa e aproxima o nível de realidade que o atacante tem ao interagir com o *Honeypot*.

3.3. Módulo de Geração de Dados de Contexto

Este módulo é responsável em extrair informações de contexto das interações do atacante com o *honeypot*. Essa extração é proveniente da análise dos dados armazenados nos logs

e nas interações atuais. Os dados de contexto, são informações que visa complementar os dados capturados pelo *honeypot* tradicional. Exemplos de dados de contexto são:

- Intenção do Atacante: Análise dos comandos e arquivos acessados para inferir o objetivo do ataque. Por exemplo, uma sequência de comandos como uname *-a, whoami, cat /etc/passwd*, seguida por tentativas de usar wget ou scp para baixar arquivos, pode indicar uma tentativa de escalada de privilégios e exfiltração de dados.
- Sofisticação do Atacante: Avaliação do nível técnico do invasor com base nos comandos utilizados e na compreensão do sistema. Por exemplo, o uso de scripts complexos ou a tentativa de explorar vulnerabilidades específicas (que podem ser detectadas nos logs de tentativas de ataque) indicariam um nível mais elevado de sofisticação
- Origem (Inferida): Embora o endereço IP de origem já seja um dado capturado, a análise das interações pode fornecer pistas adicionais sobre a possível origem geográfica e até mesmo o idioma nativo do atacante. A dedução da possível localização geográfica ou idioma do atacante através de padrões nos comandos e payloads.

3.4. Módulo de Geração de Relatórios, Alertas e Ações de Defesa

Este módulo visa analisar as informações dos dados brutos capturados e das informações de contexto para decidir uma recomendação, emissão de alertas e ações automatizadas.

- Bloqueio Dinâmico de IP com Base na Intenção Maliciosa Inferida: Se a LLM identificar, através da sequência de comandos e tentativas de acesso, uma alta probabilidade de uma tentativa de ataque direcionado (como exploração de vulnerabilidade específica ou tentativa de exfiltração após reconhecimento), o sistema pode automaticamente adicionar o endereço IP de origem do atacante a uma blacklist no firewall do *honeypot* e, potencialmente, em firewalls perimetrais (se a integração for configurada). O nível de confiança da LLM na inferência da intenção maliciosa pode ser usado como um limiar para acionar essa ação. Por exemplo, se a LLM estiver 90% confiante de que se trata de uma tentativa de exfiltração, o bloqueio é acionado.
- Criação Dinâmica de Regras de Firewall Específicas: Caso a LLM detecte um padrão de ataque específico, como repetidas tentativas de conexão a portas não usuais ou a exploração de um serviço simulado específico no *honeypot*, o sistema pode gerar e implementar dinamicamente regras de firewall mais granulares. Por exemplo, se a LLM identificar múltiplas tentativas de explorar uma vulnerabilidade simulada no serviço SSH em uma porta não padrão, uma regra de firewall temporária pode ser criada para bloquear o tráfego para essa porta específica vindo do IP atacante. Essa ação é mais específica que um bloqueio total e pode mitigar a ameaça sem interromper outras interações menos suspeitas.
- Geração e Envio de Alertas Enriquecidos com Contexto: Quando uma atividade suspeita é detectada e analisada pela LLM, o sistema pode gerar alertas que vão além das informações básicas do log. O alerta pode incluir a descrição da possível intenção do atacante (inferida pela LLM), o nível de sofisticação estimado, os comandos chave utilizados e até mesmo a possível origem geográfica inferida. Esses alertas enriquecidos podem ser enviados para equipes de segurança via SIEM, e-mail ou outras plataformas de comunicação, fornecendo um contexto valioso para uma análise mais rápida e informada por analistas humanos, mesmo que ações automatizadas não sejam totalmente implementadas em um primeiro momento. A LLM pode até sugerir ações de remediação com base no contexto da ameaça.

4. Prova de Conceito do Módulo de interação com o Atacante e Geração Adaptativa de Respostas de Linha de Comando

Nesta seção apresentamos uma prova de conceito que consiste na implantação do *honeypot Cowrie* em um ambiente real para captura de dados de atacantes reais e os dados obtidos são utilizados para testar o desempenho das LLMs. Portanto, a finalidade desta seção é apresentar uma discussão sobre a viabilidade do uso de LLM no módulo responsável pela interação com o atacante.

4.1. Metodologia dos Testes Conduzidos

A Figura 2 apresenta a metodologia seguida nesta seção para testar a viabilidade do uso de LLMs no módulo de interação com o atacante. Esta metodologia é dividida em seis etapas, como descrito a seguir:

1. Etapa 1: Implantar o *Cowrie* em ambiente real, realizar a captura de dados reais com os atacantes. Fazer uma análise da interação do atacante com o *honeypot* implantando, verificando os comandos usados, tempo de sessão, IP de origem, número de acessos, entre outros.
2. Etapa 2: Realizar uma seleção dos comandos de terminal usados pelos atacantes, os quais serão considerados nos testes.
3. Etapa 3: Após a fase de seleção, estes comandos são submetidos ao *honeypot Cowrie*. As respostas obtidas são armazenadas para serem usadas como referência.
4. Etapa 4: As respostas de referência dos comandos obtidas na etapa anterior são armazenadas.
5. Etapa 5: Nesta etapa, a LLM recebe as referências, dispara um processo generativo e gera uma nova saída.
6. Etapa 6: As saídas da LLM são armazenadas para comparação com as referências e cálculo de desempenho.

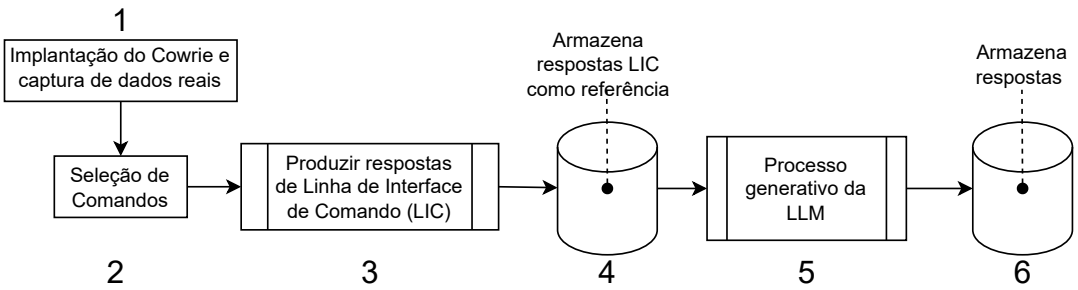


Figura 2. Etapas da Metodologia aplicada nos testes.

4.2. Etapa de Implantação do Honeypot Cowrie em Ambiente Real e Seleção dos Comandos

O *honeypot Cowrie* foi implantado usando a versão v2.6.1+126, com as seguintes especificações de hardware e software: PC Intel Core 2 Duo, 8GB de memória RAM, Sistema Operacional Ubuntu 20.04.6 LTS, com um endereço IP público 200.239.xxx.xxx (ofuscado por motivos de segurança), escutando ativamente na porta 22. A configuração de autenticação do *Cowrie* permite a aceitação de qualquer tentativa de login com quaisquer credenciais. Essa abordagem visa maximizar a captura de interações de atacantes. Além disso, o *honeypot* foi configurado para responder *Linux svr04 3.2.0-4-amd64 #1 SMP*

Debian 3.2.68-1+deb7u1 x86_64 quando o atacante enviar o comando *uname -s -v -n -r -m*.

A coleta de dados por meio do *honeypot* instalado ocorreu entre os dias 13/05/2025 à 10/06/2025, totalizando 29 dias. O número de acessos ao *Cowrie* chegou a atingir um pico de 19529 acesso no dia 27 do experimento. Como pode ser observado na Figura 3, nota-se que os acessos foram originados de IPs distintos, principalmente nos dias 28 e 29. O grande número de tentativas de acesso é coerente com o fato do *honeypot* ter sido instalado em uma rede de um campus de uma instituição de ensino superior que é alvo constante de ataques cibernéticos.

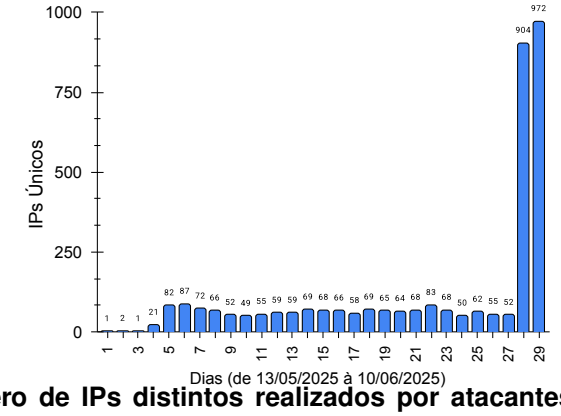


Figura 3. Número de IPs distintos realizados por atacantes na porta SSH do Cowrie.

Uma vez que o atacante obtém acesso ao ambiente do *honeypot*, o *Cowrie* inicia um processo detalhado de registro. Toda e qualquer entrada digitada pelo invasor é capturada, seja um comando legítimo, uma tentativa de comando, ou qualquer outra sequência de caracteres. Esses logs são meticulosamente armazenados em uma pasta dedicada, com arquivos organizados por data e disponíveis nos formatos .json e .log, facilitando a análise posterior das atividades maliciosas.

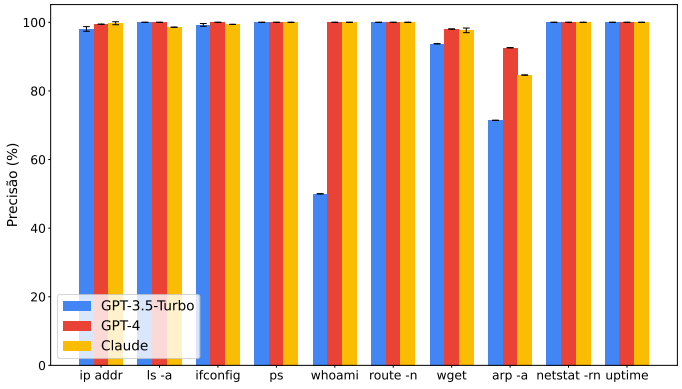
Ao analisar os dados coletados pelos atacantes, foram selecionados 10 comandos para serem utilizados nas etapas seguintes da metodologia de testes com as LLMs. Os comandos selecionados foram: *ip addr, ls -a, ifconfig, ps, whoami, route -n, wget, arp -a, netstat -rn e uptime.*

4.3. Avaliação e Resultados Obtidos das LLMs

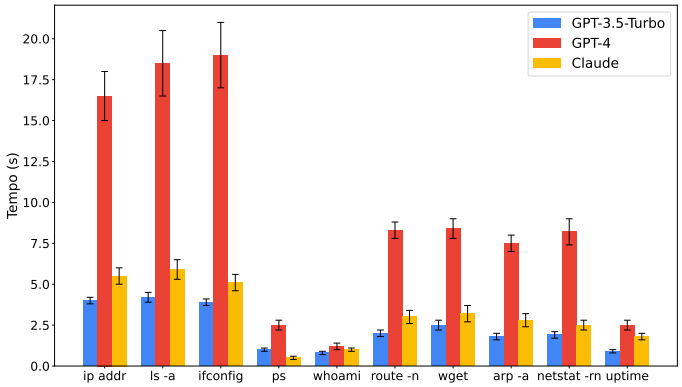
O objetivo principal dessa avaliação é realizar um estudo de viabilidade das LLMs em prover as funcionalidades necessárias para o módulo de Módulo de interação com o atacante e geração adaptativa de respostas de linha de comando. Para atingir esse objetivo, foram selecionadas as seguintes métricas para aferir o desempenho das LLMs:

- **Precisão**: Valor percentual referente ao número de palavras corretamente alterados em relação ao total.
- **Tempo**: Esta métrica refere-se ao tempo levado pela LLM para gerar o terminal de saída.
- **Número de Tokens**: Este é o número de tokens totais gerados pela LLM. Essa métrica indica a quantidade de texto alterado no comando.

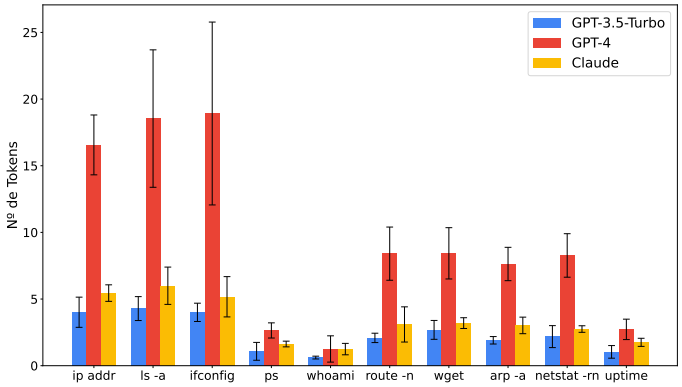
Para os resultados apresentados neste artigo, os 10 comandos selecionados foram executados 30 vezes em cada LLM. Dessa maneira, os valores apresentados em cada



(a)



(b)



(c)

Figura 4. Desempenho do Módulo de Interação com o Atacante: (a) Precisão. (b) Tempo de resposta. (c) Número de Tokens gerados.

gráfico é uma média de 30 execuções. Um especialista independente foi utilizado para o cálculo da precisão.

A Figura 4 apresenta os resultados obtidos para três LLMs testadas, nomeadamente Claude-sonnet, GPT-4 e GPT-3.5-Turbo. Essas LLMs foram executadas através de API, assim não foram executadas localmente. Estes modelos não foram retreinados, a técnica utilizada para que os modelos gerassem as respostas desejadas foi Engenharia de *Prompt*.

A Figura 4.a) mostra a precisão obtida para os dez comandos testados. Neste quesito, o Claude-sonnet e o GPT-4 obtiveram precisão média de 98%, enquanto o GPT-3.5-Turbo ficou com 93%. Quanto ao tempo, ver 4.b), o modelo Claude-sonnet se destaca com o melhor resultado, seguido do GPT-3.5-Turbo e, por último, GPT-4. Em média, o melhor modelo levou cerca de 4 segundos para gerar a saída do terminal.

Por fim, a Figura 4.c) apresenta o número de tokens totais gerados pelos modelos. É possível observar que o GPT-3.5-Turbo, apesar de ter a menor precisão, obteve maior resultado em termos de geração de tokens. Indicando que este modelo alterou mais conteúdo do terminal de referência do que o necessário.

4.4. Discussão sobre a viabilidade

Por meio da análise dos resultados obtidos, pode-se notar que a precisão atingida pelo melhor modelo (GPT-3.5-Turbo) é satisfatória, porém é importante promover melhorias. Esse nível de precisão, nos leva a ideia de que com técnicas de refinamento (*fining tuning*), as LLMs irão atingir uma precisão mais alta, atingindo níveis que não comprometam o engano que o *honeypot* precisa produzir.

Quanto ao tempo, os resultados obtidos são mais alarmantes, pois 4 segundos é um valor alto que pode comprometer a interação com o atacante. Dessa forma, ao invés da geração completa das respostas de comandos, deve-se buscar soluções que façam apenas a supervisão das respostas geradas pelo *Cowrie*. Assim, o número de tokens podem ser reduzidos e tempo necessário para produzir uma saída da LLM pode também diminuir.

5. Conclusões e Trabalhos Futuros

Botnet é uma ameaça real para ambientes cibernéticos, particularmente para redes de Internet das Coisas (IoT). A proliferação de *botnets* causa prejuízos recordes todos os anos. Assim, é um desafio real mitigar os efeitos de ataques e infecção de *botnets*.

Com o objetivo de encontrar uma solução inovadora para este desafio, este artigo apresenta um *Honeypot* baseado em LLM. A solução proposta é projetada para gerar saídas de terminais de comando de modo a enganar atacantes *botnets*. Tal solução foi implementada e testada usando três modelos LLMs usando engenharia de prompt. Os resultados obtidos mostram um melhor desempenho do modelo Claude-sonnet.

Os resultados obtidos neste trabalho são promissores para os trabalhos futuros. Planeja-se comparar os resultados obtidos com uma das soluções presentes na literatura. Outro ponto a ser avançado é o retreinamento da LLM, conduzindo uma fase de *fine-tunning*, e rodar o modelo localmente. Além disso, será criado um ambiente real de teste no Campus da Universidade Federal do Pará (UFPA) para capturar dados das tentativas reais de atacantes *botnets*. Os dados coletados na UFPA serão utilizados por um módulo a ser adicionado na solução que será capaz de bloquear e coletar informações úteis sobre os *botnets*.

Agradecimentos

Este trabalho foi parcialmente financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) projeto 444978/2024-0 e 405026/2024-2 e também conta com recursos PROAP oriundos da CAPES/UFPA.

Referências

Bagui, S. e Li, K. (2021). Resampling imbalanced data for network intrusion detection datasets. *Journal of Big Data*, 8(1):6.

Bertino, E. e Islam, N. (2017). Botnets and internet of things security. *Computer*, 50(2):76–79.

Christli, J. A., Lim, C., e Andrew, Y. (2024). Ai-enhanced honeypots: Leveraging llm for adaptive cybersecurity responses. In *2024 16th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pages 451–456. IEEE.

Guan, C., Cao, G., e Zhu, S. (2024). Honeyllm: Enabling shell honeypots with large language models. In *2024 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9.

Guan, C., Liu, H., Cao, G., Zhu, S., e La Porta, T. (2023). Honeyiot: Adaptive high-interaction honeypot for iot devices through reinforcement learning. In *Proceedings of the 16th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 49–59.

Koroniotis, N., Moustafa, N., Sitnikova, E., e Turnbull, B. (2019). Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Generation Computer Systems*, 100:779–796.

Kristyanto, M. A. e Louk, M. H. L. (2024). Evaluation and comparison of the use of reinforcement learning algorithms on ssh honeypot. *Teknika*, 13(1):77–85.

Ma, L., Chen, M., e Liu, L. (2024). Vdh: a dynamic honeynet technology based on game theory. In *Fourth International Conference on Machine Learning and Computer Application (ICMLCA 2023)*, volume 13176, pages 430–438. SPIE.

Memos, V. A. e Psannis, K. E. (2022). Nfv-based scheme for effective protection against bot attacks in ai-enabled iot. *IEEE Internet of Things Magazine*, 5(1):91–95.

Omar, A. H. E., Soubra, H., Moulla, D. K., e Abran, A. (2024). An innovative honeypot architecture for detecting and mitigating hardware trojans in iot devices. *IoT*, 5(4):730–755.

Trajanovski, T. e Zhang, N. (2021). An automated and comprehensive framework for iot botnet detection and analysis (iot-bda). *IEEE Access*, 9:124360–124383.

Winkler, I. e Gomes, A. T. (2016). *Advanced persistent security: a cyberwarfare approach to implementing adaptive enterprise protection, detection, and reaction strategies*. Syngress.

Zhang, W., Zhang, B., Zhou, Y., He, H., e Ding, Z. (2019). An iot honeynet based on multiport honeypots for capturing iot attacks. *IEEE Internet of Things Journal*, 7(5):3991–3999.