UNIVERSIDADE DE CAXIAS DO SUL
ÁREA DO CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIAS

MARCELO VINICIUS ZANOL

INTEGRAÇÃO DE FERRAMENTAS CONTRA ATAQUES DDOS E MALWARE

CAXIAS DO SUL -RS
2018

MARCELO VINICIUS ZANOL

INTEGRAÇÃO DE FERRAMENTAS CONTRA ATAQUES DDOS E MALWARE

Trabalho de Conclusão de Curso para obtenção do título de Bacharel em Sistemas de Informação pela Universidade de Caxias do Sul, na Área do Conhecimento de Ciências Exatas e Engenharias.

Orientadora Profa. Dr. Maria de Fátima Webber do Prado Lima

CAXIAS DO SUL - RS

2018

RESUMO

O objetivo principal deste trabalho foi o desenvolvimento do *software* Iot Manager and Security Toolkit para prevenção de ataques DDoS e *malware*. O número de dispositivos conectados à Internet (IoT) aumenta a cada dia. O avanço da tecnologia cada vez mais adentra o nosso cotidiano contribuindo para a flexibilidade e mobilidade das mais diversas tarefas: segurança, automação doméstica, indústria 4.0, automação de hospitais e serviços médicos, etc. O maior desafio a este avanço é manter um controle de segurança contra ataques de *malwares, DoS e DDoS*. Ferramentas *open-source* de segurança contra ataques DDoS possuem funcionalidades limitadas ou muito específicas, obrigando o responsável de segurança da rede a possuir diferentes aplicações, uma para cada funcionalidade. Este trabalho aprimorou o *software* de administração de redes *BYOD Manager Toolkit* aumentando sua abrangência contra este tipo de ataque utilizando algumas ferramentas *open-source* disponíveis no mercado. As ferramentas selecionadas para compor o *software* de integração de ferramentas foram *Port Tester 1.0, THC Hydra 8.6, Angry IP Scanner* e *John the Ripper*. O *software* desenvolvido possui compatibilidade com a plataforma Windows e uma de suas principais características é a fácil utilização e estrutura para adição de novas funcionalidades sem grandes alterações no código.

Palavras-chave: IoT. Ferramentas *Open-Source*. DoS. DDoS. Malware.

ABSTRACT

The main purpose of this paper was the development of the software IoT Manager and Security Toolkit to prevent DDoS and malware attacks. The number of devices connected to the Internet (IoT devices) increases every day. The advancement of technology increasingly penetrates our daily life contributing to flexibility and mobility of the most diverse tasks: security, home automation, 4.0 industry, automation of hospitals and medical services, and others. The biggest challenge to this advancement is to maintain a security control against malware, DoS and DdoS attacks. Open-source security tools against DDoS attacks have limited or very specific features, forcing network security officers to have different applications, one for each feature. This work has improved the BYOD Manager Toolkit network administration software using some open source tools to increase its functionalities against this type of attack. The tools cited in this work to compose the tool integration software are Port Tester 1.0, THC Hydra 8.6, Angry IP Scanner and John the Ripper. The developed software is Windows compatible and its main features are ease of use and structure for further development of new functionalities without big changes in its code.

Keywords: IoT. Open Source Tools. DoS. DDoS. Malware.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGLAS

Sigla	**Significado**
ACK	*Acknowledge*
API	*Application Programming Interface*
BYOD	*Bring Your Own Device*
CPU	*Central Process Unit*
CSV	*Comma-separated Values*
DDOS	*Distribute Denial of Service*
DOS	*Denial of Service*
DVR	*Digital Video Recorder*
FTP	*File Transfer Protocol*
GUI	*Graphical User Interface*
HTTP	*Hypertext Transfer Protocol*
HTTPS	*Hypertext Transfer Protocol Secure*
ICMP	*Internet Control Message Protocol*
IDE	*Integrated Development Environment*
IP	*Internet Protocol*
IOT	*Internet of Things*
JNA	*Java Native Access*
JVM	*Java Virtual Machine*
MAC	*Media Access Control*
OS	*Operating System*
RAM	*Random Access Memory*
SMTP	*Simple Mail Transfer Protocol*
SSH	*Secure Shell*
TCP	*Transmission Control Protocol*
TI	Tecnologia da Informação
TXT	Extensão para Arquivos de Texto
UDP	*User Datagram Protocol*
WEB	*World Wide Web*
XML	E*xtensible Markup Language*

SUMÁRIO

1. INTRODUÇÃO

O constante avanço e estudo em tecnologia garante uma evolução que cada vez mais adentra o cotidiano, aumentando flexibilidade e mobilidade até nas atividades mais básicas. Comprar mantimentos em um mercado ou até analisar a imagem de câmeras de segurança em tempo real podem ser realizados através de um dispositivo móvel como um *smartphone* ou um computador conectado à internet independente da sua posição atual. Este constante avanço é denominado como Internet das Coisas ou *Internet of Things* (IoT).

O termo IoT refere-se à conexão de diferentes dispositivos, desde carros, *laptops*, *smartphones* até microondas, geladeiras, semáforos e dispositivos cardíacos. A lista de dispositivos conectados à internet é extensa e a cada dia aumenta. Com dispositivos conectados é possível analisar os dados coletados através de sensores e trabalhar eles para atingir diferentes dispositivos (MEOLA, 2016).

Cada dispositivo que está conectado ganha privacidade e preocupações de segurança em torno da IoT. Essas preocupações variam de *hackers* roubando dados e até ameaçando vidas, até corporações que podem descobrir facilmente dados privados disponibilizados de forma descuidada (EASTWOOD, 2017).

A fraqueza inerente da segurança em dispositivos IoT, combinada ao lançamento do código fonte do *botnet* Mirai, fomentaram a inovação dos invasores e mudaram a situação para empresas que dedicam a atenção necessária à defesa contra DDoS e às melhores práticas de defesa (CIO, 2017).

Relacionando a rede e nossos dispositivos conectados à Internet com moradias, na qual há um grande investimento em segurança para evitar quaisquer invasões físicas, deve-se investir em segurança da rede virtual contra diferentes ataques, e entre eles, os Ataques de Negação de Serviço ou *Denial of Service* (DoS) e sua variante Ataques de Negação de Serviço Distribuidos ou *Distribute Denial of Service* (DDoS).

Ataques DoS consistem em tentativas de fazer com que computadores tenham dificuldade, ou mesmo sejam impedidos, de executar suas tarefas. Para isso, em vez de "invadir" o computador ou mesmo infectá-lo com *malwares*, o autor do ataque faz com que a máquina receba tantas requisições que esta chega ao ponto de não conseguir dar conta delas. Em outras palavras, o computador fica tão sobrecarregado que nega serviço (ALECRIM, 2012).

Ataques DDoS é um tipo de ataque DoS de grandes dimensões, ou seja, que utiliza até milhares de computadores para atacar uma determinada máquina (Figura 1), distribuindo a ação entre elas (ALECRIM, 2012). O atacante, através de uma máquina mestre ou mais, infecta

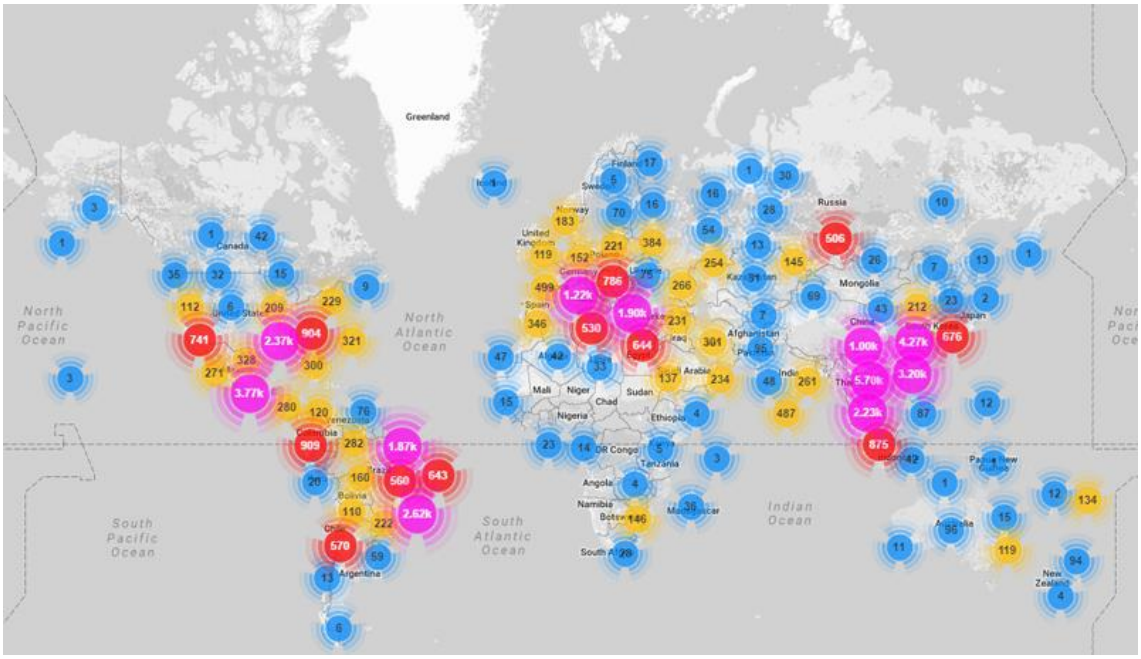
diferentes dispositivos tornando-os dispositivos zumbis, que podem permanecer assim por muito tempo já que pode não apresentar um comportamento anormal. Com diversos dispositivos zumbis, o atacante os utiliza remotamente para efetuar requisições à um site ou serviço alvo.

Figura 1 - Representação de um ataque DDoS.

Fonte: ALECRIM, 2012.

A partir de setembro de 2016, uma série de ataques DDoS temporariamente derrubaram os sites Krebs on Security, OVH e Dyn. O ataque inicial em Krebs excedeu 600 Gbps em volume, um dos maiores registrados. Destaca-se que esse tráfego imenso foi proveniente de centenas de milhares de *hosts* sob o controle da *botnet*, conhecida como Rede Mirai (ANTONAKAKIS, 2017). O ataque a fornecedora de infraestrutura Dyn dificultou o acesso a sites como Twitter, Amazon, e portais de notícias, além de outros dispositivos que formam a chamada Internet das Coisas (IoT). Ataques deste tipo estão cada vez mais frequentes após a liberação do código fonte do *botnet* (COMPUTERWORLD, 2017). Dentre os países com mais dispositivos infectados pela Rede Mirai (Figuras 2 e 3), o Brasil encontra-se em segundo lugar, perdendo apenas para o Vietnã, aumentando mais ainda a importância de uma análise de segurança em todos os dispositivos (HERZBERG; BEKERMAN; ZEIFMAN, 2016).

Figura 2 - Geolocalização de dispositivos infectados pela Rede Mirai descobertos até o momento.



Fonte: HERZBERG; BEKERMAN; ZEIFMAN, 2016.

Figura 3 - Relação dos países com maior proporção de dispositivos infectados pelo Mirai

País	% de IPs dos botnets Mirai
Vietnã	12.8%
Brasil	11.8%
Estados Unidos	10.9%
China	8.8%
México	8.4%
Coréia do Sul	6.2%
Taiwan	4.9%
Russia	4.0%
Romênia	2.3%
Colômbia	1.5%

Fonte: HERZBERG; BEKERMAN; ZEIFMAN, 2016.

Em uma análise do código fonte do Mirai foi constatado que este *malware* possui dois propósitos. O primeiro deles é localizar e comprometer dispositivos IoT. O segundo propósito é realizar ataques DDoS através de instruções recebidas remotamente.

Para aumentar sua distribuição em diferentes dispositivos o Mirai varre grandes faixas de IPs e tenta conectar utilizando portas abertas e credenciais disponibilizadas por diferentes fabricantes de hardware. Credenciais estas que nunca são alteradas pelos consumidores por

acreditarem que ninguém fará uso de sua rede. Através de uma vasta lista de senhas padrões (Figura 4), o Mirai conecta-se ao dispositivo e elimina um possível concorrente (Anime *malware*), garantindo o seu controle territorial.

Figura 4 - Algumas senhas padrões do dicionário utilizado pelo Mirai.

Password	Device Type	Password	Device Type	Password	Device Type
123456	ACTi IP Camera	klv1234	HiSilicon IP Camera	1111	Xerox Printer
anko	ANKO Products DVR	jvbzd	HiSilicon IP Camera	Zte521	ZTE Router
pass	Axis IP Camera	admin	IPX-DDK Network Camera	1234	Unknown
888888	Dahua DVR	system	IQinVision Cameras	12345	Unknown
666666	Dahua DVR	meinsm	Mobotix Network Camera	admin1234	Unknown
vizxv	Dahua IP Camera	54321	Packet8 VOIP Phone	default	Unknown
7ujMko0vizxv	Dahua IP Camera	00000000	Panasonic Printer	fucker	Unknown
7ujMko0admin	Dahua IP Camera	realtek	RealTek Routers	guest	Unknown
666666	Dahua IP Camera	1111111	Samsung IP Camera	password	Unknown
dreambox	Dreambox TV Receiver	xmhdipc	Shenzhen Anran Camera	root	Unknown
juantech	Guangzhou Juan Optical	smcadmin	SMC Routers	service	Unknown
xc3511	H.264 Chinese DVR	ikwb	Toshiba Network Camera	support	Unknown
OxhlwSG8	HiSilicon IP Camera	ubnt	Ubiquiti AirOS Router	tech	Unknown
cat1029	HiSilicon IP Camera	supervisor	VideoIQ	user	Unknown
hi3518	HiSilicon IP Camera	<none>	Vivotek IP Camera	zlxx.	Unknown
klv123	HiSilicon IP Camera				

Análise em 30/09/2016. Fonte: ANTONAKAKIS, 2017.

A fim de mitigar o avanço do Mirai, há duas ações principais a serem tomadas. A primeira delas é alterar as senhas padrões dos dispositivos. A segunda ação a ser tomada é desabilitar o acesso remoto (WAN) dos dispositivos e fechar as portas SSH (22), Telnet (23) e HTTP/HTTPS (80/443) (HERZBERG; BEKERMAN; ZEIFMAN, 2016).

Assim como o Mirai, existem outros tipos de vírus que realizam ataques DoS e DDoS. É importante realizar uma análise de segurança contra ataques DDoS em todos os dispositivos IoT das redes para que possa ser mitigado o potencial dos ataques. Além de identificar possíveis *malwares* já instalados.

Prevenir ou identificar um ataque DDoS em andamento não é uma tarefa simples que pode ser realizada totalmente sem intervenção humana, porém há técnicas que facilitam este trabalho, como o uso de *proxy* reverso e um firewall com uma boa configuração de regras de segurança. Algumas ferramentas *proxy* disponíveis no mercado são a Squid, DeleGate, DansGuardian e Oops. De acordo com (FIDELIS, 2013) *proxy* significa um programa ou serviço que faz o papel de intermediário entre o cliente e o servidor para o acesso de um determinado serviço. O *proxy* pode ser utilizado para várias aplicações, como HTTP, HTTPS, FTP e outros, fazendo o trabalho de interceptar as informações trafegadas e liberando ou negando as informações ao cliente

O engenheiro militar do Centro de Defesa Cibernética do Exército, Alexandre Godinho, afirma que os ataques por meio de dispositivos de IoT têm uma dimensão "incalculável", especialmente porque as pessoas que usam as tecnologias não se preocupam com a segurança (BERBERT, 2017).

Com cada dia mais dispositivos conectados, ataques DoS atuais serão pequenos comparado à possíveis ataques que estão por vir, visto que há grande interesse de hackers em ataques à dispositivos IoT e não há uma regulamentação de segurança específica para este meio.

Existem ferramentas de segurança que permitem identificar portas desprotegidas em dispositivos na rede, outras apenas escaneiam quais dispositivos estão conectados na rede. Ferramentas mais completas e complexas são pagas, porém há uma grande diversidade de soluções *open-source* para realizar atividades específicas. Uma ferramenta em que unisse elementos como identificação de dispositivos e verificação de portas desprotegidas pode ser muito eficiente para a análise de segurança de uma rede com dispositivos IoT.

1.1 PROBLEMA E QUESTÃO DE PESQUISA

A todo momento novos dispositivos são conectados à Internet. Esta conexão atinge usuários dos mais variados meios, desde hospitais que conectam monitores cardíacos até residências para controlar suas câmeras de segurança pelo *smartphone*.

Ferramentas atuais para segurança contra ataques DDoS possuem funcionalidades limitadas ou muito específicas, como por exemplo, o Port Tester 1.0 permite apenas identificar portas desprotegidas, enquanto outras, como o Angry IP Scanner, permitem apenas escanear dispositivos conectados na rede, obrigando o responsável de segurança da rede a possuir diferentes aplicações, uma para cada funcionalidade específica.

Desta forma, a principal motivação deste trabalho é a pesquisa e o incremento de funcionalidades a uma ferramenta de código aberto, permitindo que o responsável pela segurança da rede possa mapear todos os dispositivos conectados e identificar quais possuem portas desprotegidas utilizando um *software* que concentre todas as funcionalidades. Além disso, identificar quais destes dispositivos estão vulneráveis a ataques do *malware* como o Mirai e ataques de força bruta.

Com a definição do problema supra citado, a questão de pesquisa pode ser definida por: Quais funcionalidades e ferramentas *open-source* podem ser agregadas a um *software* de segurança de redes, para fins de análise de vulnerabilidades contra ataques DDoS.

1.2 OBJETIVO GERAL

Analisar e integrar soluções *open-source* desenvolvidas para o ambiente Windows que auxiliem na proteção dos dispositivos IoT conectados a rede contra ataques DDoS e *malwares*. Os objetivos específicos são:

- Incrementar com novas funcionalidades o *software open-source* desenvolvido pelo aluno Vinícius Lahm, aumentando sua abrangência de segurança.
- Realizar um levantamento ferramentas *open-source* para análise de vulnerabilidades contra DDoS e códigos *malware*.
- Disponibilizar mais ferramentas de segurança *open-source* para ambientes Windows através de uma integração de diferentes *softwares*.
- Permitir a execução individual dos softwares selecionados.
- Integrar os softwares selecionados a fim de que os mesmos compartilhem informações e operem de forma complementar.

1.3 METODOLOGIA

A metodologia adotada neste trabalho é composta por quatro etapas. Na primeira etapa será feito um estudo teórico sobre ataques DoS, DDoS e *malware* focando em como funciona e como prevenir os ataques, e realizar uma busca por ferramentas *open-source* com funcionalidades importantes para prevenção dos ataques.

Na segunda etapa do processo, será feito um breve estudo sobre as ferramentas existentes. Deste modo será possível identificar quais destas ferramentas previamente selecionadas serão escolhidas para terem funções agregadas no *software* desenvolvido por Vinícius Lahm Perini. Os incrementos devem ser desenvolvidos em Java.

Depois de informações coletadas e analisadas, será o momento de desenvolver as novas funcionalidades para o *software*. Nesta etapa, todos os conceitos e tecnologias necessárias serão aplicados durante o processo de programação.

Finalmente, na quarta etapa será feito um teste para verificar se o *software* cumpre os seus requisitos e se contribui para o responsável pela segurança da rede com uma análise contra ataques DDoS.

1.4 ESTRUTURA DO TRABALHO

O trabalho está estruturado da seguinte maneira: Capítulo 2 (Ataques em Dispositivos IoT) no qual são apresentados diferentes tipos de ataques à dispositivos IoT. Além disso, há um detalhamento do conceito de ataques DoS e DDoS. No Capítulo 3 (Levantamento de Ferramentas Open Source) é definida qual a metodologia de pesquisa adotada durante o desenvolvimento do projeto. Diversas opções de ferramentas disponíveis são exemplificadas neste capítulo. Finalmente, as ferramentas que foram escolhidas para o trabalho são citadas. No Capítulo 4 (Definição do Software), é apresentada a ferramenta BYOD Manager Toolkit (PERINI, 2017) que receberá a integração de ferramentas de segurança, tal como sua arquitetura de software e modelagem. No Capítulo 5 (Desenvolvimento do Software) é apresentada a estrutura de funcionamento e as alterações realizadas no *software*. No Capítulo 6 (Testes e Resultados) são apresentados os diferentes testes realizados a fim de validar as funcionalidades. No Capítulo 7 (Conclusão) é apresentada a conclusão final do trabalho.

2. ATAQUES EM DISPOSITIVOS IOT

Antes do uso de *smartphones* o principal paradigma para segurança de computadores e redes nas empresas era uma TI corporativa muito controlada. Dispositivos eram limitados à computadores com Windows em terminais finais e servidores. A segurança da rede era clara e tinha perímetros bem definidos que separavam as redes confiáveis da Internet não confiável. Com o aumento da utilização de dispositivos móveis nas organizações os perímetros que definiam redes seguras de não seguras de uma forma simples não são mais aplicáveis exigindo uma segurança maior à este tipo de dispositivo (STALLINGS, 2013).

Ameaças à segurança das redes através destes dispositivos fez com que fossem necessárias medidas de proteção adicionais e especializadas. O uso de aplicativos criados por terceiros impõe um risco de ter instalado um *software* malicioso de forma proposital que posteriormente poderá se instalar na rede interna da empresa espalhando para os demais dispositivos conectados.

A seção 2.1 apresenta o OWASP *Internet of Things Project* e algumas de suas considerações sobre serviços de redes. A seção 2.2 exemplifica os principais tipos de *malwares* conhecidos até o momento. A seção 2.3 conceitua os principais tipos de ataques que as redes de computadores sofrem: *malwares* e ataques de negação de serviço.

2.1 OWASP INTERNET OF THINGS PROJECT

O OWASP *Internet of Things Project* é um projeto liderado por Daniel Miessler e Craig Smith que visa prover informações de segurança para fabricantes, desenvolvedores e usuários de dispositivos IoT, auxiliando a compreender problemas de segurança para construir e desenvolver dispositivos mais seguros (OWASP, 2017).

O projeto é estruturado em diversos subprojetos como, por exemplo, áreas de ataque, orientações para testes e principais vulnerabilidades.

O subprojeto de Áreas de Ataque lista diversas áreas de dispositivos IoT e suas vulnerabilidades. Uma das áreas é a *Device Network Services* (Serviços de Rede do Dispositivo) que possui diversas vulnerabilidades como, DoS, serviços UDP vulneráveis, vulnerabilidades no gerenciamento de credenciais (usuário enumerado, senhas fracas, credenciais padrões conhecidos) e serviços de Testes/Desenvolvimento (OWASP, 2017).

O subprojeto de Orientações para Testes visa auxiliar testadores de dispositivos IoT a efetuar uma avaliação de segurança disponibilizando uma diretriz de testes básicos como um

guia inicial. Não trata-se de um guia detalhado porém é abrangente quanto a áreas de teste, como por exemplo, interface WEB insegura, autenticação/autorização insuficiente, serviços de redes não seguros, falta de criptografia no transporte e preocupações com a privacidade. Em serviços de redes o projeto define que é necessário testar e garantir que os serviços não respondam mal a ataques de negação de serviço ou *buffer overflow*, e também verificar se as portas de testes não ficaram ativas (OWASP, 2017).

O subprojeto de principais vulnerabilidades segue a mesma estrutura dos outros subprojetos e apresenta as principais vulnerabilidades de cada área do dispositivo e um breve detalhamento de cada uma delas. Senhas fracas como ‘1234’ ou ‘123456’, bem como o uso de senhas default, são citadas como uma vulnerabilidade em quatro possíveis áreas: Interface Administrativa, Interface WEB, Interface Cloud e aplicações para dispositivos móveis. Outra vulnerabilidade citada é a possibilidade do dispositivo sofrer ataques de negação de serviço tornando determinado serviço ou o dispositivo todo inacessível (OWASP, 2017).

O projeto disponibiliza um Guia de Segurança para IoT para os fabricantes, desenvolvedores e usuários, separados por categorias e suas considerações. Para que o fabricante garanta um nível básico de segurança para os serviços de rede do dispositivo é necessário certificar que o dispositivo possua o mínimo possível de portas ativas, certificar que não seja utilizado portas/serviços de rede (UPnP) com conexão à Internet e revise os serviços vulneráveis aos ataques DoS. Para consumidores, o projeto considera que sejam ativados os serviços de *firewall*, se existirem, e também segmente a rede isolando os dispositivos IoT de outros sistemas críticos (OWASP, 2017).

O OWASP possui uma ligação direta com o *software* uma vez que ele apresenta medidas de segurança mínimas que preferencialmente devem ser tomadas também por usuários, porém a falta de informação na hora da aquisição do *hardware* favorece o cenário atual de desproteção.

2.2 MALWARE

A denominação *malware* é proveniente da combinação das palavras *malicious* e *software*, significando "programa malicioso". Portanto, *malware* nada mais é do que um nome criado para fazer alusão a um *software* malicioso, seja ele um vírus, um *worm*, um *spyware*, um *ransonware* ou demais tipos (ALECRIM, 2017).

O Cavalo de Troia (*trojan*) é um tipo de *malware* com o objetivo de disponibilizar um acesso remoto externo à um dispositivo infectado. O *trojan* não foi desenvolvido para se replicar, porém sempre vem carregado com outros tipos de *malware*, como *backdoors*, *rootkits*,

ransomwares e *spywares.* Para executar a infecção o *trojan* geralmente se passa por um programa ou arquivo legítimo enganando o usuário (INCAPSULA, 2017).

O *Backdoor* tem a finalidade de anular a autenticação necessária para acessar um sistema, seja ele um banco de dados ou um servidor web. Combinado com a engenharia social para estudar a vítima e obter suas credenciais de *login*, o atacante poderá remotamente executar comandos no sistema sem que a vítima perceba. Recentemente, *Backdoors* foram encontrados em vários dispositivos da Internet das coisas (IoT), como as câmeras Wi-Fi de segurança usadas por organizações. Uma vez que um dispositivo IoT foi pirateado e transformado em um *backdoor*, ele efetivamente fornece uma forma de acesso sem executar os procedimentos de autenticação devidos (INCAPSULA, 2017).

Rootkits são softwares com alta complexidade de desenvolvimento que concedem acesso à partes sensíveis de aplicativos e podem até alterar as configurações do sistema (INCAPSULA, 2017). Apesar de não possuir uma grande variedade devido sua complexidade, *rootkits* são difíceis de serem detectados conseguindo se camuflar até à processos ativos na memória (ALECRIM, 2017). Esta dificuldade de detecção permite que ele permaneça instalado na máquina da vítima por muito tempo, fazendo desde capturas de telas, até monitoramento do tráfego na rede.

Spywares são responsáveis por fazer uma espionagem na vítima, capturando entradas de teclas e posteriormente enviando estes dados ao atacante, dados como senhas e logins de aplicativos e sites que o usuário tenha acessado (ALECRIM, 2017).

Ransomware é responsável por sequestrar o computador da vítima criptografando pastas e arquivos. Este tipo de *malware* é capaz de se espalhar por uma rede inteira criptografando não apenas uma máquina, mas sim todas máquinas e servidores conectados. Com os dados criptografados, uma janela informa ao usuário que seus arquivos permanecerão criptografados a menos que seja pago um resgate (INCAPSULA, 2017).

Em maio de 2017 o *malware* WannaCry paralisou diversos hospitais e instalações governamentais no mundo todo solicitando resgates de 300 à 600 dólares que deveriam ser pagos em Bitcoins (SYMANTEC, 2017).

Worms inicialmente foram feitos para infectar um computador, se clonar, e infectar outros computadores conectados na rede, através de emails ou até mesmo a rede local. Esta infecção não se limita apenas à computadores, mas sim todos dispositivos IoT conectados na rede, criando *botnets*, no qual são utilizados não para danificar ou roubar dados pessoais da vítima, mas sim, utilizar todos dispositivos infectados para executar um ataque muito maior, por exemplo, um ataque de negação de serviço (DoS) (INCAPSULA, 2017).

2.3 DOS E DDOS

Ataques de Negação de Serviço (DoS) é um tipo de ataque onde o invasor não instala *malware* no equipamento da vítima, porém tenta impedir o acesso à informações ou serviços. Segmentando a conexão de rede, ou os computadores, o invasor pode impedir acessos a *sites*, *e-mails* e outros serviços *online* (US-CERT, 2013).

Dentre os diversos tipos de ataques DoS, o mais comum deles é o de inundação (*flood*) (MCDOWELL, 2013). Um ataque *flood* consiste em um invasor enviar numerosas requisições para um servidor web e quando este servidor não conseguir processar todas requisições, ele não envia as respostas mesmo que a requisição seja autêntica, gerando uma negação de serviço.

DDoS é uma variação de ataque DoS no qual um ataque é disparado através de diversos dispositivos infectados pelo mundo todo (*botnet*) potencializando um ataque à uma escala global. Ataques DDoS podem ser divididos em três tipos principais: Ataques Baseado em Volume, Ataques de Protocolo e Ataques de Camada de Aplicação (INCAPSULA, 2017).

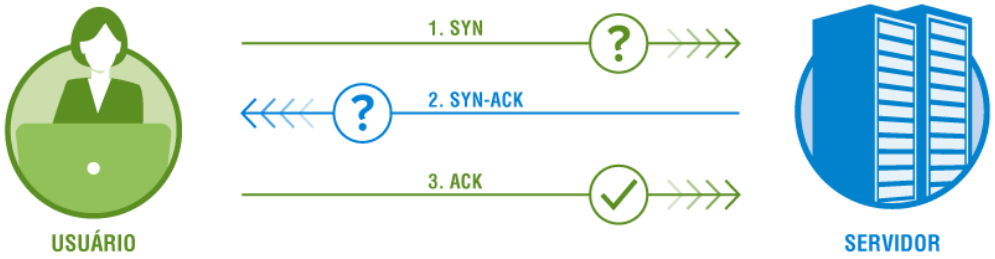
UDP *Flood* (Inundação UDP (*User Datagram Protocol*)) é um ataque baseado em volume e consiste em ataques DDoS utilizando pacotes UDP em portas aleatórias do equipamento alvo. O receptor checa as aplicações associadas ao pacote recebido, não encontrando uma aplicação que esteja esperando uma resposta da porta que recebeu o pacote, retorna um outro pacote com a informação de “Destino Inacessível”. Conforme aumenta o número destes pacotes o sistema fica sobrecarregado e não atende novas solicitações. Como o protocolo UDP não estabelece conexão, um grande volume de pacotes pode ser enviado para um *host* sem qualquer proteção interna contra este tipo de ataque, mostrando que ataques UDP são muito efetivos e exigem poucos recursos para executá-lo (INCAPSULA, 2017).

ICMP *Flood* (Inundação ICMP) é similar ao ataque UDP, porém a forma de sobrecarregar o servidor é através de pacotes ICMP (*ping*), com uma velocidade superior comparado ao UDP *Flood*. Este ataque não aguarda a resposta do pacote anterior para enviar uma nova solicitação, sabendo que haverá o mesmo número de pacotes sendo recebidos, um para cada solicitação.

SYN *Flood* (Inundação SYN) consiste em um ataque enviando diversos pacotes que explorem o estabelecimento de conexão do protocolo TCP e o torne o alvo inacessível. O estabelecimento de conexão funciona em três etapas. Na primeira etapa o processo cliente envia um pacote SYN (Sincronizar) para o processo servidor solicitando o estabelecimento de conexão. Na segunda etapa, o processo servidor envia um pacote SYN-ACK (Reconhecimento

de Sincronismo) indicando que aceita o pedido de estabelecimento de conexão. Na terceira etapa, o processo cliente envia um novo pacote ACK (Reconhecimento) confirmando o estabelecimento da conexão. Nesse momento a conexão é estabelecida com sucesso. (Figura 5) (INCAPSULA, 2017).

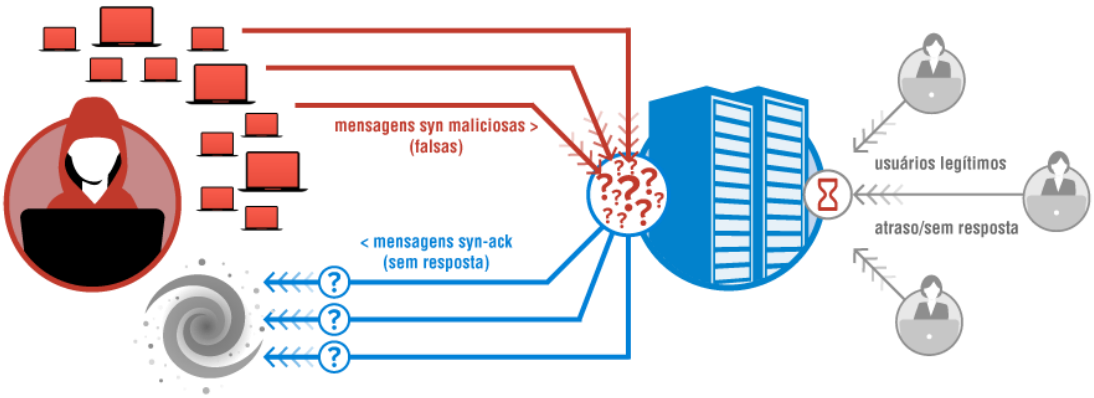
Figura 5 - Conexão Three-way Handshake.



Fonte: VERISIGN, 2017.

Em um ataque deste tipo, o atacante, muitas vezes utilizando um IP falso, envia pacotes SYN para cada porta do servidor alvo, que por sua vez identifica como uma tentativa de conexão legítima e responde enviando um pacote SYN-ACK para cada solicitação. O atacante então envia um pacote ACK que já era esperado ou então nunca recebe o pacote SYN-ACK, no qual o servidor fica esperando por uma resposta por um certo período (Figura 6). Neste meio tempo novos pacotes SYN são enviados para o servidor, que irá mantendo as conexões meio abertas até que a tabela de conexão do servidor fique cheia e comece a recusar novos pacotes SYN de clientes legítimos (INCAPSULA, 2017).

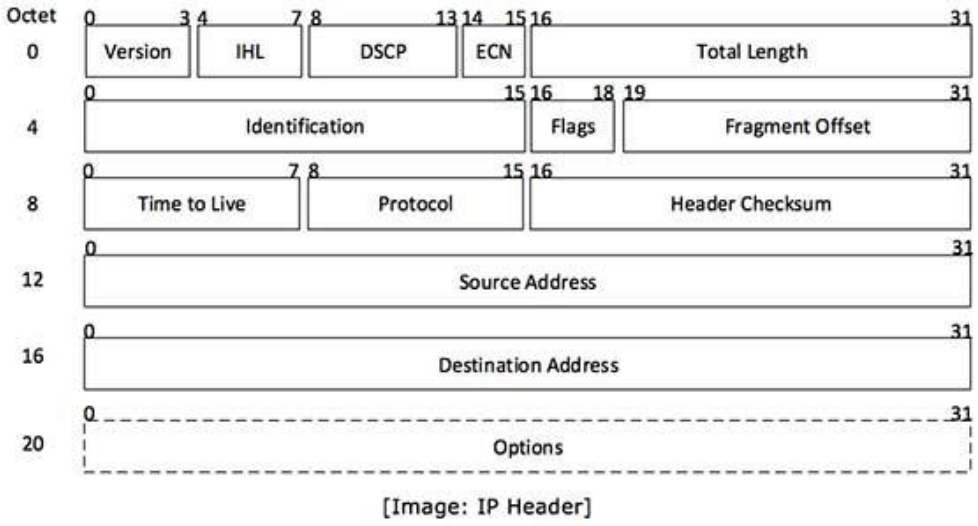
Figura 6 - SYN Flood.



Fonte: VERISIGN, 2017.

Ping of Death (Ping da Morte) é um tipo de ataque onde são utilizados pacotes mal formados ou com tamanho acima do permitido utilizando comandos de *ping*, que não necessita esperar por um pacote de resposta para completar a operação. Um pacote IPv4 bem formado e com seu cabeçalho incluído possui até 65,535 bytes. Como enviar pacotes com tamanho superior ao permitido viola o protocolo da Internet, o atacante envia pacotes mal formados ou pacotes fragmentados para o servidor alvo possibilitando que envie um pacote com tamanho superior à 65,535 bytes. O servidor vítima do ataque através do identificador (Figura 7) enviado no cabeçalho do pacote verifica que trata-se de pacotes fragmentados e ao tentar unir os pacotes pode ter seu buffer de memória totalmente alocado causando uma negação de serviço (INCAPSULA, 2017).

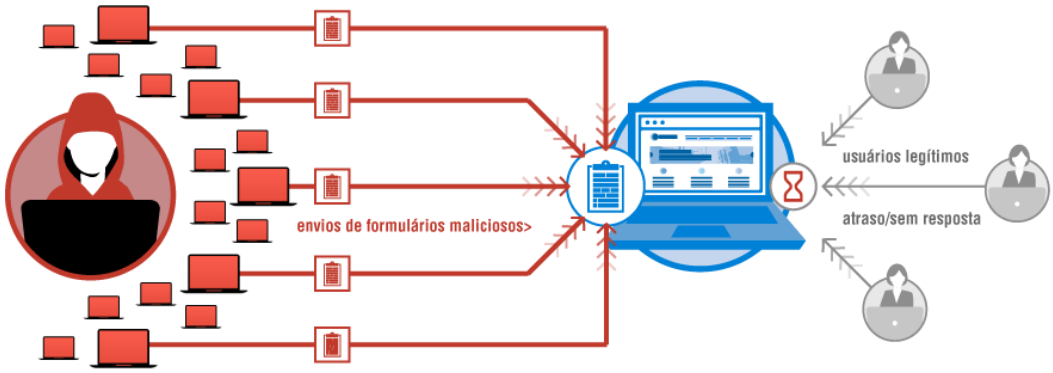
Figura 7 – Estrutura cabeçalho pacote IPv4.



Fonte: W3II, 2017.

Diferente dos ataques que exploram as camadas 3 e 4 (protocolos IP e TC/UDP), o ataque *HTTP Flood* (Inundação HTTP) explora a camada de aplicação. Este tipo de ataque imita o comportamento humano o que o torna mais difícil de mitigar (SECURITYLOCK, 2016). Para realizar o ataque, ele utiliza os dois comandos básicos do protocolo, os comandos GET e POST. O comando GET é utilizado para recuperar imagens ou conteúdos estáticos. O comando POST (Figura 8) é utilizado para enviar formulários ou comandos dinâmicos, sendo o comando POST mais efetivo para este tipo de ataque. Neste tipo de ataque, o atacante utilizando *botnets* sobrecarrega o servidor web que tenta alocar o máximo de recursos para atender à todas requisições, tornando o serviço indisponível (INCAPSULA, 2017).

Figura 8 - HTTP Flood (Comando POST).



Fonte: VERISIGN, 2017.

Este tipo de ataque é um dos utilizados pelo *botnet* Mirai, utilizando dispositivos IoT como câmeras de segurança espalhadas pelo mundo todo. Em 2017 uma nova variante deste *botnet* efetuou um ataque contínuo por 54 horas à uma faculdade nos Estados Unidos. Partindo de quase 10 mil (Figura 9) dispositivos IoT (câmeras de segurança, DVRs e roteadores), este ataque DDoS explorava as portas Telnet (23) e TR-069 (7547) e gerou mais de 2,8 bilhões de requisições (BEKERMAN, 2017).

Figura 9 - Geolocalização de dispositivos utilizados no ataque.

Fonte: BEKERMAN, 2017.

2.4 CONSIDERAÇÕES FINAIS

Ataques à dispositivos IoT estão se tornando cada vez mais frequentes devido ao aumento no uso destes dispositivos sem prestar a devida atenção à segurança contra os mais diferentes ataques e *malwares* existentes. Ataques DDoS conseguem causar grandes impactos nos mais diferentes serviços oferecidos *online*, sendo estes ataques impulsionados principalmente por *botnets* que utilizam muitos dispositivos zumbis, desde *smartphones* e *laptops*, até câmeras de segurança e monitores cardíacos que possuem conexão com a Internet.

É necessário uma atenção do setor de TI quanto à segurança de dispositivos IoT em redes corporativas, pois os usuários desta rede podem utilizar dispositivos próprios que podem possuir *malwares* instalados sem nem perceber a presença dele. O uso de dispositivos pessoais ou profissionais nas organizações aumenta a preocupação com segurança pois o setor de TI não tem o controle sob estes dispositivos. Um dispositivo infectado pode colocar toda a rede corporativa em risco possibilitando que o *malware* espalhe-se pela rede dificultando muito a contenção de sua atividade.

Atualmente, diversas ferramentas de segurança contra ataques e *malwares* à dispositivos IoT estão disponíveis no mercado. Muitas delas estão sendo desenvolvidas e possuem seu

código aberto, bem como há empresas especializadas apenas em desenvolver soluções para este tipo de situação. Algumas destas soluções são apresentadas no capítulo 3.

3. LEVANTAMENTO DE FERRAMENTAS OPEN-SOURCE

Este capítulo relaciona diferentes ferramentas *open-source* disponíveis para tornar os objetivos deste projeto viáveis. O processo de análise e desenvolvimento do *software* será realizado de acordo com as informações descritas neste capítulo.

As ferramentas descritas abaixo foram selecionadas de acordo com o ranking das melhores ferramentas de segurança disponibilizado pelo site SecTools.Org que cataloga ferramentas de segurança desde 2011. Ferramentas não listadas diretamente no ranking foram retiradas de informações contidas na descrição de outras ferramentas ranqueadas.

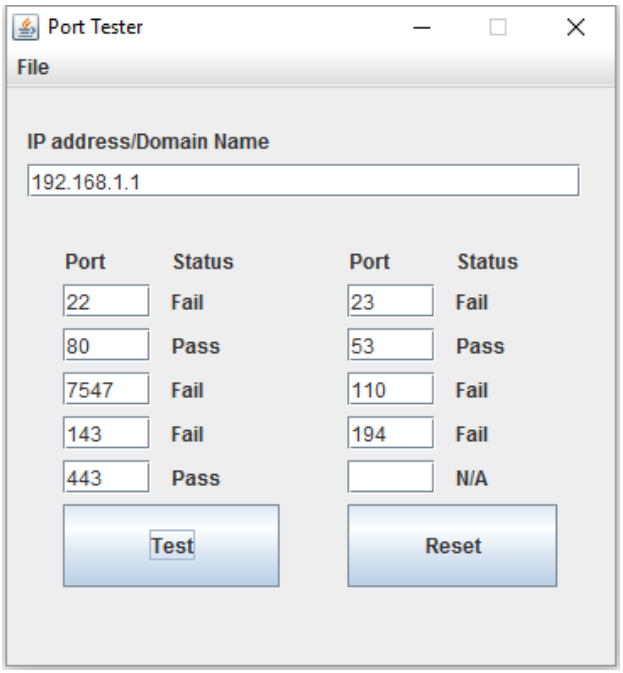
3.1 PORT TESTER 1.0

O Port Tester 1.0[1] (Figura 10) é um *software open-source* desenvolvido por Mateo Marquez Conley em linguagem Java e tem como objetivo testar a conectividade *online* de portas específicas. Este *software* requer que ao menos o Java 6 esteja instalado no ambiente Windows. O software foi atualizado pela última vez em 2016. Ele utiliza uma quantidade moderada de CPU (*Central Process Unit*) e RAM (*Random Access Memory*) e exibe os resultados rapidamente sem apresentar erros.

A interface é simples e para utilizar o Port Tester o usuário precisa informar um IP/domínio de destino e dentre os 10 campos disponíveis informar quais portas deseja testar. O status de cada porta irá aparecer ao lado do campo indicando se há conectividade (*Pass*) ou se a porta está fechada (*Fail*). Casos em que o retorno demorar mais de 300ms serão considerados como portas fechadas.

[1] https://sourceforge.net/projects/porttester/

Figura 10 - Interface de Usuário do Port Tester 1.0.



Fonte: PORT TESTER.

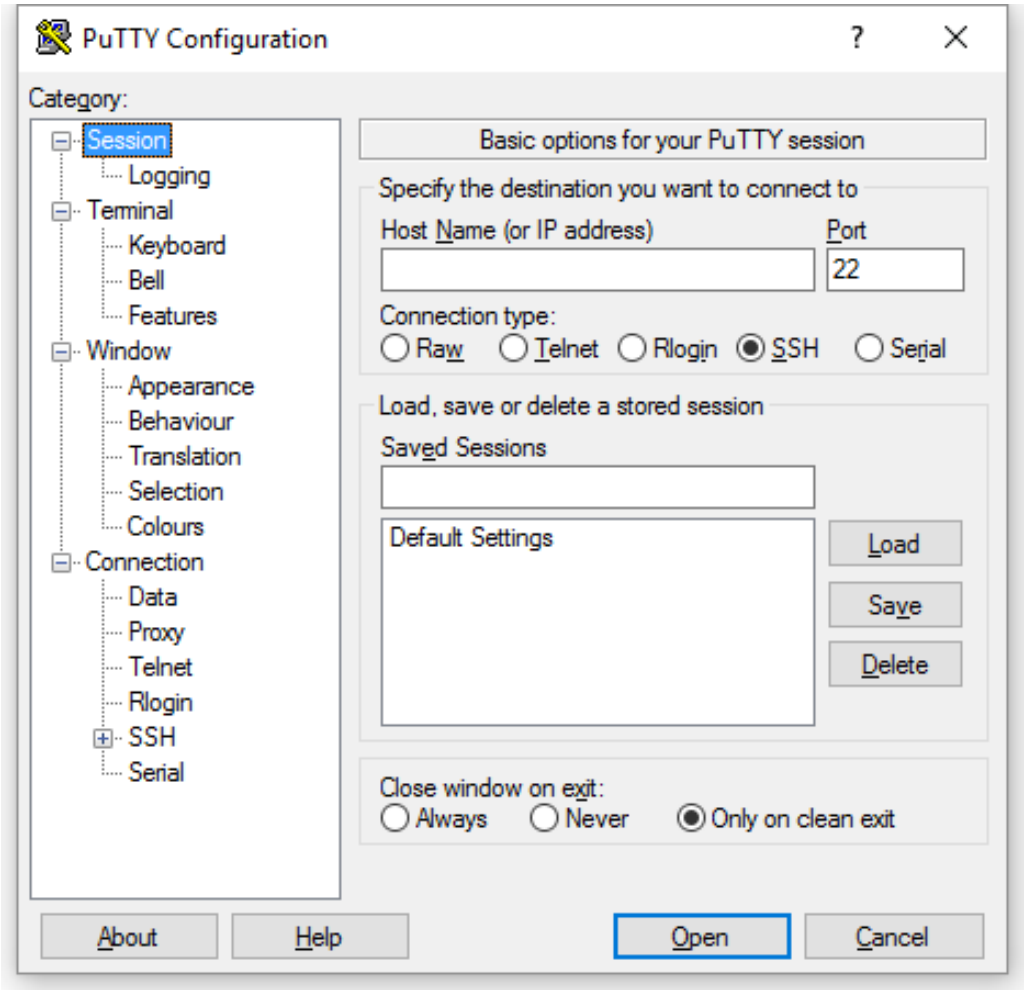
3.2 PUTTY

O Putty[2] (Figura 11) é uma ferramenta *open-source* desenvolvido por Simon Tatham. É um terminal de simulação para atuar como um *cliente* de conexões seguras para os protocolos Raw, Telnet, Rlogin, SSH e Porta Serial.

A interface é simples e para utilizar não é necessário realizar uma instalação, portanto ao deletar a pasta do *software* não ficará dados nos registros do Windows. Originalmente desenvolvido para a plataforma Windows, porém foi portado para outros sistemas operacionais como o Unix e Mac OS. A ferramenta é destinada principalmente para desenvolvedores e administradores de rede, o *software* permite realizar diferentes configurações para cada conexão, seja ela estando salva ou não.

[2] https://www.putty.org/

Figura 11 - Interface de Usuário do Putty.



Fonte: PUTTY.

3.3 THC HYDRA 8.6

O THC-Hydra[3] (Figura 12) é um *software open-source* desenvolvido por Van Hauser. É uma das melhores e mais rápidas ferramentas para a quebra de senhas em diversos protocolos diferentes. O *software* está em constante evolução e é desenvolvido inicialmente para todas plataformas UNIX, porém pode ser compilado e utilizado nas plataformas MacOS, Windows de diversas versões utilizando o *software* Cygwin, Windows versão 10 com o subsistema Ubuntu nativo e para plataformas *mobile* baseadas em Linux, MacOS ou QNX.

Conforme documentação do *software*, na versão 8.6 são suportados diversos protocolos e dentre eles o FTP, HTTP-GET, HTTP-POST, SMTP, SSH (v1 e v2) e Telnet.

[3] https://www.thc.org/thc-hydra/

A interface é por linhas de comando, porém a documentação disponibilizada junto com o código fonte possui exemplos de diversas opções de uso. Para utilizar é necessário apenas digitar “hydra” na console e aparecerá a sintaxe que deve ser informada, bem como suas opções de uso. No uso do protocolo Telnet é recomendado utilizar outro *software* para verificar se a porta 23 está aberta antes de iniciar um ataque garantindo que não dê resultados falso-positivo.

Figura 12 - Interface do THC Hydra.

Fonte: THC Hydra.

3.4 JOHN THE RIPPER

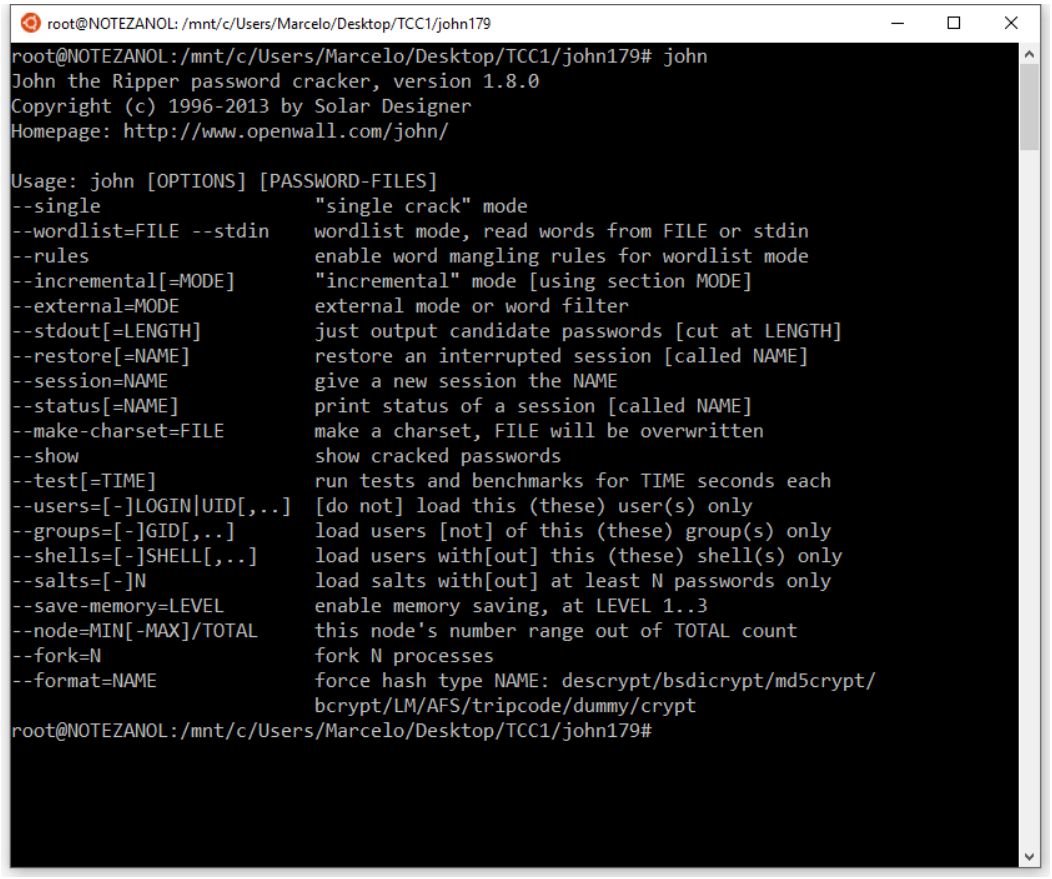
O John the Ripper[4] (Figura 13) é uma ferramenta desenvolvida pelo russo Solar Designer. Ela é *open source* e desenvolvida em linguagem C. A ferramenta está disponível nas plataformas UNIX/Linux e Mac OS X. Seu objetivo inicial é detectar senhas fracas na

[4] http://www.openwall.com/john/

plataforma Unix e na plataforma Windows, é possível identificar senhas utilizando *hashs* de senhas.

Com uma documentação completa com exemplos e instruções de instalação, o *software* é de fácil uso e não é necessário um grande conhecimento para utilizá-lo. Há disponível quatro opções principais para a quebra de senhas, sendo elas, ataque com dicionário, ataque simples, ataque incremental e ataque externo descritos em sua documentação.

Figura 13 - Interface do John the Ripper.



Fonte: John the Ripper.

3.5 ANGRY IP SCANNER

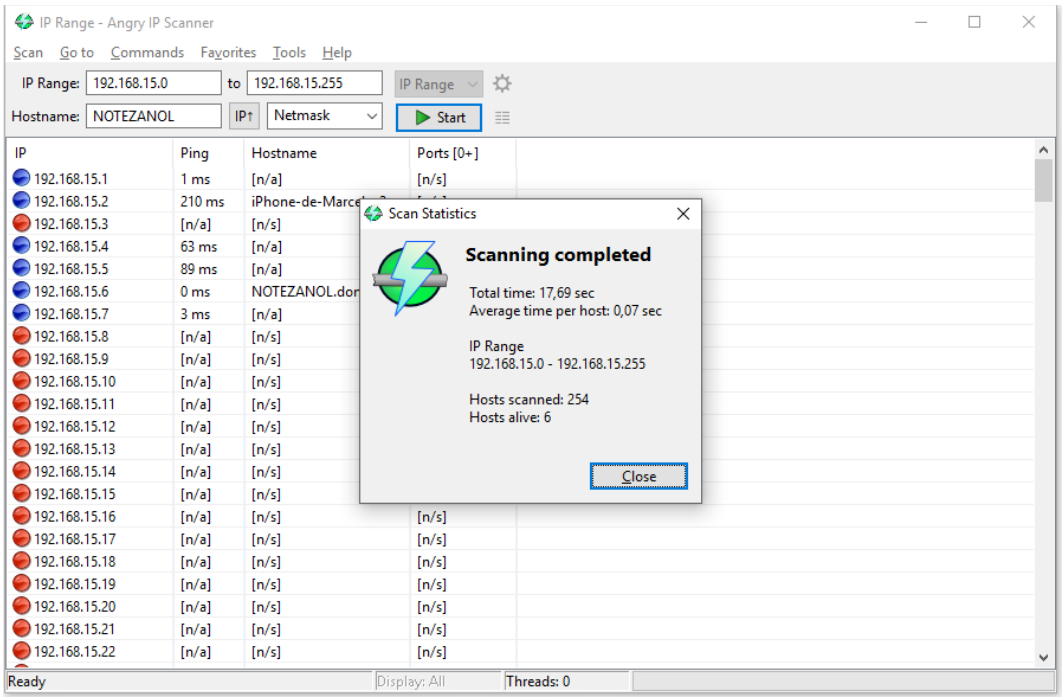
O Angry IP Scanner[5] (Figura 14) é um *software open-source* desenvolvido por Anton Keks em linguagem Java e tem como objetivo escanear IPs e portas em dispositivos conectados em uma rede local. O *software* pode ser utilizado em plataformas Windows, Linux e Mac OS.

5 http://angryip.org/

Sua alta velocidade de escaneamento está diretamente ligada ao uso de numerosas *threads* para executar os procedimentos necessários.

A interface é simples e campos como uma faixa de IPs para serem escaneados já vem previamente preenchidos. Em poucos segundos é possível visualizar todos dispositivos ativos na rede e se necessário resolver o nome dos hosts, portas abertas e seus endereços MAC. Diversos *plug-ins* podem ser utilizados junto com o *software* e o usuário pode optar em exportar os resultados para um arquivo CSV, TXT ou XML.

Figura 14 - Interface do Angry IP Scanner.



Fonte: ANGRY IP SCANNER.

3.6 CONSIDERAÇÕES FINAIS

Ao finalizar a pesquisa por ferramentas *open-source*, conclui-se que dentro as pesquisadas não há ferramentas completas que possuam ênfase à prevenção de ataques DDoS para aplicativos IoT por tratar-se de um assunto muito específico e recente.

As ferramentas elencadas nesse capítulo são compatíveis para tornar a proposta deste projeto viável e atingir os objetivos propostos. As ferramentas *open-source* selecionadas, ao serem integradas irão produzir um software completo para a análise de segurança de dispositivos IoT contra ataques das mais diversas fontes. O Angry IP Scanner ao ser

incorporado no *software* escaneia a rede local localizando o IP dos dispositivos IoT. Com os IPs identificados o Port Tester pode viabiliza o teste de portas vulneráveis e com essa informação o Putty e o THC Hydra conseguem efetuar a conexão a partir de listas de senhas padrões. Para conseguir quebrar senhas de dispositivos como computadores e notebooks o John de Ripper agrega valor ao *software*.

A escolha das ferramentas listadas foi realizada com o intuito de desenvolver um *software* que auxilie administradores de redes à garantir uma maior segurança aos seus dispositivos IoT, indo de encontro com o objetivo do projeto.

4. DEFINIÇÃO DO SOFTWARE

Este capítulo detalha o *software* a ser desenvolvido para integrar ferramentas de segurança disponíveis no mercado com a ferramenta BYOD Manager Toolkit[6]. Para tornar a estruturação possível faz-se necessário o estudo e compreensão dos códigos fontes das ferramentas selecionadas e da ferramenta base que terá suas funcionalidades aumentadas.

Na seção 4.1 é apresentado o software BYOD Manager Toolkit desenvolvido por Perini (2017). A seção 4.2 define os incrementos que foram realizados no software de Perini (2017) nesse Trabalho de Conclusão de Curso.

4.1 BYOD MANAGER TOOLKIT

O BYOD Manager Kit é um *software* de integração de ferramentas *open-source* desenvolvido em linguagem Java por Perini (2017) e tem como objetivo auxiliar o profissional de TI na administração e segurança de redes BYOD (*Bring Your Own Device*).

Redes BYOD são redes corporativas nas quais é permitido o uso de dispositivos pessoais para o desenvolvimento de tarefas profissionais, aumentando a complexidade na segurança de rede que deve ser analisada e implantada.

O software (Figura 15) é dividido nas categorias Análise e Descobrimento, Segurança e Senhas e Administração da Rede, na qual estão inclusas diferentes ferramentas *open-source*, cada uma com funcionalidades importantes melhorando a qualidade do software. As ferramentas estão agrupadas de acordo com a sua funcionalidade: análise e descobrimento, segurança e senhas e administração de redes.

[6] O *software* não possui site próprio ou repositório de fontes. Sua documentação é referenciada no documento de Integração de Ferramentas de Administração e Segurança BYOD.

Figura 15 - Interface do BYOD Manager Kit.

Fonte: BYOD Manager Kit (PERINI, 2017).

Na categoria de Análise e Descobrimento estão integradas as ferramentas *NetworkMiner* e o *jNetMap*. O *NetworkMiner* é um *sniffer* de rede passivo, desenvolvido em linguagem C#. O *JNetMap* é uma ferramenta gráfica de monitoramento e documentação de redes desenvolvido em linguagem Java.

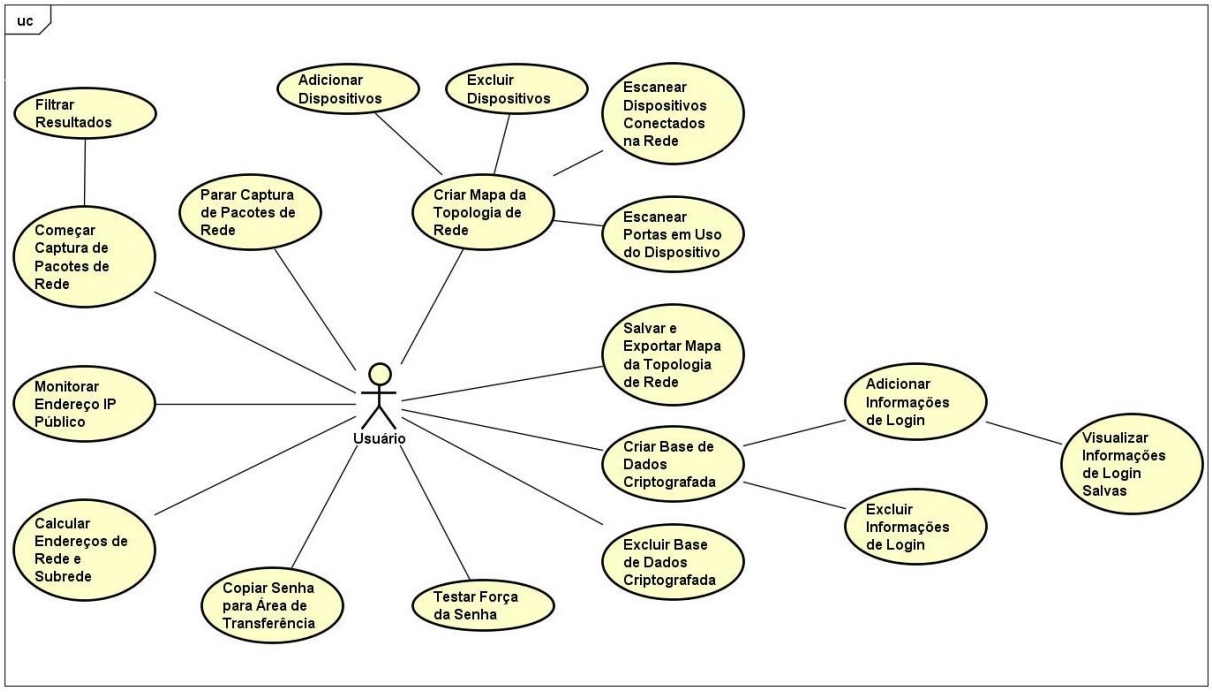
Na categoria de Segurança e Senhas estão integradas as ferramentas *Universal Password Manager* e o *Password Strengh Meter*. O *Universal Password Manager* é uma ferramenta para gerenciamento de usuários e senhas desenvolvida em Java. O *Password Strengh Meter* é uma biblioteca também desenvolvida em Java com uma funcionalidade para medir a força das senhas e também informar a quantidade de iterações necessárias para a quebra da senha através de algoritmos de força bruta.

Na categoria de Administração de Rede estão integradas as ferramentas *IP Monitor* e o *Net Calculator*. O *IP Monitor* é uma ferramenta de monitoramento de IPs públicos desenvolvida em Java, seu objetivo principal é notificar quando há alguma mudança no IP que está sendo monitorado. O *NetCalculator* é uma ferramenta desenvolvida em linguagem C# que tem como principal objetivo auxiliar na configuração de uma subrede, já que cálculos de endereçamentos de subredes, apesar de não serem complexos, tomam tempo do administrador da rede.

Com as funcionalidades das ferramentas listadas acima, em um único programa, o profissional de TI consegue administrar de forma mais simples redes BYOD.

A figura 16 apresenta um diagrama de casos de usos, na qual lista as diferentes funcionalidades todas centralizadas em um único programa permitindo também que o usuário utilize diferentes recursos diretamente entre os programas *open-source.*

Figura 16 – Diagrama Casos de Uso do BYOD Manager Kit.

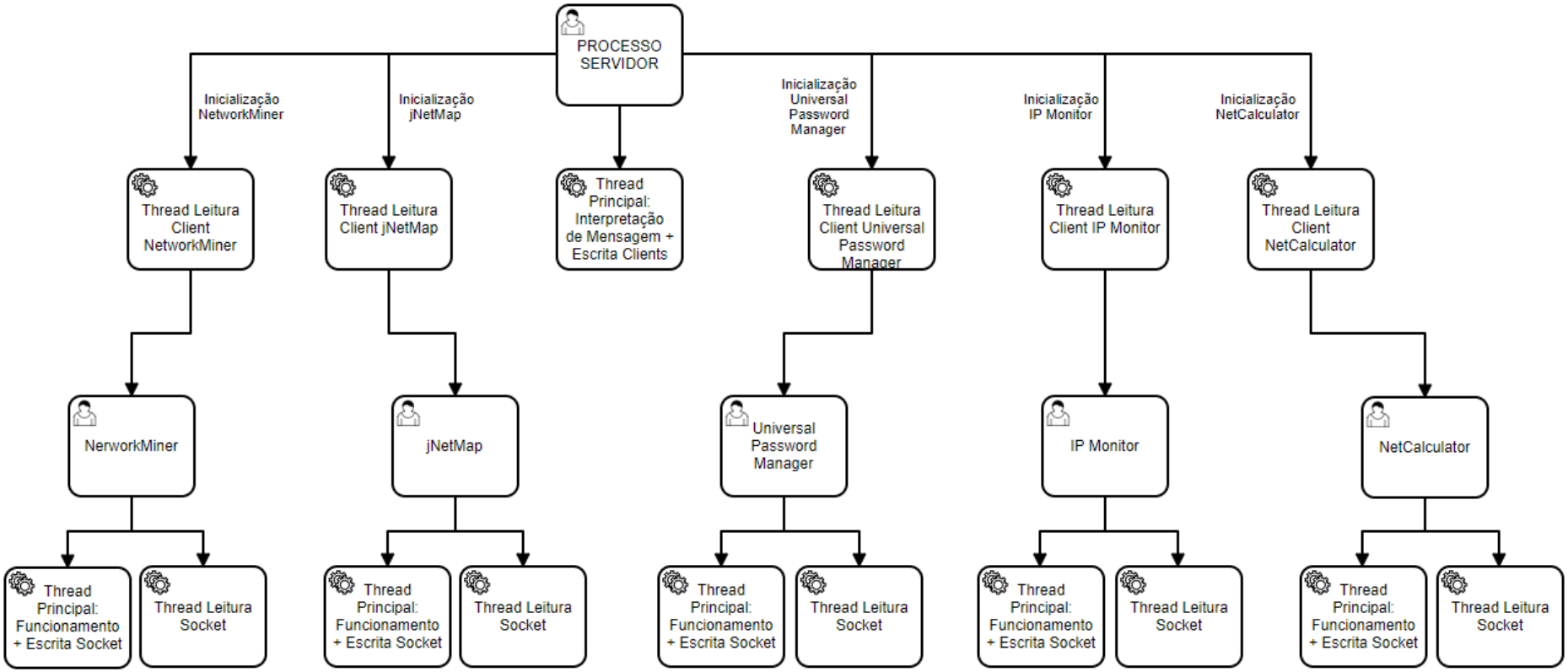


Fonte: BYOD Manager Kit (PERINI, 2017).

4.1.1 FLUXO DE COMUNICAÇÃO ENTRE PROCESSOS

A comunicação entre os *softwares* é realizada através utilização de *threads* e *sockets*. O processo servidor é inicializado juntamente com a interface principal da suíte. Os clientes são inicializados e conectados no momento em que o usuário abrir as ferramentas. Conforme ilustrado na figura 17, os nodos com engrenagens representam as *threads* e os nodos com usuários representam as ferramentas integradas.

Figura 17 – Fluxograma de Comunicação do BYOD Manager Kit.



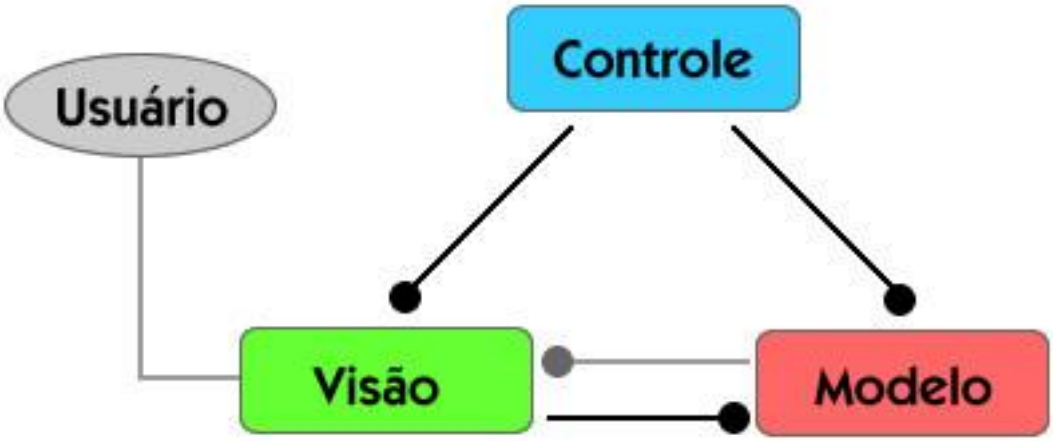
Fonte: Perini, 2017.

4.1.2 MODELO MVC

O *software* foi desenvolvido utilizando o modelo MVC (*Model View Controller*). O modelo MVC é estruturado através de três camadas que podem interagir entre si. Definições de regras de negócio, persistências e dados são implementadas na camada Modelo (*model*). Para interação do usuário e inserção de novas instruções é utilizada a camada Visão (*view*). Para interpretação das instruções da camada Visão é utilizada a camada Controlador (*controller*), a qual também é responsável por conferir se as instruções estão corretas, e enviar para o modelo para que posteriormente receba comandos da camada Modelo e as envie para a camada Visão (GULZAR, 2002).

A Figura 18 ilustra o modelo de camadas MVC.

Figura 18 - Modelo de Camadas MVC.



Fonte: CELESTINO, 2014.

Na camada modelo do BYOD Manager Kit ficam os arquivos fonte das ferramentas *open-source* que foram selecionadas. As ferramentas que foram desenvolvidas em Java foram salvas em formato JAR (Java Archive – Arquivo Java) e adicionadas a esta camada. Com exceção da ferramenta Password Strength Meter que teve os arquivos incluídos em formato Java. As ferramentas que foram desenvolvidas em C# tiveram seus respectivos arquivos executáveis e bibliotecas também (formato DLL3) incluídas na camada modelo. Em nenhuma ferramenta o código fonte desta camada foi alterado. A arquitetura criada pelos desenvolvedores oficiais foi mantida.

Na camada visão foram criadas as interfaces gráficas do *software*. Foram desenvolvidas diferentes classes Java, responsáveis por painéis de mensagens, caixas de diálogos e janelas principais. Todas essas interfaces foram desenvolvidas utilizando o Eclipse com o *plug-in* WindowBuilder.

A camada controle é responsável pelo gerenciamento da quantidade de ferramentas que estão abertas, pela chamada de inicialização das ferramentas, além do gerenciamento de comunicação entre processos clientes e o processo servidor. Por fim, a documentação original das ferramentas também é tratada nesta camada.

Para que a comunicação entre as ferramentas e o processo servidor ocorresse, foram realizadas algumas alterações no código-fonte destas ferramentas. Para as ferramentas desenvolvidas em Java e C#, foram criadas duas classes "SocketClient", uma para cada linguagem. Como as informações são enviadas ao processo servidor e recebidas pelo processo cliente em formato String, o ambiente Java e o ambiente C# conseguem trocar dados sem a necessidade de um tratamento específico.

A classe "SocketClient" foi adicionada aos projetos *open-source* respeitando o modelo MVC. Como a maioria das ferramentas selecionadas possui esta arquitetura de software, a classe "SocketClient" foi adicionada à camada controle destes projetos. Para seguir o padrão de desenvolvimento, a camada controle foi criada nos projetos que não utilizavam o modelo MVC.

Além disso, todas as demais modificações e novas funcionalidades, como a criação de um filtro de resultados no NetworkMiner, a fila de importação de dispositivos no jNetMap, entre outras, foram feitas nas suas respectivas camadas de *software*.

Foram necessárias alterações na camada visão do projeto NetworkMiner, pois um novo menu foi adicionado à tela principal da ferramenta. O mesmo ocorreu nas ferramentas jNetMap e Universal Password Manager.

A ferramenta NetCalculator não necessitou da inclusão de novos elementos na interface gráfica, mas algumas mudanças, como por exemplo o tratamento de eventos, foram realizadas na camada visão.

A ferramenta Password Strength Meter não possui uma interface gráfica. A camada visão do *software* de integração foi utilizada para criar uma interface que pudesse facilitar o uso da biblioteca pelo usuário final.

A ferramenta IP Monitor sofreu alterações somente na camada controle com a adição da classe "SocketClient".

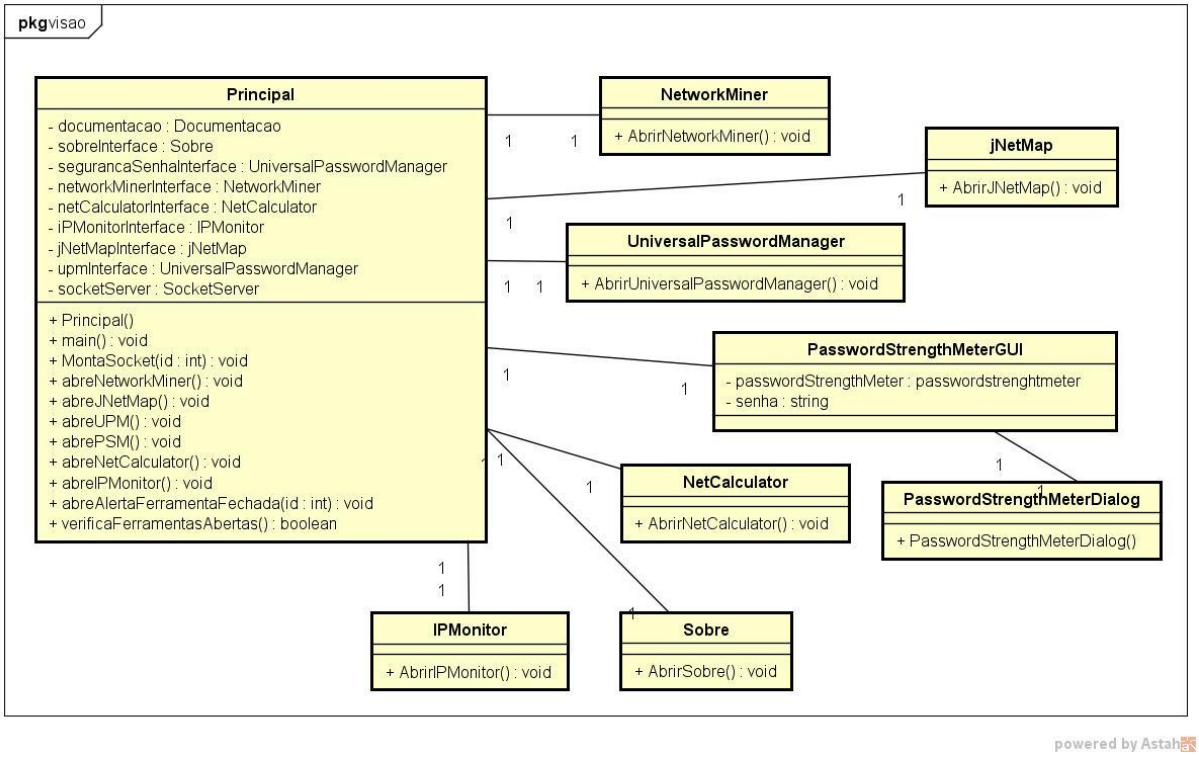
4.1.3 DIAGRAMA DE CLASSES

Os diagramas de classes foram estruturados de acordo com as três camadas do MVC. O diagrama criado para a camada visão possui as principais classes e métodos utilizados no desenvolvimento da interface. O diagrama contendo as informações mais importantes foi criado para a camada controle. Para a camada modelo foi criado um diagrama contendo modificações e implementações realizadas nas ferramentas *open-source* (PERINI, 2017).

A Figura 19 representa o diagrama de classes da camada visão. Os objetivos específicos das classes mais importantes são:

- Principal: classe que contém a função *main*. Seu funcionamento consiste em exibir a interface principal de usuário e interpretar os eventos de cliques do usuário. Responsável pela troca de dados com a camada controle.
- Sobre: classe responsável por mostrar ao usuário mais detalhes sobre o *software* e as licenças de *software* livre.
- NetworkMiner: classe responsável por mostrar uma pequena janela de carregamento enquanto o computador trabalha para abrir a ferramenta *NetworkMiner*.
- PasswordStrengthMeterGUI: classe responsável por exibir uma interface gráfica que foi desenvolvida exclusivamente para a biblioteca *Password Strength Meter*.
- Demais classes seguiram o modelo da classe NetworkMiner porém com a janela de carregamento desativada.

Figura 19 - Diagrama de Classes da Camada Visão.

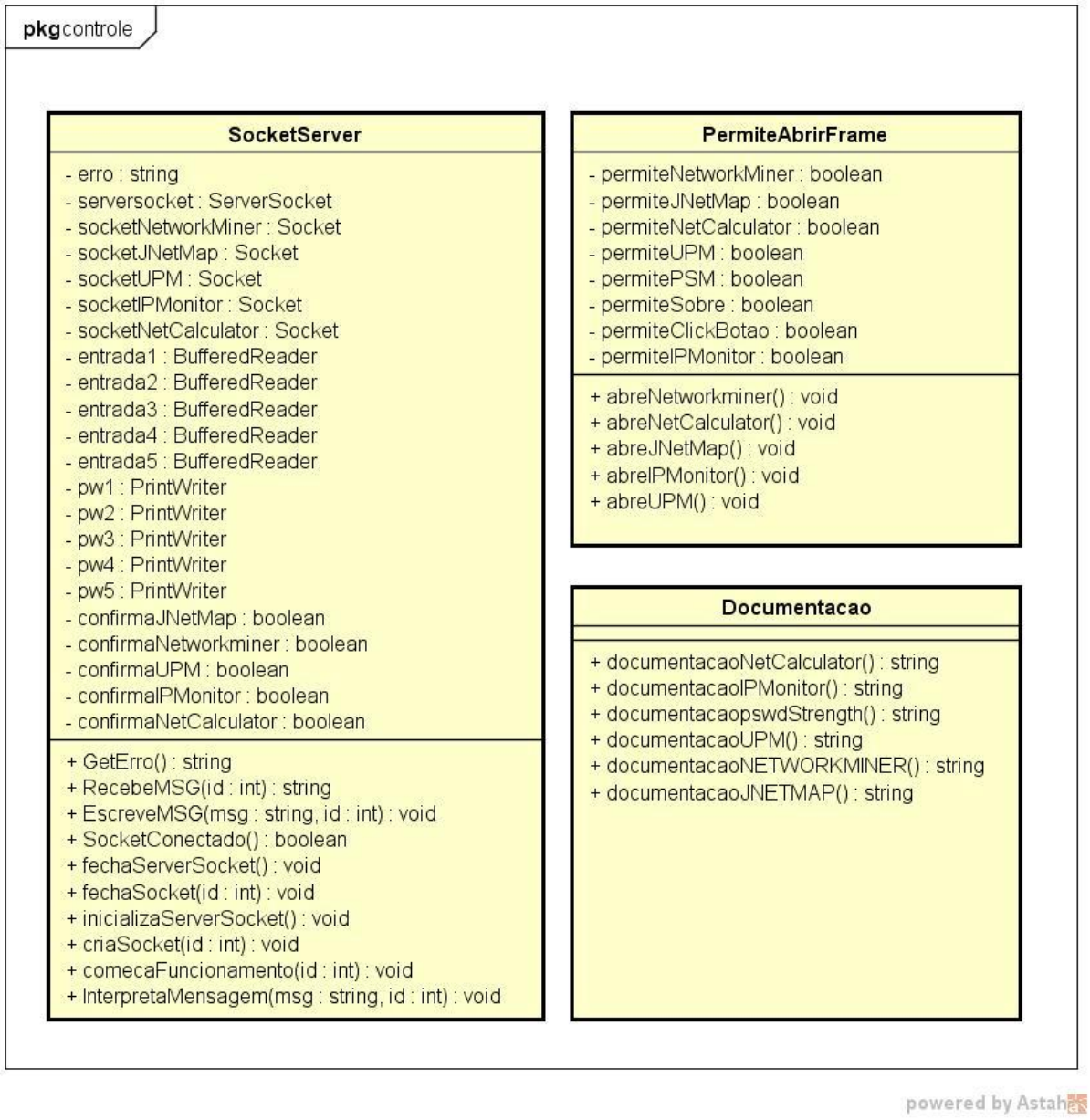


Fonte: Perini, 2017.

A Figura 20 representa o diagrama de classes da camada controle. Os objetivos específicos das classes mais importantes são:

- SocketServer: Classe responsável por criar o *server socket*, estabelecer a conexão dos *clients sockets*, enviar mensagens e interpretar os dados recebidos.
- PermiteAbrirFrame: Classe responsável pelos *flags* que garantem que uma ferramenta não seja inicializada mais de uma vez simultaneamente e funções que realizam as chamadas de inicialização das ferramentas.
- Documentacao: Classe responsável por detectar se o sistema possui um navegador de internet ou leitor de PDF para abrir sites ou documentos de algumas ferramentas.

Figura 20 - Diagrama de Classes da Camada Controle.



Fonte: Perini, 2017.

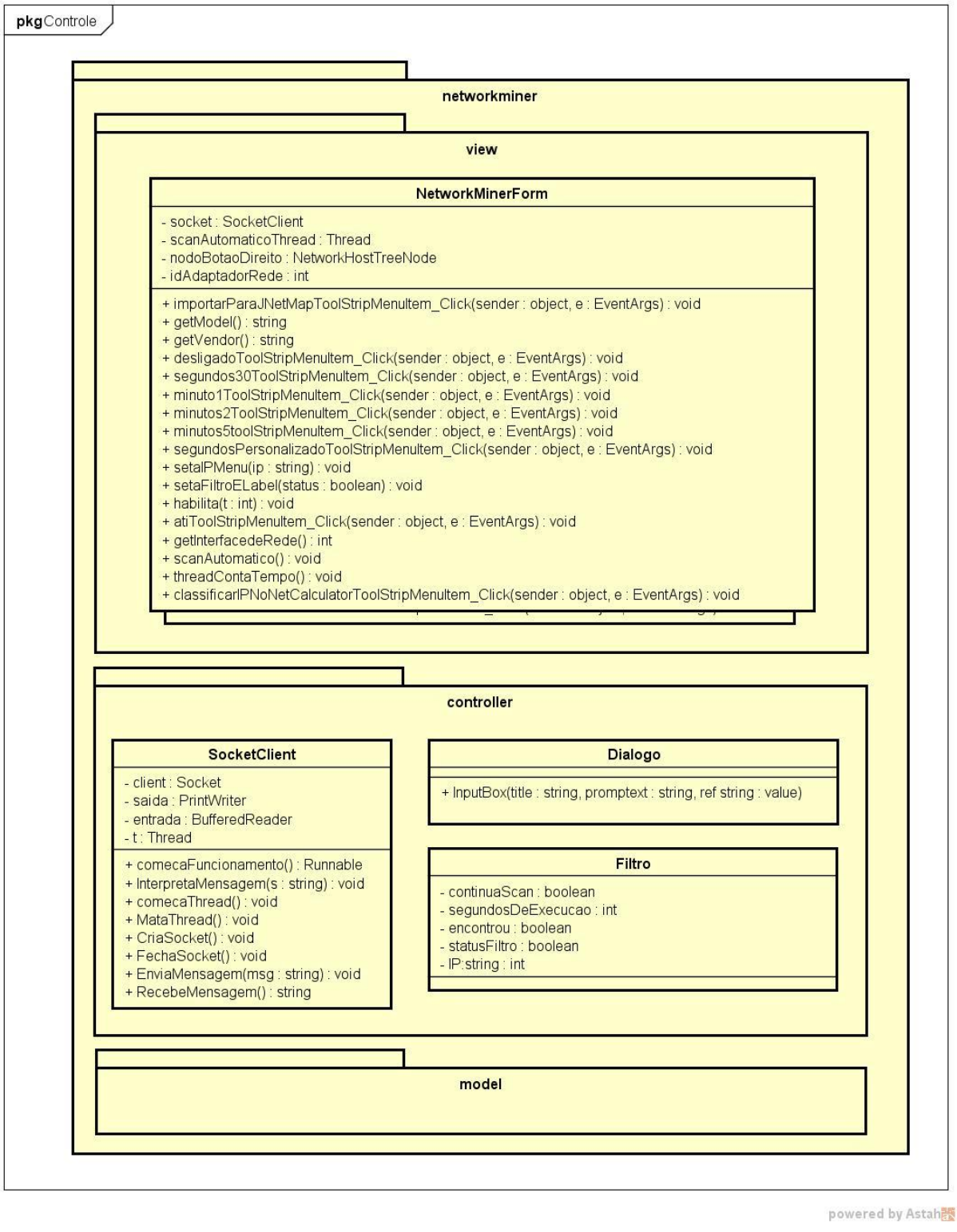
As Figuras 21, 22, 23 e 24 representam o diagrama de classes da camada modelo. Os objetivos específicos das classes mais importantes são:

- SocketClient (NetworkMiner, jNetMap, IP Monitor, NetCalculator, Universal Password Manager): Classe responsável por se conectar com o servidor criado na camada controle, enviar e receber informações e por interpretar os dados recebidos.
- Filtro (NetworkMiner): Classe responsável pela filtragem dos resultados exibidos na interface principal e por controlar o tempo de escaneamento automático.

- FilaDevices (jNetMap): Classe responsável por armazenar em fila os dados recebidos do *NetworkMiner*. Esta classe possibilita que o usuário importe os dispositivos respeitando o modelo FIFO (*First in, First Out* – Primeiro a Entrar, Primeiro a Sair).

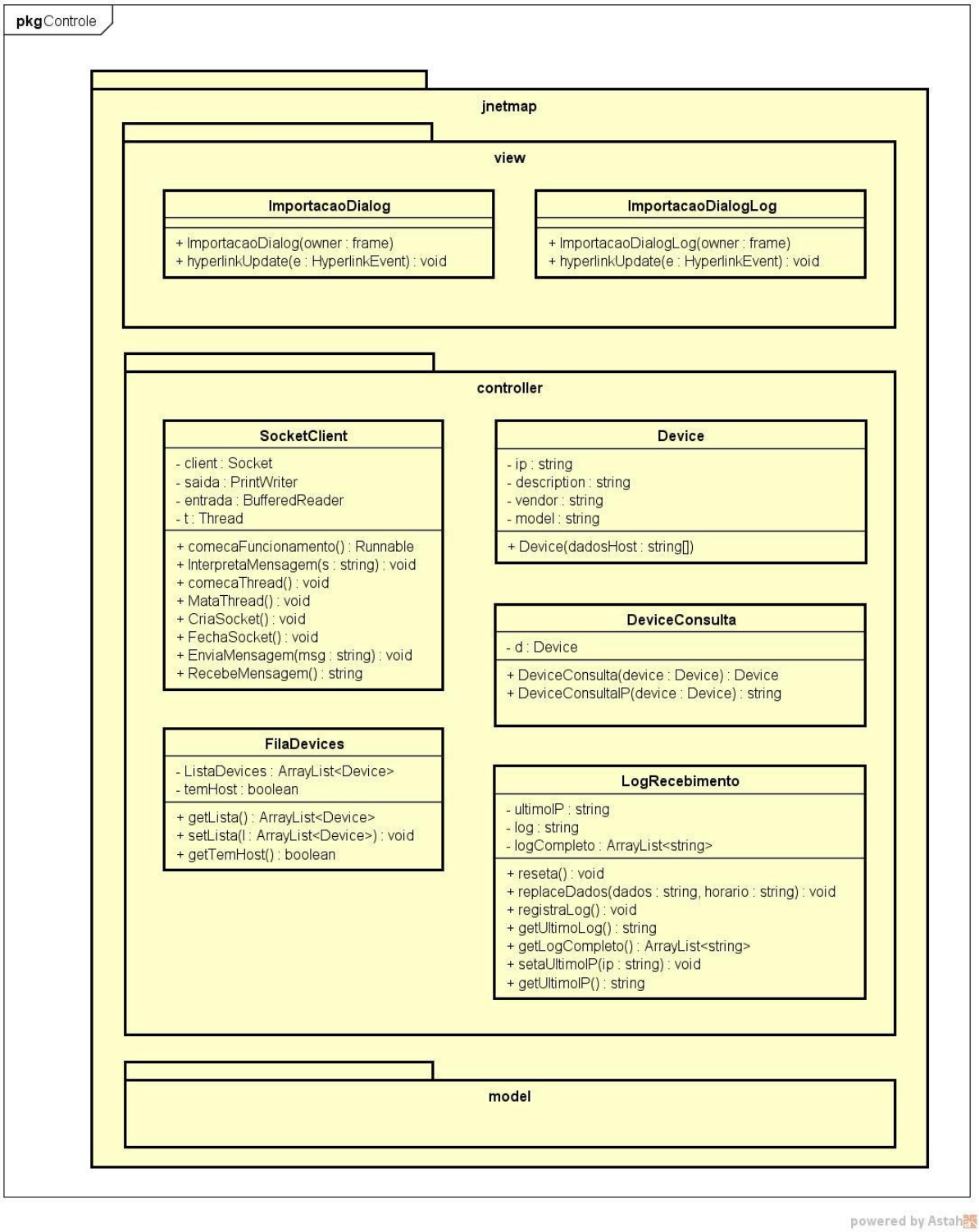
- PasswordStrengthMeterGUI (Universal Password Manager): essa classe é responsável por exibir uma interface gráfica que foi desenvolvida exclusivamente para a biblioteca *Password Strength Meter*. Como citado na seção 5.2.1, essa biblioteca foi importada duas vezes, sendo uma delas no projeto do *Universal Password Manager*.

Figura 21 - Diagrama de Classes do NetworkMiner.



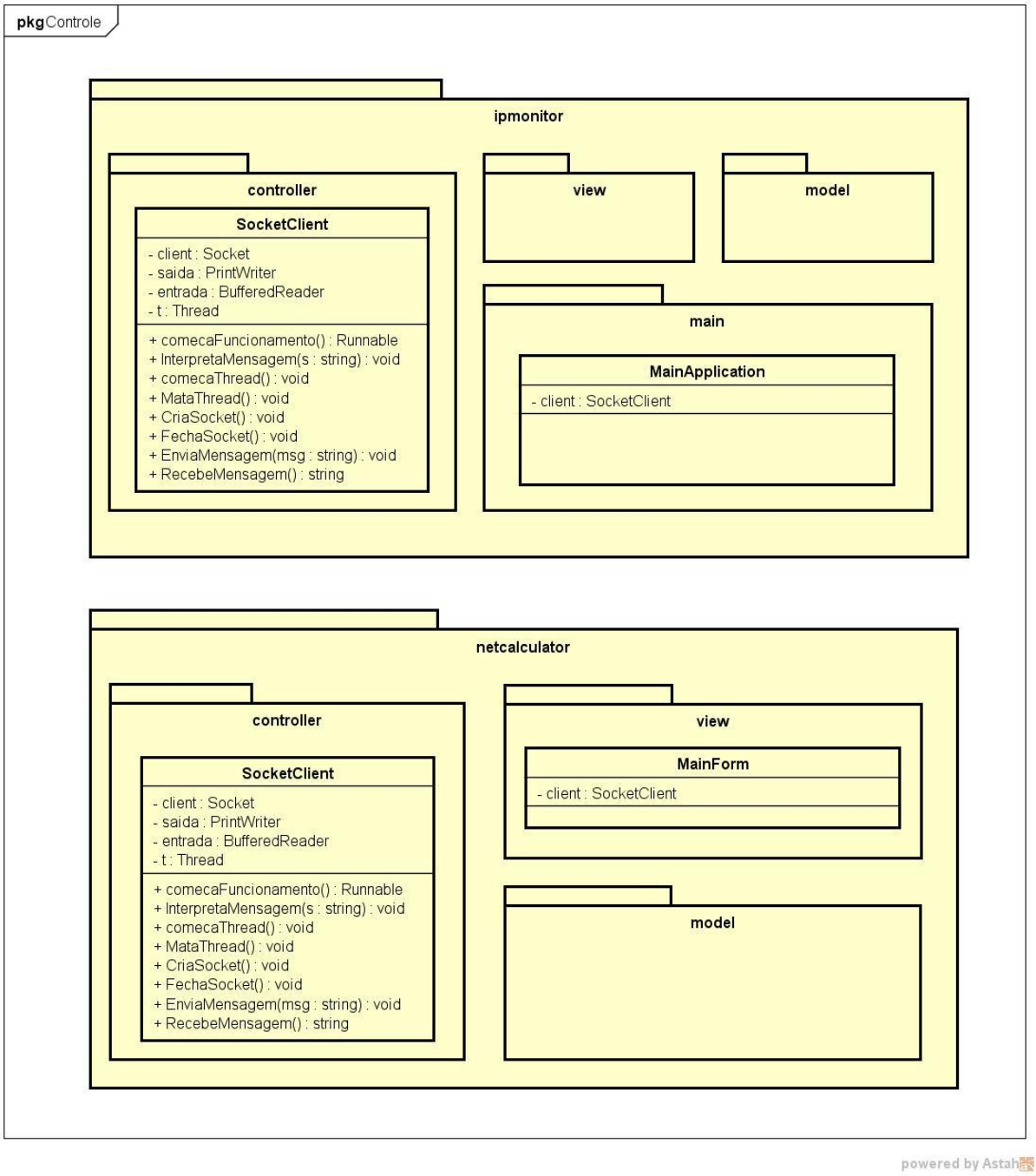
Fonte: Perini, 2017.

Figura 22 - Diagrama de Classes do jNetMap.



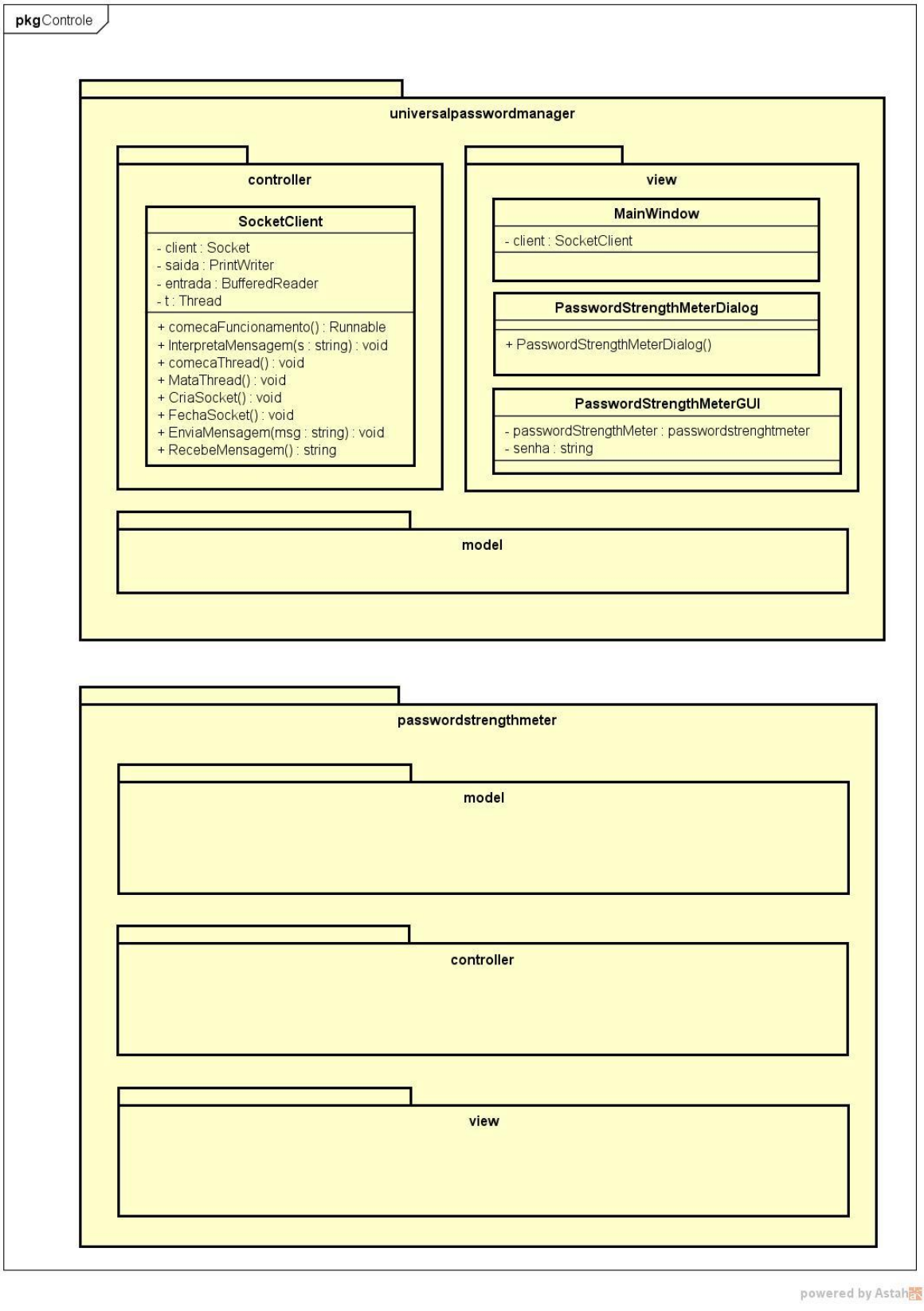
Fonte: Perini, 2017.

Figura 23 - Diagrama de Classes do IP Monitor e do NetCalculator.



Fonte: Perini, 2017.

Figura 24 - Diagrama de Classes do Universal Password Manager e do Password Strength Meter.



Fonte: Perini, 2017.

4.2 PROPOSTA DE SOLUÇÃO

Para realizar o desenvolvimento dos incrementos no *software* é utilizada a linguagem de programação Java. O ambiente de desenvolvimento integrado utilizado é o Eclipse. A escolha desta IDE (*Integrated Developtment Environment*) deve-se ao prévio conhecimento e experiência do autor, sendo uma ferramenta que possui atualizações e melhorias constantes. A escolha da linguagem Java foi escolhida para manter a mesma linguagem utilizada no BYOD Manager Kit.

A plataforma Java é composta por dois componentes principais, a API (*Application Programming Interface*) e a JVM (*Java Virtual Machine*). Esta estrutura permite que aplicações desenvolvidas sejam compatíveis com diferentes sistemas operacionais, como Windows, UNIX e Mac OS, garantindo uma ampla compatibilidade do *software*. JVM é um software utilizado para carregar e executar os aplicativos Java, convertendo os bytecodes em código executável de máquina. A API é uma grande coleção de componentes de *software* agrupados em bibliotecas de classes e interfaces relacionadas (pacotes) e estruturas de manipulação de dados e arquivos (ORACLE, 2017).

Para o desenvolvimento em Java da aplicação proposta neste trabalho, o uso da ferramenta Eclipse é essencial. A versão escolhida é a Eclipse Oxygen (4.7) publicada no dia 28 de junho de 2017. O uso do *plug-in* WindowBuilder disponível para o Eclipse é importante para tornar a montagem de telas ágil e produtiva, na qual são utilizadas as principais APIs gráficas disponíveis (ECLIPSE, 2017).

O desenvolvimento exige um ambiente com uma configuração adequada. A arquitetura necessária é composta por um ambiente de desenvolvimento e um ambiente de testes.

Para o desenvolvimento é necessário o uso da plataforma Windows de 64 bits, através do uso do Eclipse Oxygen. Os requisitos mínimos para o funcionamento do Java no Windows são: processador Pentium 2 266 MHz, 128MB de memória RAM e 181MB de espaço livre em disco (ORACLE, 2016). Os requisitos mínimos para o Eclipse Oxygen são: 300MB de espaço livre em disco, 1GB de memória RAM e Java 8 ou superior (ECLIPSE, 2017).

Para a execução dos testes é necessário o uso da plataforma Windows versão 10 de 64 bits com o subsistema Linux instalado e o pacote do THC Hydra instalado neste subsistema. No ambiente de testes será possível executar o *software* e analisar os resultados obtidos após execução das ferramentas propostas.

Para a execução das ferramentas *open-source* e do software base que terá as ferramentas integradas é necessário que o sistema ofereça:

- Java versão 8 ou superior
- GCC (GNU *Compiler Collection*)
- Microsoft .NET versão 2.0 ou superior
- Sistema Operacional Ubuntu

Algumas das ferramentas são desenvolvidas em linguagens orientadas a objeto, o que faz ser necessário uma possível refatoração de trechos de código utilizando a IDE apropriada.

A fim de manter uma compatibilidade para uso do *software* em diferentes plataformas será mantido o uso do *framework* JNA (*Java Native Access*). O *framework* permite que programas Java acessem bibliotecas nativas e escrita em outras linguagens sem a utilização do JNI (*Java Native Interface*). A vantagem do uso do JNA ao invés do JNI é a possibilidade de executar uma integração mais simples, sem exigir muitas configurações e sem poluir o código das aplicações com chamadas e tipos específicos do JNI (JNA, 2017).

4.2.1 DEFINIÇÃO DOS REQUISITOS

Os incrementos a serem incluídos no BYOD Kit Manager tem por objetivos: oferecer um complemento que agregue valor ao *software*, funcionar corretamente, ser simples de operar, manter uma confiabilidade nos dados e evitar possíveis falhas. Seu código deve ser escrito e documentado de maneira clara facilitando inclusões de novas funcionalidades e manutenções.

Os requisitos funcionais do *software* são:

- Permitir a validação de portas abertas de dispositivos IoT conectados na rede.
- Permitir efetuar validação da senha de dispositivos IoT através de listas de senhas utilizadas pelo Mirai para ataques DDoS.
- Permitir avaliar se os dispositivos IoT estão protegidos contra ataques DDoS.
- Permitir o escaneamento de IPs ativos na rede local sem necessidade de criar o mapa de rede.
- Permitir a quebra de senhas através de seu *hash*.

Os requisitos não funcionais são:

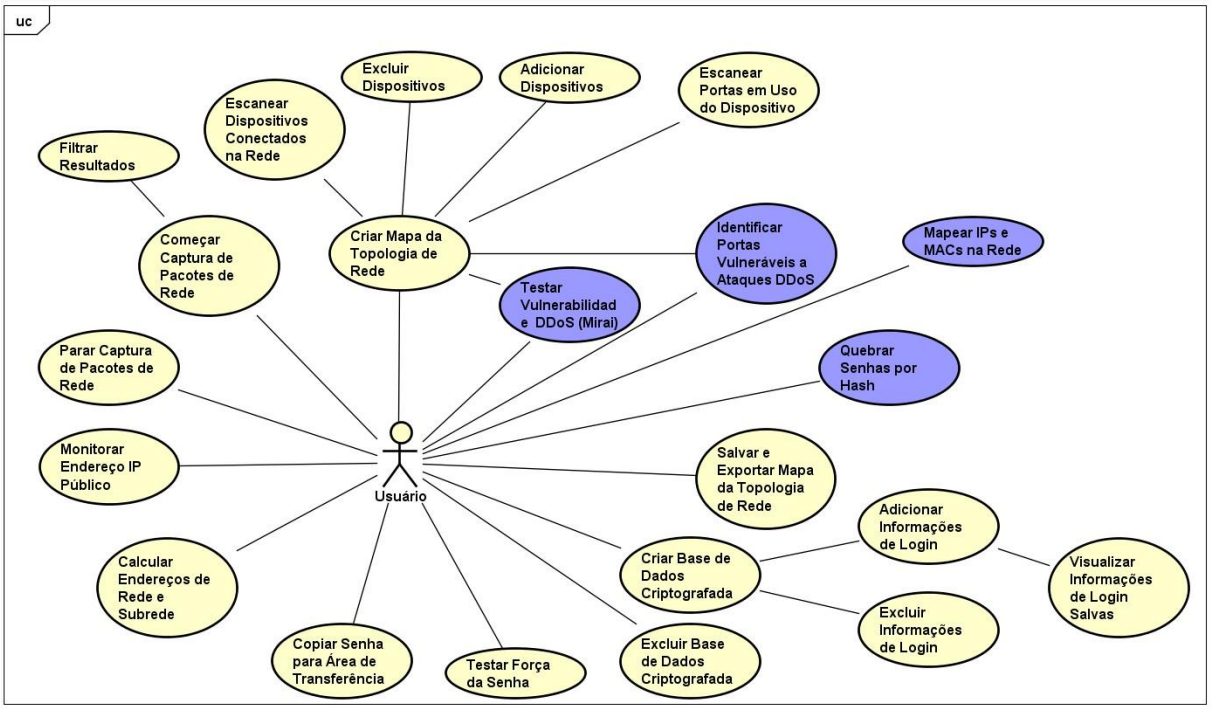
- Manter a compatibilidade com a plataforma: garantir que o *software* funcione corretamente no ambiente Windows.
- Manter a confiabilidade dos resultados: garantir que os resultados obtidos sejam verdadeiros.

- GUI (*Graphical User Interface* – Interface Gráfica de Usuário) de fácil utilização: oferecer uma interface de usuário *desktop* que seja acessível e de baixa complexidade.
- Documentação de Auxílio: oferecer a documentação e manuais para evitar a necessidade de treinamentos para utilização do *software* e facilitar a integração de novas funcionalidades.

4.2.2 DIAGRAMAS DE CASOS DE USO

O diagrama de casos de uso do software BYOD Manager Kit e as novas funcionalidades inseridas para prevenção de ataques DDoS no software são mostrados na Figura 25. Os incrementos estão destacados na cor azul (balões escuros). O detalhamento dos novos caso de uso são definidos nos quadros 1 a 5.

Figura 25 - Casos de Uso do software.



Fonte: Autoria própria.

Quadro 1- Caso de Uso 01.

Caso de Uso 01	Mapear IPs e MACs na Rede
Descrição	O usuário deverá utilizar a ferramenta de mapeamento de IPs e MACs na Rede.
Atores envolvidos	Usuário do sistema.
Pré-condição	Usuário deve navegar pelos menus da interface até a ferramenta.
Cenário Principal de Sucesso	1. O usuário deve navegar até a opção de Análise e Descobrimento de Rede.
	2. O usuário deve selecionar para iniciar a varredura.
	3. O sistema realiza o escaneamento e mostra os resultados na tela.
Cenário Secundário de Falha	1. A ferramenta retorna a mensagem de erro e não executa o mapeamento.

Fonte: Autoria própria.

Quadro 2- Caso de Uso 02.

Caso de Uso 02	Identificar Portas Vulneráveis a Ataques DDoS
Descrição	O usuário deverá utilizar a ferramenta de mapeamento de Verificação de Portas Vulneráveis.
Atores envolvidos	Usuário do sistema.
Pré-condição	Usuário deve navegar pelos menus da interface até a ferramenta.
Cenário Principal de Sucesso	1. O usuário deve navegar até a opção de Administração de Rede.
	2. O usuário deve informar o dispositivo IoT para verificar as portas vulneráveis.
	3. A ferramenta executa o processo e retorna o status de cada porta analisada.
Cenário Secundário de Falha	1. A ferramenta retorna a mensagem de erro e não executa o processo.

Fonte: Autoria própria.

Quadro 3- Caso de Uso 03.

Caso de Uso 03	Identificar Portas Vulneráveis a Ataques DDoS
Descrição	O usuário deverá utilizar a ferramenta de mapeamento de Verificação de Portas Vulneráveis.
Atores envolvidos	Usuário do sistema.
Pré-condição	Usuário ter o Mapa da Topologia de Rede criado.
Cenário Principal de Sucesso	1. O usuário deve navegar até a opção de Análise e Descobrimento de Rede.
	2. O usuário deve abrir o Mapa da Topologia de Rede no jNetMap.
	3. O usuário deve selecionar o dispositivo IoT para verificar as portas vulneráveis.
	4. A ferramenta executa o processo e retorna o status de cada porta analisada
Cenário Secundário de Falha	1. A ferramenta retorna a mensagem de erro e não executa o processo.

Fonte: Autoria própria.

Quadro 4- Caso de Uso 04.

Caso de Uso 04	Testar Vulnerabilidade DDoS
Descrição	O usuário deverá utilizar a ferramenta de mapeamento de Teste de Vulnerabilidade DDoS.
Atores envolvidos	Usuário do sistema.
Pré-condição	Usuário deve navegar pelos menus da interface até a ferramenta.
Cenário Principal de Sucesso	1. O usuário deve navegar até a opção de Segurança e Senhas.
	2. O usuário deve informar o dispositivo e o protocolo que será testado.
	3. A ferramenta executa o processo e retorna o status da verificação.
Cenário Secundário de Falha	1. A ferramenta retorna a mensagem de erro e não executa o processo.

Fonte: Autoria própria.

Quadro 5- Caso de Uso 05.

Caso de Uso 05	Quebrar senhas por Hash
Descrição	O usuário deverá utilizar a ferramenta de mapeamento de Quebra de Senhas por Hash.
Atores envolvidos	Usuário do sistema.
Pré-condição	Usuário deve criar um arquivo de *hashs* de senhas ou um dicionário de senhas.
Cenário Principal de Sucesso	1. O usuário deve navegar até a opção de Segurança e Senhas.
	2. O usuário deve selecionar o arquivo para análise das senhas.
	3. O sistema realiza a análise da senha e retorna o status da análise.
Cenário Secundário de Falha	1. A ferramenta retorna a mensagem de erro e não retorna as informações.

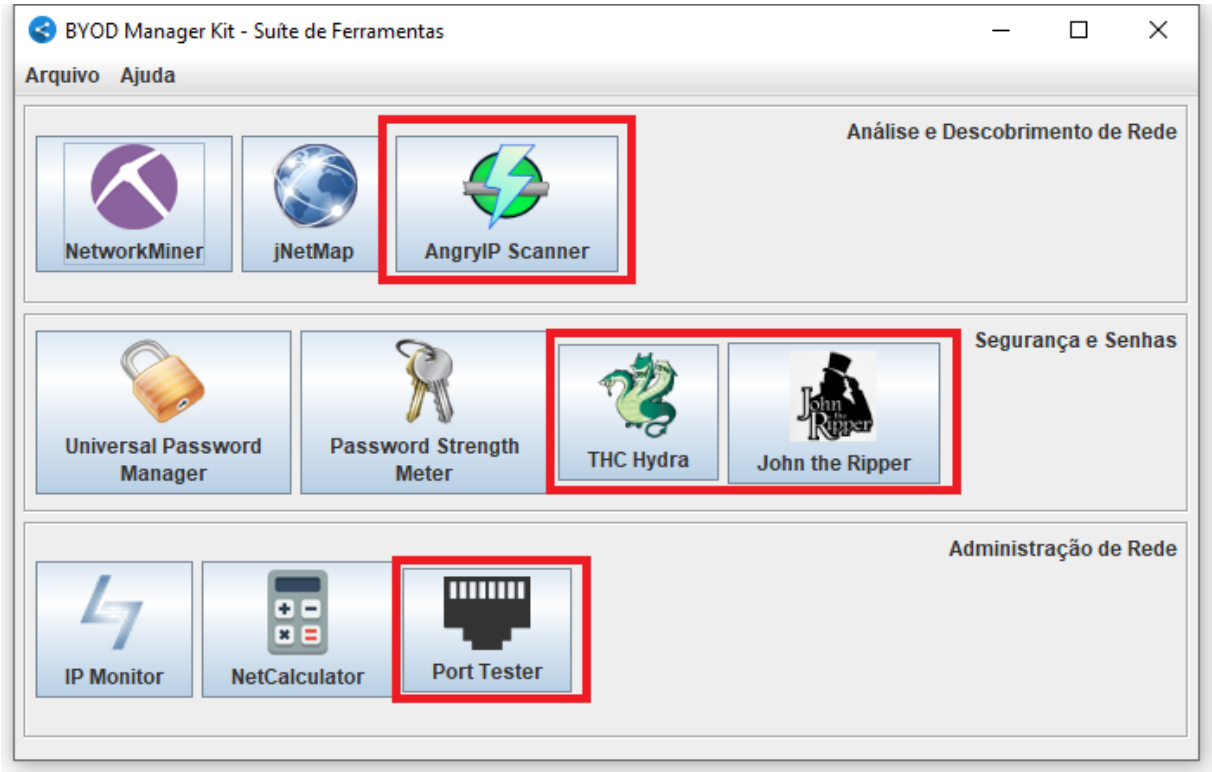
Fonte: Autoria própria.

4.2.3 INTERFACES GRÁFICAS DE USUÁRIO

Para ilustrar o planejamento de interfaces gráficas de usuário são apresentadas algumas telas. As telas apresentadas não representam a versão final das interfaces, apenas a ideia da integração e funcionamento das ferramentas.

A Figura 26 ilustra um modelo de interface geral. Através desta será possível acessar as ferramentas do *software*.

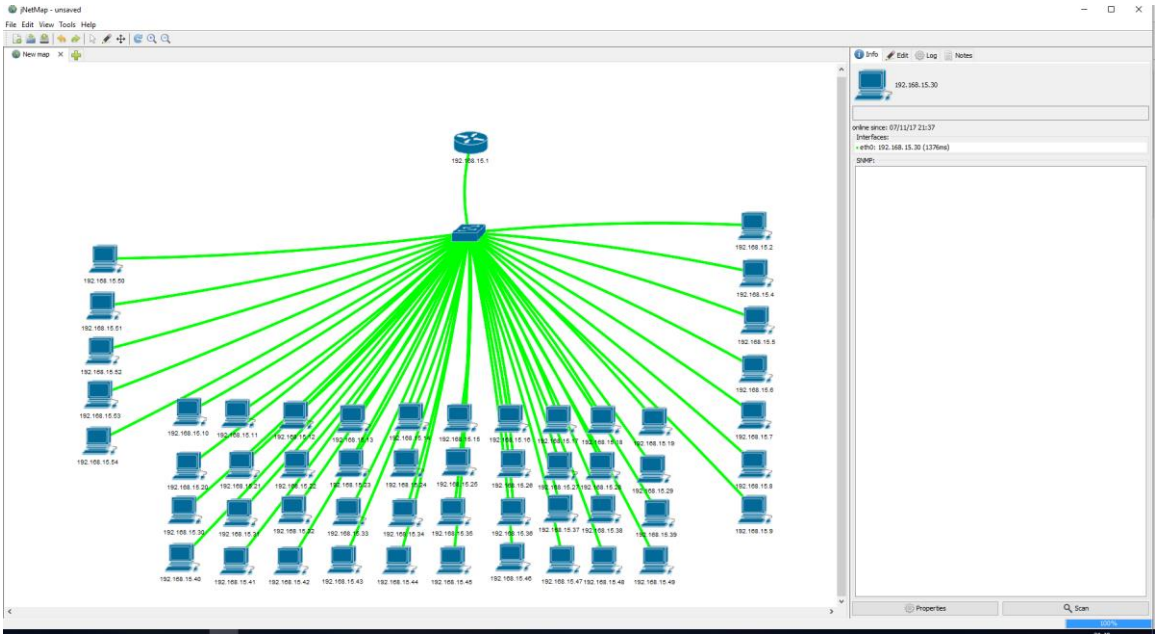
Figura 26 - Interface Gráfica Principal do software.



Fonte: Autoria própria.

A interface original do *software* foi alterada para contemplar novas funcionalidades que podem ser acessadas do menu principal. Algumas funcionalidades, como por exemplo, identificar portas vulneráveis à ataques DDoS e conectar ao dispositivo IoT, podem ser acessadas do *software* jNetMap através de atalhos específicos. No menu principal, estão classificadas dentro das categorias pré-existentes (Análise e Descobrimento de Rede, Segurança e Senhas e Administração de Rede). Os *softwares* Angry IP Scanner e John the Ripper possuem acesso apenas pela tela principal. O *software* Port Tester possui acesso pela tela principal e por um atalho criado ao clicar com o botão direito em um dispositivo mapeado no jNetMap (Figura 27).

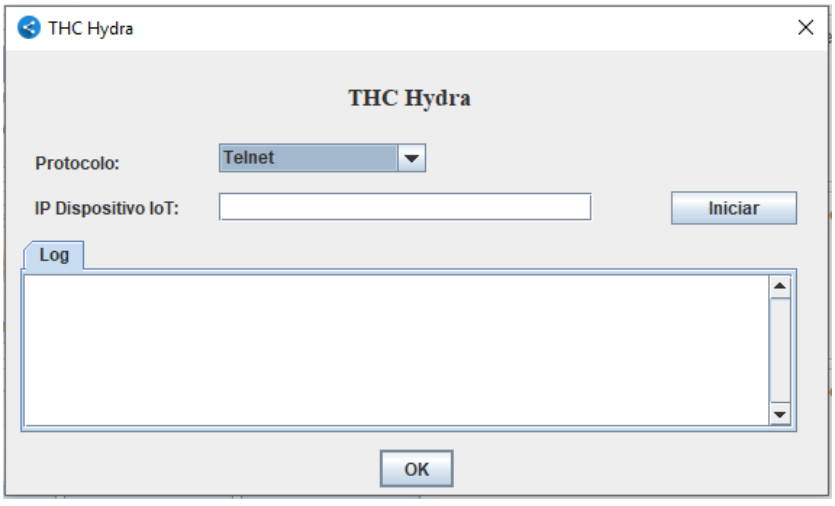
Figura 27 - Interface Gráfica Principal do JnetMap.



Fonte: Autoria própria.

O *software* THC Hydra possui uma tela auxiliar (Figura 28) para informar a configuração utilizada para o teste de vulnerabilidade contra DDoS. Esta tela é chamada a partir da tela principal e de um atalho criado no jNetMap. Ao ser iniciada através do jNetMap a informação de endereço do dispositivo é carregada automaticamente. Os diferentes protocolos que podem ser validados são exibidos para seleção do usuário. Por padrão a opção selecionada é Telnet, principalmente utilizada pelo *botnet* Mirai.

Figura 28 - Interface Gráfica Teste Vulnerabilidade DDoS.



Fonte: Autoria própria.

As funcionalidades adicionadas ao *software* jNetMap serão previamente carregadas com informações dos dispositivos selecionados na Topologia de Rede previamente definida.

4.3 CONSIDERAÇÕES FINAIS

O uso do *software BYOD Manager Toolkit* como base para a integração é positivo pois a funcionalidade de mapeamento da topologia de rede (*jNetMap*) já está integrado e funcional evitando que seja realizado um retrabalho sem necessidade.

As funcionalidades agregadas são úteis não somente para administradores de redes BYOD, mas administradores de redes sem fio em geral. Novos dispositivos, pessoais ou não, podem ser conectados na rede comprometendo sua segurança.

O funcionamento do *software* no ambiente Windows é garantido através da utilização conjunta do IDE Eclipse, do *framework* JNA e das demais definições de tecnologia e seus respectivos requisitos.

5. DESENVOLVIMENTO DO SOFTWARE

A integração das ferramentas *open-source* Port Tester 1.0, THC Hydra 8.6, John de Ripper e Angry IP Scanner, citadas na seção 4.2, foi realizada com o objetivo de oferecer praticidade ao profissional de TI ou usuário doméstico. Com a integração, é disponibilizado um conjunto de ferramentas que possuem objetivos diferentes em apenas um *software*, em que algumas das ferramentas possuem comunicação direta entre elas, evitando possíveis problemas de compatibilidade que o usuário poderia sofrer ao usar ferramentas desconexas.

A interface gráfica do software foi readequada conforme as novas funcionalidades. O processo de desenvolvimento e todas as alterações do projeto estão relatados nas próximas seções.

Para o pleno desenvolvimento do projeto os seguintes requisitos foram instalados e configurados:

- Windows 10;
- JDK (*Java SE Development Kit*) 9;
- Linguagem de Programação: Java;
- WindowBuilder 1.9 *plug-in;*
- Eclipse Oxygen IDE;
- Subsistema Operacional Ubuntu no Windows 10.

Por acreditar que o software desenvolvido por Perini (2017) não se restringe apenas a proteção de dispositivos BYOD, mas a uma gama mais ampla de equipamentos, a partir desse momento, utilizaremos o nome IoT Manager and Security Toolkit para referenciar o software e seus novos incrementos. IoT (Internet of Things) refere-se à conexão de diferentes dispositivos, como por exemplo, carros, *laptops*, *smartphones* até microondas, geladeiras, semáforos e dispositivos cardíacos. (MEOLA, 2016).

5.1 ESTRUTURA DE FUNCIONAMENTO

O software de integração, nomeado de *IoT Manager and Security Toolkit*, visa à criação de uma suíte de ferramentas para a administração e segurança de redes. Através desta suíte é possível acessar as ferramentas *open-source* identificadas na seção 4.2 através dos ícones ou via menu “Arquivo”. A Figura 29 ilustra a interface principal da suíte. A interface foi ajustada

e adaptada para oferecer o correto funcionamento em diversas resoluções de tela utilizando o *plug-in* WindowBuilder do Eclipse IDE.

Figura 29 - Interface Principal do software

Fonte: Autoria própria.

As ferramentas acrescentadas ao software original foram classificadas dentro das três áreas de atuação já existentes: a área de "Análise e Descobrimento de Rede", a área de "Segurança e Senhas" e a área de "Administração de Rede".

Na área de "Análise e Descobrimento de Rede" foi adicionada a ferramenta Angry IP Scanner que permite um escaneamento rápido da rede conectada, garantindo um descobrimento dos dispositivos conectados através do uso de diversas *threads*. Além desta funcionalidade principal é possível executar funções secundárias para cada dispositivo listado no *software*, entre outras funções é possível executar comandos de *ping*, *trace route* e abrir o endereço no navegador.

Na área de "Segurança e Senhas" foram adicionadas as ferramentas THC Hydra e John the Ripper. As ferramentas foram classificadas neste grupo pois a ferramenta THC Hydra permite que seja feito uma simulação de um ataque utilizando um conjunto de senhas padrões, utilizadas pelos fornecedores de hardware. Este mesmo conjunto de senhas é o utilizado pelo *botnet* Mirai responsável pela infecção de milhares de dispositivos IoT. A ferramenta John the

Ripper é capaz de quebrar a criptografia de senhas que utilizam o tipo de salto DES e MD5, mesma criptografia utilizada em sistemas Unix.

Na área de "Administração de Rede" foi adicionada a ferramenta Port Tester 1.0. A ferramenta foi classificada neste grupo pois permite que o usuário verifique quais portas de seus dispositivos estão abertas, vulneráveis à ataques DDoS e de outros tipos.

Além da interface principal, a estrutura de comunicação através de um processo servidor (*Server Socket* [7]) na porta 7000 do *localhost,* foi mantida e adaptada para funcionar com as novas ferramentas incorporadas ao *software.* Este processo servidor é responsável por realizar a comunicação entre diferentes ferramentas que podem ser utilizadas ao mesmo tempo e possuem funções que uma complementa a outra. Todo o detalhamento de implementação destas funcionalidades também é descrito nas próximas seções.

A arquitetura e modelagem de software utilizados nas ferramentas *open-source* seguiram a mesma estrutura desenvolvida pelos autores das ferramentas originais (jNetMap, Angry IP Scanner,...). Na ferramenta Port Tester 1.0 que não segue o modelo MVC foi criada uma classe Controle para agrupar as funcionalidades da integração.

O Apêndice A é composto pelo manual de usuário da ferramenta *IoT Manager and Security Toolkit*. Neste manual é possível acompanhar passo a passo todas as funcionalidades do *software*.

5.1.1 FLUXO DE COMUNICAÇÃO ENTRE PROCESSOS

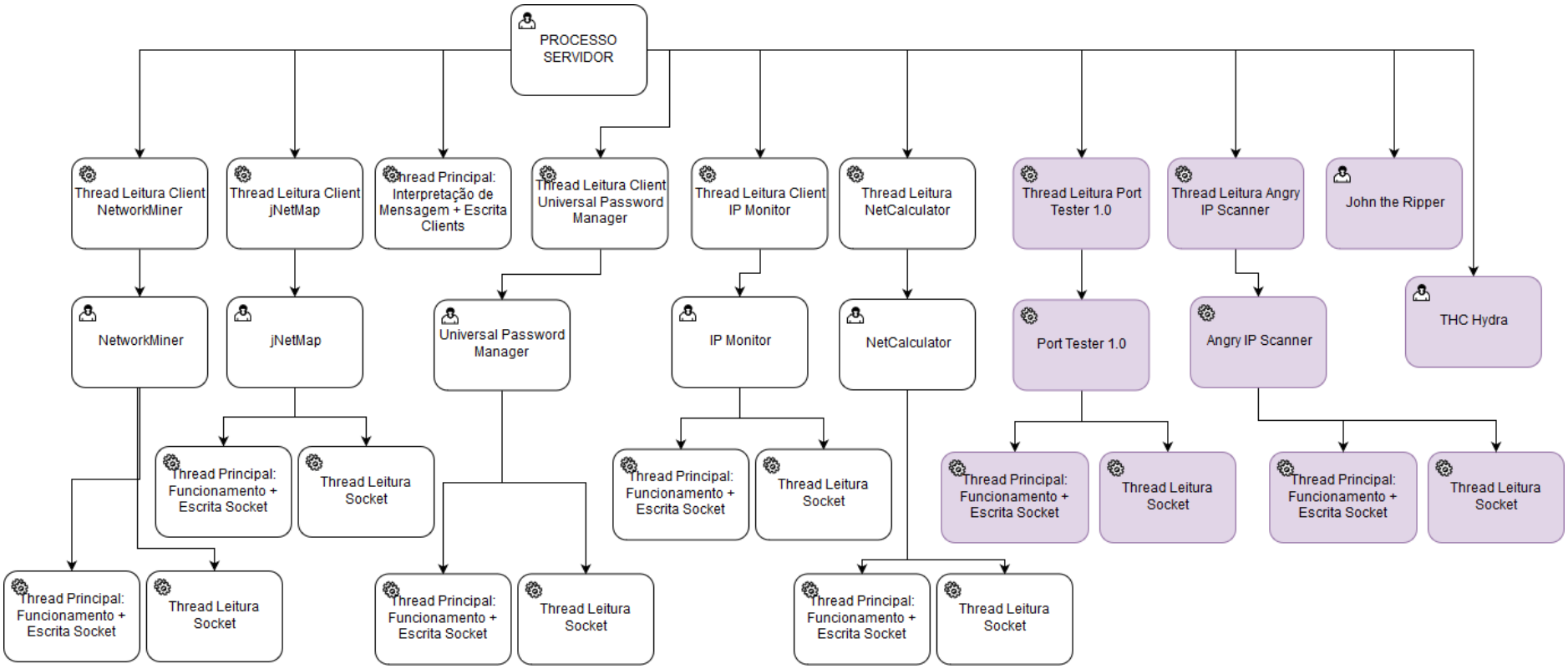
A administração do sistema de comunicação entre as ferramentas *open-source* é realizada através de um fluxo. Ao inicializar a interface principal é iniciado um processo servidor, enquanto os processos cliente são iniciados sempre que uma ferramenta *open-source* é aberta, procedimento padrão do *software* original.

O fluxo de comunicação (Figura 30) consiste na utilização de *threads* e *sockets*. No momento em que uma ferramenta é inicializada, o servidor cria uma *thread* somente para a leitura do socket. O processo cliente segue a mesma lógica, criando uma *thread* somente para leitura. As escritas no *socket* são feitas pela *thread* principal das ferramentas. Esse modelo foi adotado por ser difícil prever quando e/ou em qual ordem o usuário enviará informações de uma ferramenta para outra (PERINI, 2017).

[7] "ServerSocket" é o nome da classe Java utilizada para a criação do processo servidor. A documentação oficial em: <https://docs.oracle.com/javase/7/docs/api/java/net/ServerSocket.html>.

As ferramentas *THC Hydra e John the Ripper* não utilizam a comunicação por *sockets*, pois são ferramentas utilizadas no subsistema Linux, para elas é retornado em tela o resultado de comandos executados de forma invisível ao usuário. As funcionalidades e ferramentas agregadas estão em azul (balões escuros).

Figura 30 - Fluxograma de Comunicação Integração



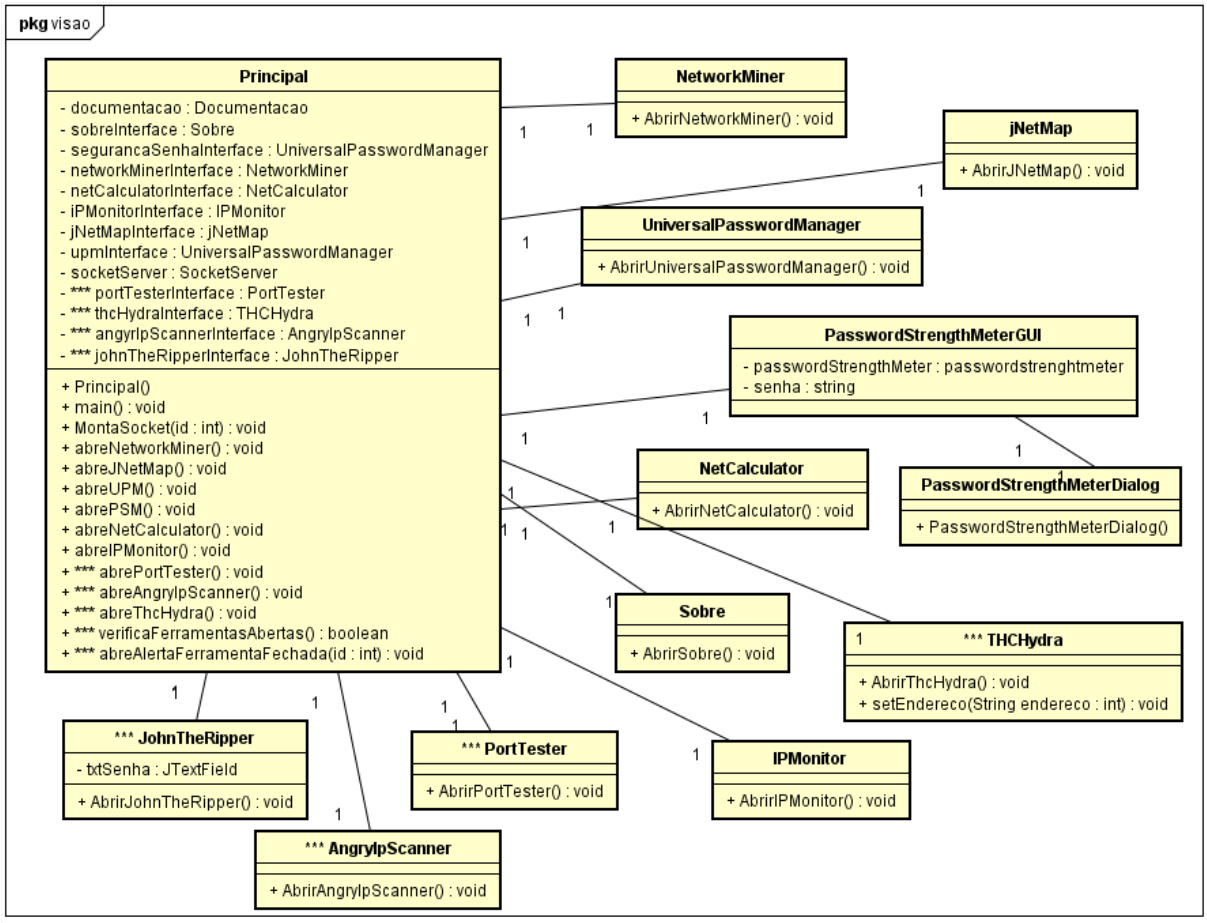
Fonte: Autoria própria.

5.1.2 DIAGRAMA DE CLASSES

A Figura 31 representa o diagrama de classes da camada visão. As classes originais do *software* foram mantidas e as novas classes e funções estão identificadas com três asteriscos (***). Os novos objetivos específicos das classes adicionadas com a integração foram:

- Principal: essa é a classe que contém a função *main*. Seu funcionamento consiste em exibir a interface principal de usuário e interpretar os eventos de cliques do usuário. Ela também é responsável por trocar dados com a camada controle, comunicando a abertura de uma nova ferramenta e teve novas funções agregadas.
- THCHydra e JohnTheRipper: essas classes são responsável por exibir uma interface gráfica que foi desenvolvida para que o usuário não precise digitar os comandos necessários de uso da ferramenta no Linux e escrever em tela o retorno da camada controle.
- AngryIpScanner: esta classe é responsável por abrir a ferramenta Angry IP Scanner.
- PortTester: esta classe é responsável por abrir a ferramenta Port Tester.

Figura 31 - Diagrama de Classes da Camada Visão



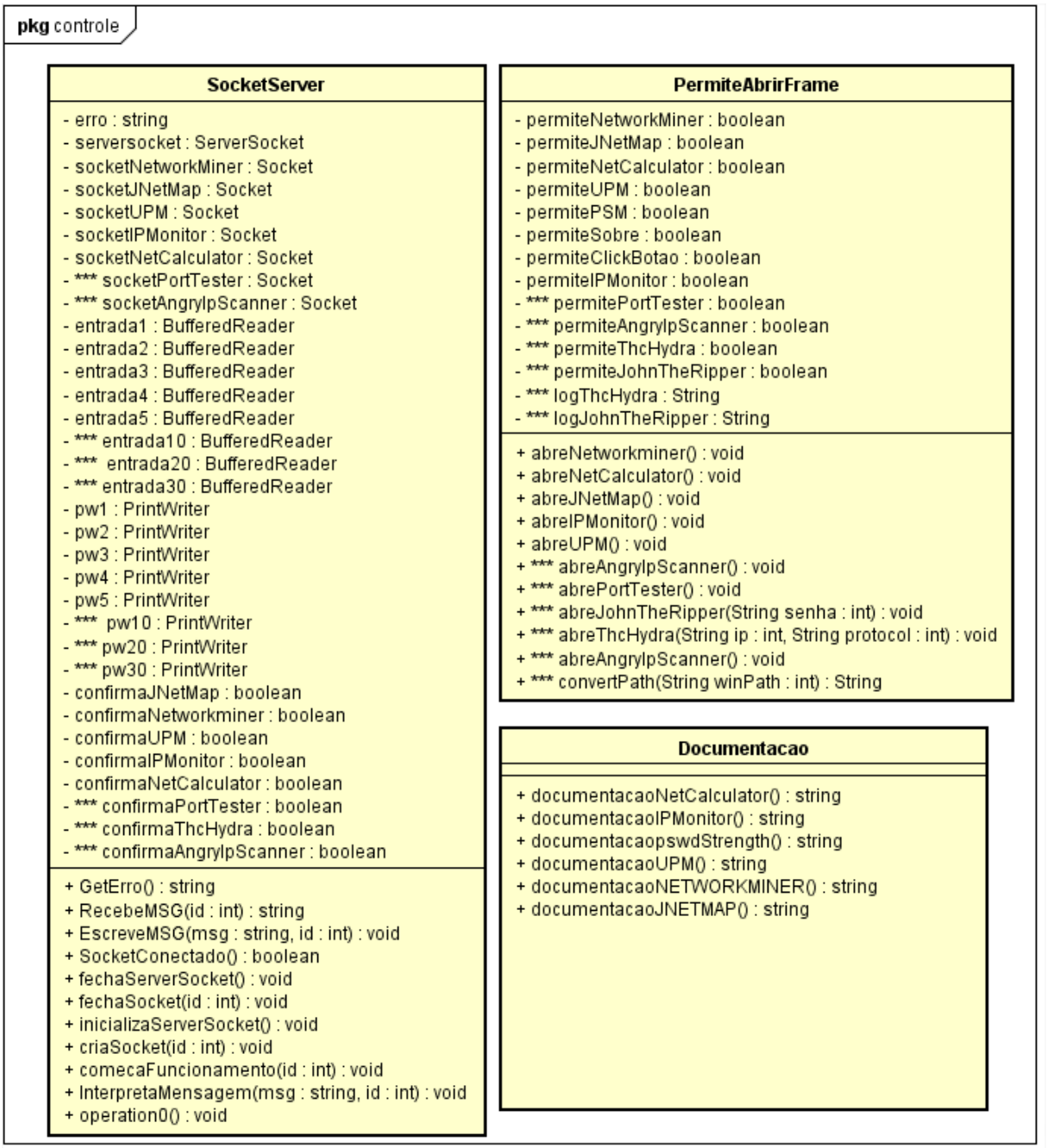
Fonte: Autoria própria.

A Figura 32 representa o diagrama de classes da camada controle. Os objetivos específicos das classes modificadas são:

- SocketServer: essa é a classe mais importante em todo o processo de integração das ferramentas. Essa classe é responsável por criar o *server socket*, estabelecer a conexão dos *clients sockets*, e por enviar mensagens e interpretar os dados recebidos. Desta forma é concretizada a comunicação entre os processos.

- PermiteAbrirFrame: essa classe é responsável pelos *flags* que garantem que uma ferramenta não seja inicializada mais de uma vez simultaneamente. Além disso, essa classe possui as funções que realizam as chamadas de inicialização das ferramentas.

Figura 32 - Diagrama de Classes da Camada Controle



Fonte: Autoria própria.

As Figuras 33 e 34 representam o diagrama de classes da camada modelo. Os objetivos específicos das classes mais importantes são:

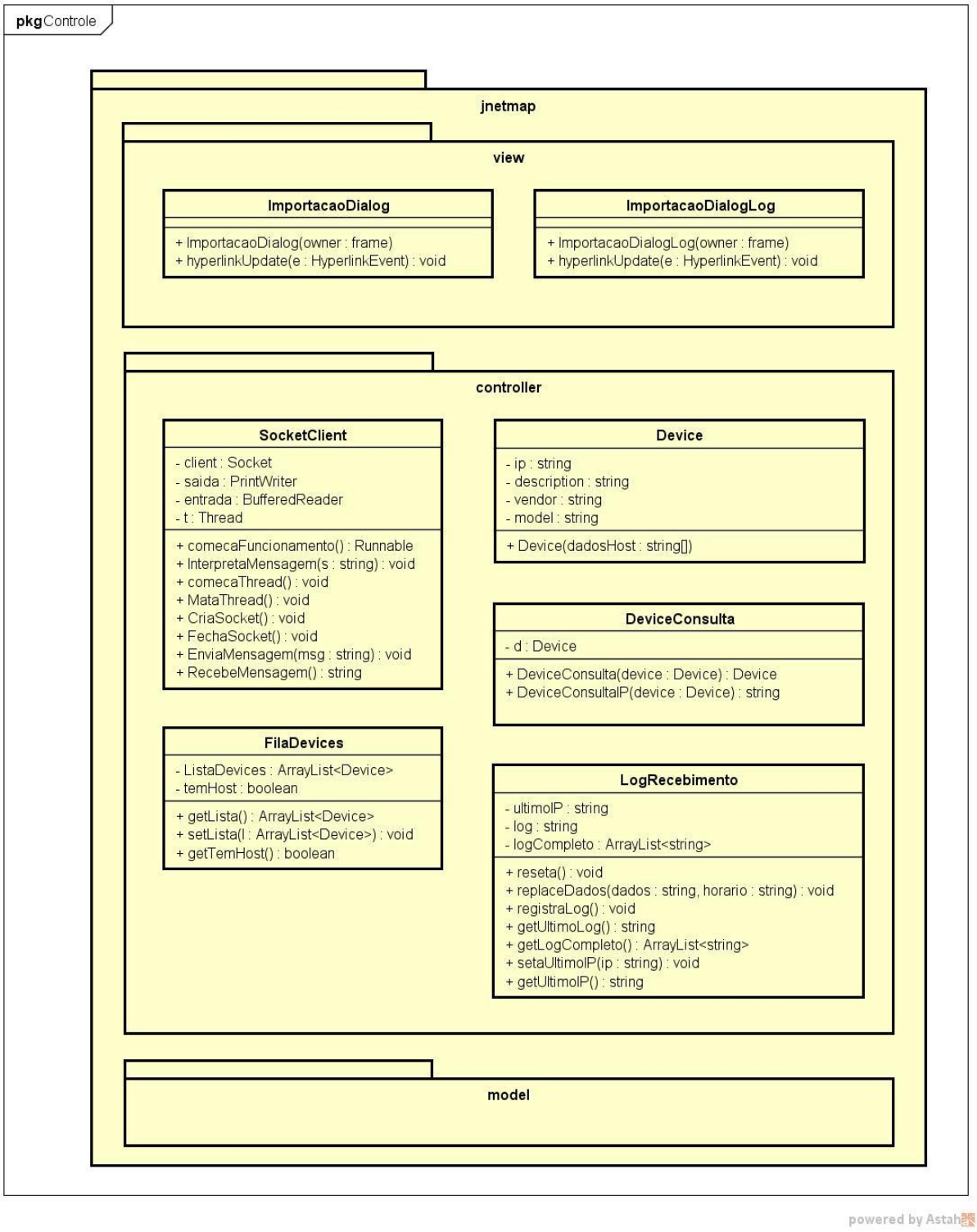
- SocketClient (Port Tester): essa classe é responsável por se conectar com o servidor criado na camada controle, por enviar e receber informações e por interpretar os dados recebidos.

- SocketClient (jNetMap): essa classe é responsável por se conectar com o servidor criado na camada controle, por enviar e receber informações e por interpretar os dados recebidos. Esta classe não precisou ser alterada.

- DeviceConsulta (jNetMap): essa classe é responsável por realizar a consulta dos dados de um dispositivo selecionado na árvore de rede. Esta classe não precisou ser alterada.

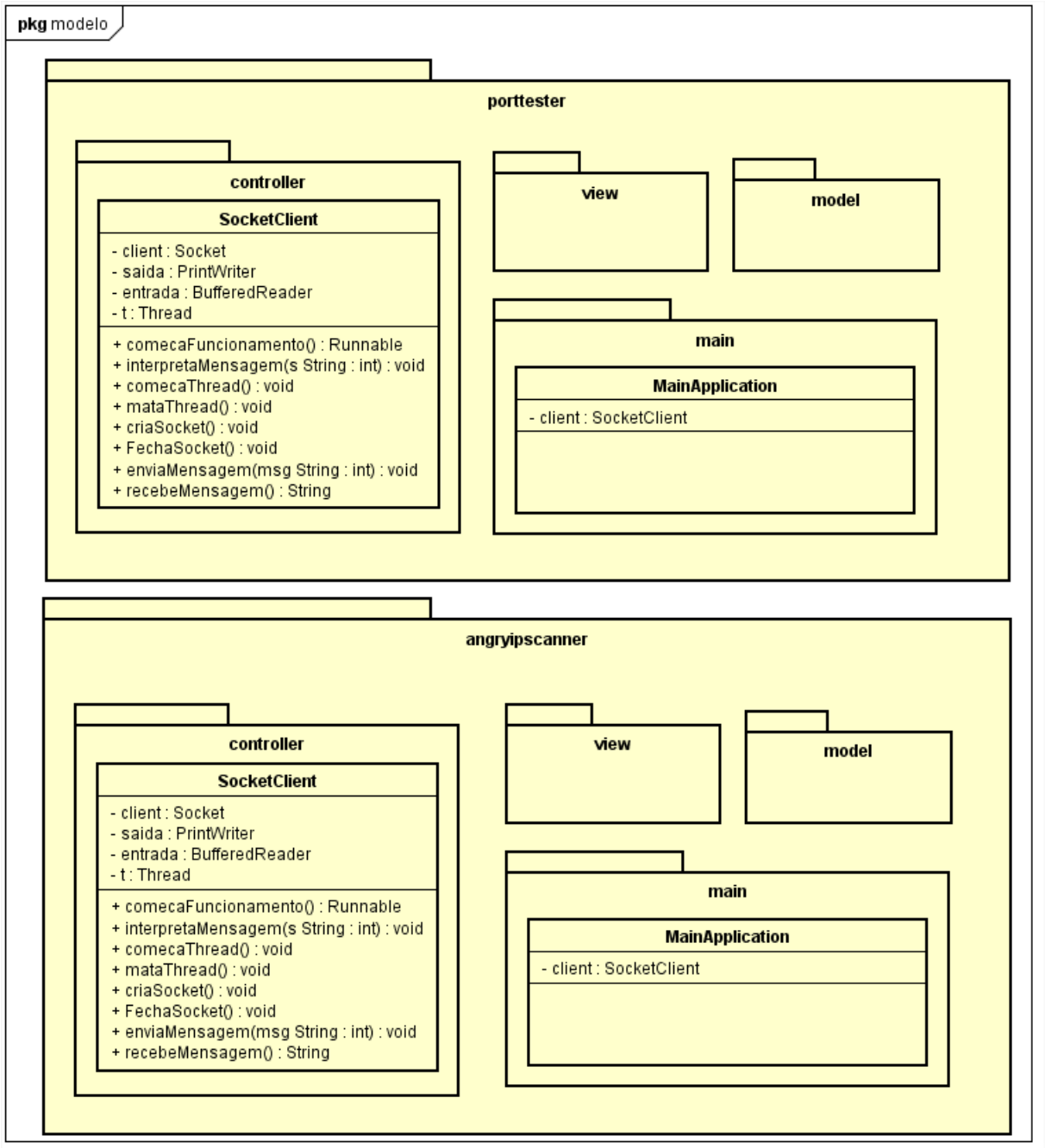
- SocketClient (Angry IP Scanner): essa classe é responsável por se conectar com o servidor criado na camada controle, por enviar e receber informações e por interpretar os dados recebidos.

Figura 33 - Diagrama de Classes do jNetMap



Fonte: Perini, 2017.

Figura 34 - Diagrama de Classes do Port Tester 1.0 e Angry IP Scanner



Fonte: Autoria própria.

5.1.3 CAMADA MODELO

Na camada modelo do *software* ficam os arquivos fonte das ferramentas *open-source* que foram selecionadas. As ferramentas são desenvolvidas em Java e após alterações realizadas

foram salvas em formato JAR[8] (Java Archive – Arquivo Java) e adicionadas a esta camada.

O código fonte da camada modelo não foi alterado nas ferramentas *open-source* mantendo a arquitetura criada pelos seus desenvolvedores oficiais.

5.1.4 CAMADA VISÃO

Na camada visão a interface principal do *software* foi alterada e novas interfaces foram adicionadas. Todas essas interfaces foram desenvolvidas utilizando o *plug-in WindowBuilder*.

A Figura 35 mostra a interface gráfica que o *WindowBuilder* proporciona no Eclipse IDE e consequentemente sua facilidade e agilidade no desenvolvimento deste tipo de objeto. Através de uma visualização gráfica, o WindowBuilder gera o código fonte de acordo com os objetos e suas propriedades, que são adicionados na tela (PERINI, 2017).

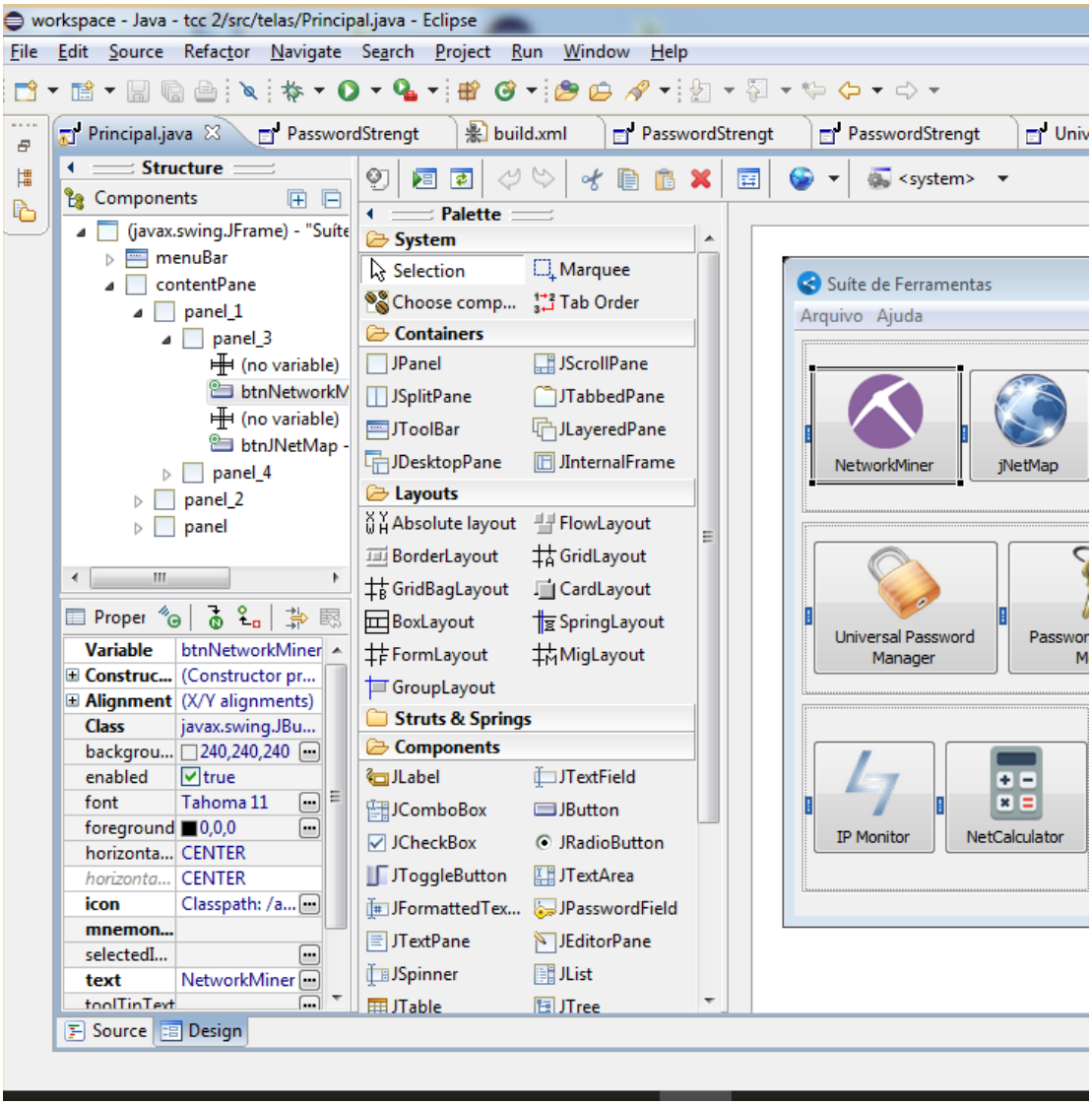
A Classe Principal.java é a classe responsável pela função *main* e por criar a janela principal da interface gráfica. Na função *main* é criada uma instância da classe *SocketServer* (Camada Controle) para possibilitar a integração de todas as ferramentas. Esta classe é responsável pelo gerenciamento dos eventos de usuário e por mostrar possíveis erros.

A classe THCHydra.java é a classe responsável por um diálogo desenvolvida especificamente para o uso da ferramenta THC Hydra que é executada pelo subsistema Linux. Ela possibilita que o usuário escolha um protocolo, dentre as opções Telnet, HTTP-GET, SMTP, SSH e FTP, e informar o IP de um dispositivo IoT para realizar um teste de vulnerabilidade através de um ataque similar ao realizado pelo *botnet* Mirai. Após isso exibe em uma caixa de texto o log da simulação identificando se o dispositivo possui uma combinação de usuário e senha vulnerável.

A classe JohnTheRipper.java é a classe responsável por um diálogo desenvolvida especificamente para o uso da ferramenta John The Ripper. Ela possibilita que o usuário quebre o *hash* de uma senha criptografada exibindo o retorno da operação em uma caixa de texto com a senha descriptografada ou um possível erro.

[8] O formato JAR é um formato utilizado para agrupar vários arquivos de classes (formato .java) e recursos associados (imagens, ícones, pdfs, etc) em um só arquivo. Ele pode ser utilizado como um arquivo executável para distribuição de programas (semelhante ao formato .exe do Windows). Mais informações em: <https://docs.oracle.com/javase/tutorial/deployment/jar/index.html>.

Figura 35 - Interface Gráfica do WindowBuilder; Editando a classe Principal.java.



Fonte: Autoria própria.

5.1.5 CAMADA CONTROLE

A camada controle é originalmente estruturada por três classes: Documentacao.java, PermiteAbrirFrame.java e SocketServer.java. A camada controle é responsável pelo gerenciamento da quantidade de ferramentas que estão abertas, pela chamada de inicialização das ferramentas, além do gerenciamento de comunicação entre processos clientes e o processo servidor. Além, da documentação original das ferramentas que também é tratada nesta camada.

A classe PermiteAbrirFrame.java é a classe que controla se uma ferramenta está ou não aberta, impedindo o usuário de abrir mais de uma vez a mesma ferramenta e também é responsável pela chamada para a inicialização das ferramentas (armazenadas na camada modelo). Esta classe sofreu mudanças para garantir que as novas ferramentas funcionassem seguindo a regra original.

A classe SocketServer.java é responsável por criar o servidor *socket* que permite a conexão dos clientes. Cada cliente é conectado respeitando um sistema de identificação "IDs" previamente definido no código. Por uma fração de segundos é realizado um bloqueio sempre que uma nova ferramenta é inicializada, garantindo o correto funcionamento da comunicação. Esta classe possui a função "InterpretaMensagem(String msg, int id)" que recebe a mensagem que deverá ser interpretada e o "id" que representa qual ferramenta enviou esta mensagem. A mensagem pode significar informações que devem ser enviadas para outra ferramenta ou até comunicar o fechamento de uma ferramenta e sofreu alterações para receber as novas funcionalidades.

5.2 ALTERAÇÕES NO CÓDIGO FONTE DAS FERRAMENTAS

A fim de executar a comunicação entre as ferramentas e o processo servidor, foi criada a classe "SocketClient" responsável por enviar e receber as informações com o processo servidor em formato String[9].

A classe "SocketClient" foi adicionada aos projetos *open-source* respeitando o modelo MVC. Nas ferramentas que utilizavam o modelo, ela foi adicionada à camada controle destes projetos. Para seguir o padrão de desenvolvimento, a camada controle foi criada nas ferramentas que não utilizavam o modelo MVC. Ferramentas que não utilizam o modelo foram

[9] String é uma estrutura de dados composta por uma cadeia de caracteres. Maiores informações sobre o tratamento de Strings em Java está disponível em: <https://docs.oracle.com/javase/7/docs/api/java/lang/String.html>.

desenvolvidas desta forma pelos autores originais pelo projeto não possuir uma maior complexidade de desenvolvimento.

Todas as demais modificações e novas funcionalidades criadas nas ferramentas Angry IP Scanner e jNetMap foram feitas nas suas respectivas camadas de *software*. Estas funcionalidades estão explícitas nas próximas seções.

A ferramenta Port Tester sofreu alterações com a adição do pacote Controle e a da classe "SocketClient" nele. Também foi alterado para já trazer alguns números de portas previamente carregados nos campos e controla o fechamento da ferramenta com alterações na classe PortTesterGUI.java.

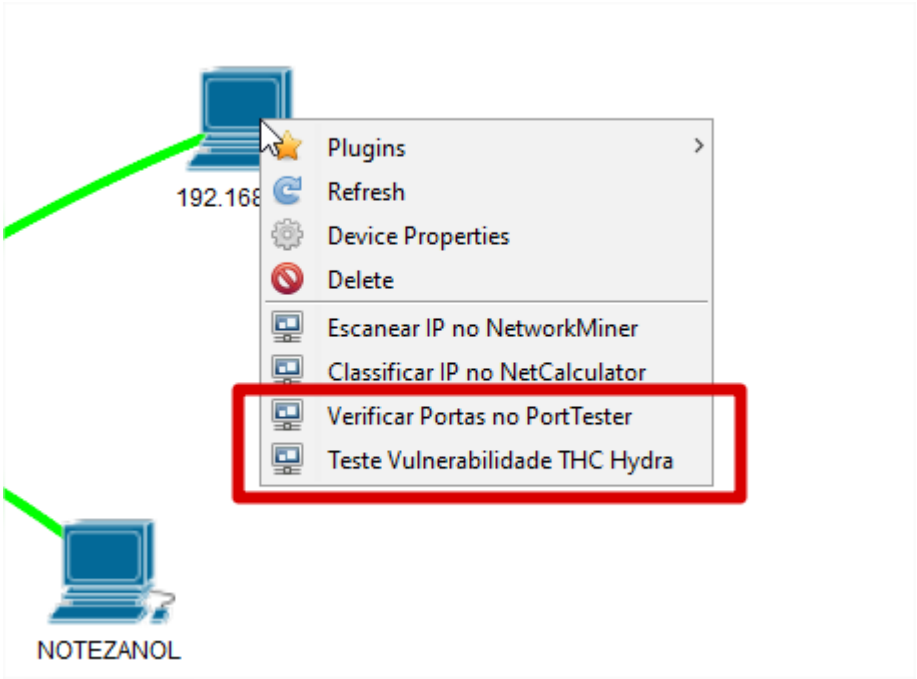
A ferramenta Angry IP Scanner sofreu alterações nas classes CommandsMenu.java e messages.properties para receber as novas opções gráficas para abrir as outras ferramentas da integração. A classe CommandsMenuActions.java recebeu as alterações para receber as alterações que enviam a mensagem para o processo servidor abrir as novas ferramentas. A classe Main.java sofreu alterações para criar o servidor de cliente de troca de mensagens e controlar o fechamento da tela. A classe CommandsMenu.java também foi alterada para controlar o fechamento da tela já que o comando de para fechar a tela possui comportamentos diferentes dependendo de onde foi clicado.

5.2.1 INTEGRAÇÃO NO JNETMAP

A integração entre o jNetMap e a ferramenta Port Tester foi desenvolvida com o intuito de possibilitar a troca de informações entre as duas ferramentas. Através dela é possível enviar dados do jNetMap para o Port Tester. Na ferramenta Port Tester, foi desenvolvido a funcionalidade para receber os dados e já carregar na tela o IP do dispositivo IoT selecionado.

A integração com o THC Hydra foi desenvolvida seguindo a mesma linha do Port Tester para já carregar o endereço de IP no campo em tela. A ferramenta sofreu alterações na classe PopupGraphMousePlugin.java para adicionar as novas funcionalidades de integração ilustradas na Figura 36.

Figura 36 - Detalhe das opções "Verificar Portas no Port Tester" e "Teste Vulnerabilidade THC Hydra" adicionada ao jNetMap.



Fonte: Autoria própria.

Ao clicar nas novas opções selecionadas uma mensagem é enviada para o processo servidor indicando qual das ferramentas deve ser executada e o endereço de IP que deve ser carregado automaticamente em tela. Ao receber um endereço IP, automaticamente a ferramenta Port Tester irá executar o teste para verificar se as portas pré-definidas na ferramenta Port Tester estão vulneráveis para o dispositivo IoT.

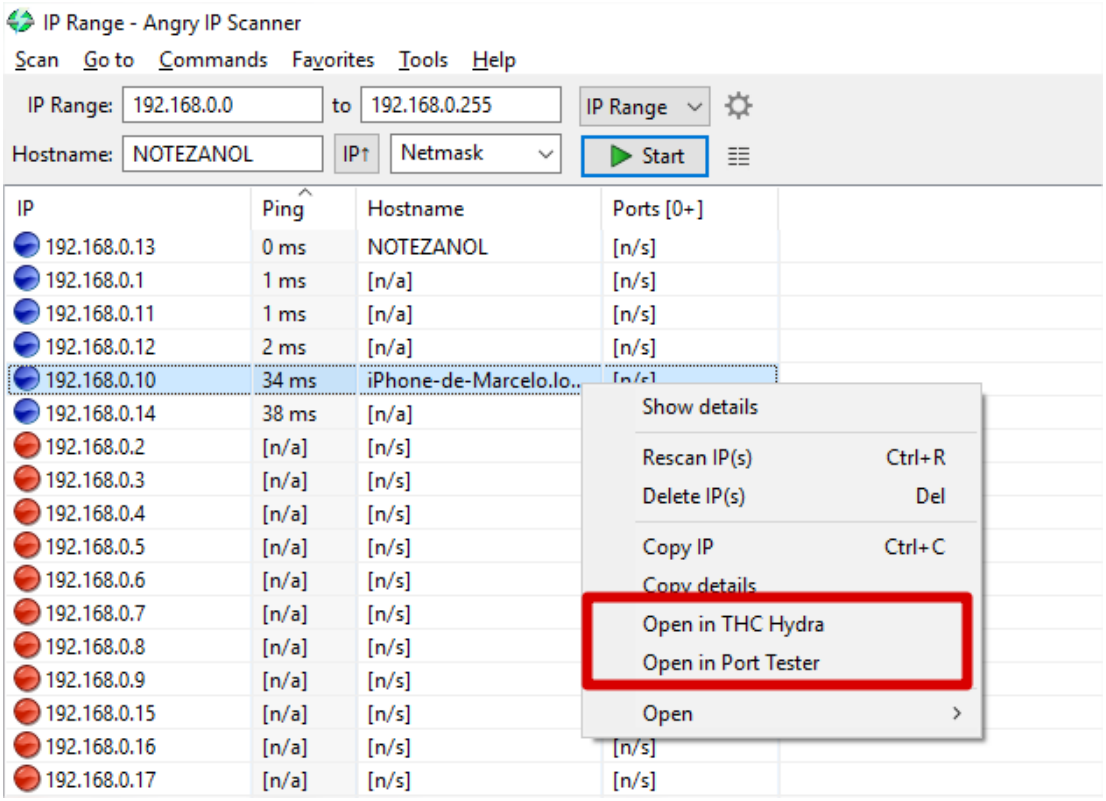
5.2.2 INTEGRAÇÃO NO ANGRY IP SCANNER

A integração entre o Angry IP Scanner e a ferramenta Port Tester e THC Hydra foi desenvolvida com o intuito de possibilitar a troca de informações entre as duas ferramentas, similar ao comportamento executado com a ferramenta jNetMap.

A ferramenta sofreu alterações na classe CommandsMenu.java para adicionar as novas funcionalidades de integração ilustradas nas Figuras 37 e 38.

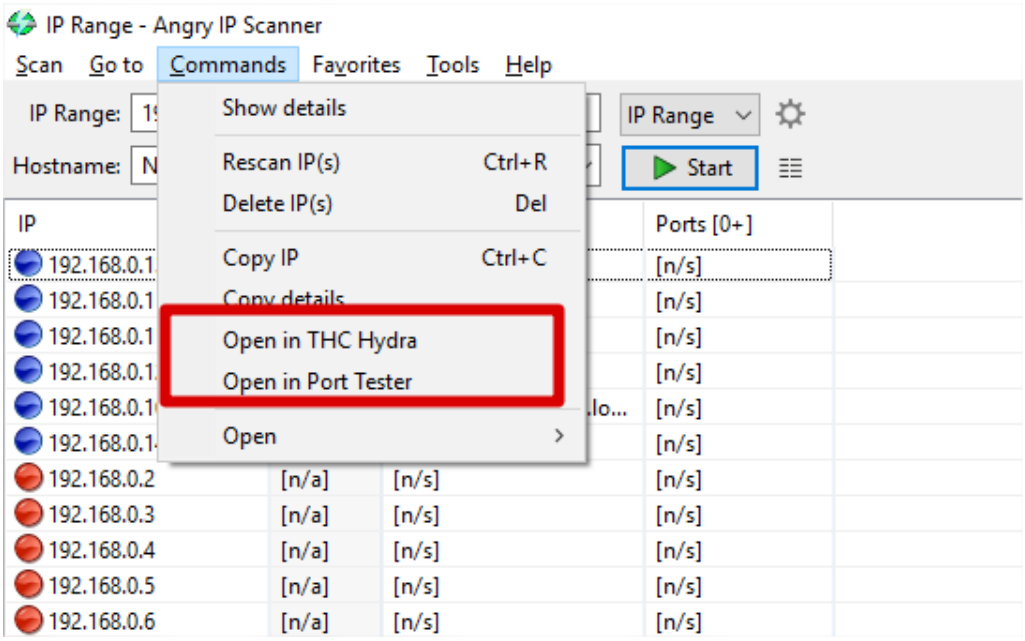
As dificuldades de compilação da ferramenta original tornaram esta integração um ponto crítico do trabalho por quase inviabilizar o seu uso. Sem sucesso, foi realizada uma pesquisa por outra ferramenta com mesmas funcionalidades para substituí-la.

Figura 37 - Detalhe das opções "Open in THC Hydra" e "Open in Port Tester" adicionada ao Angry IP Scanner no click direito do dispositivo.



Fonte: Autoria própria.

Figura 38 - Detalhe das opções "Open in THC Hydra" e "Open in Port Tester" adicionada ao Angry IP Scanner no menu principal.



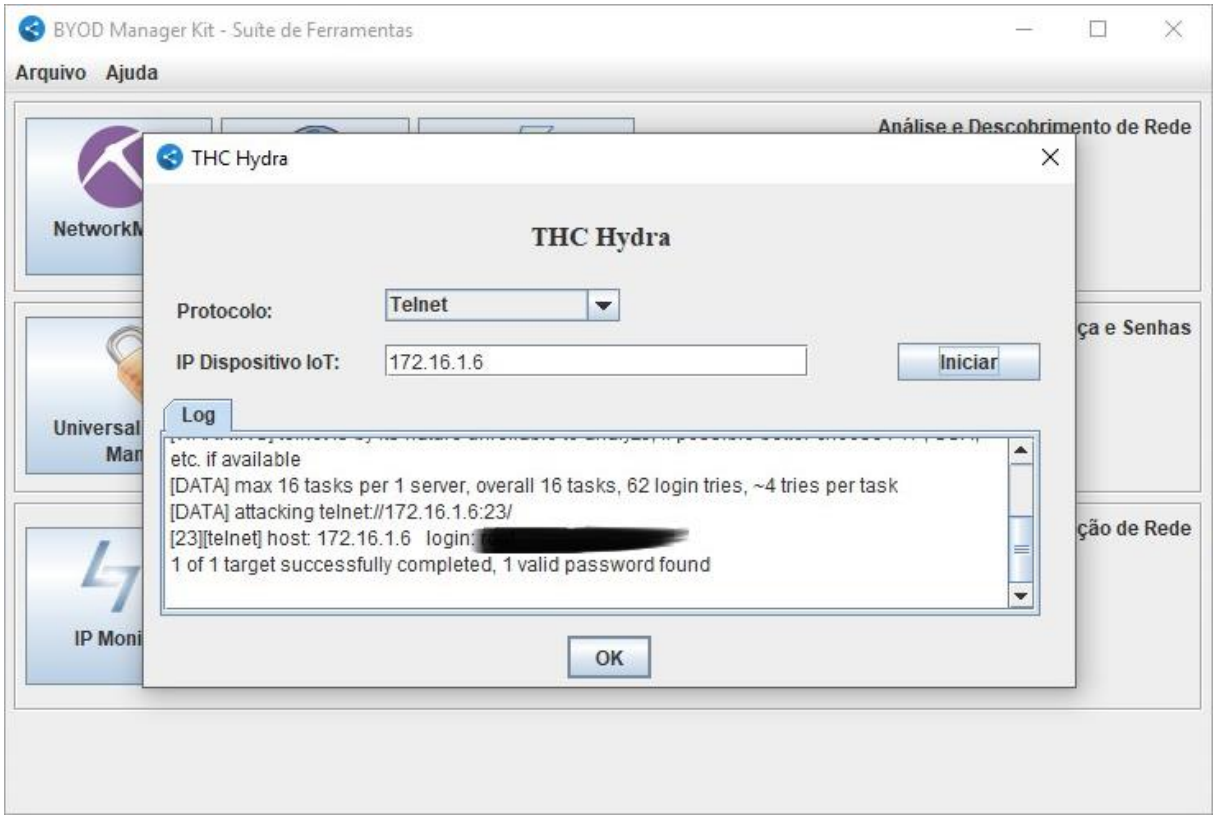
Fonte: Autoria própria.

Ao clicar nas novas opções selecionadas uma mensagem é enviada para o processo servidor indicando qual das ferramentas deve ser executada e o endereço de IP que deve ser carregado automaticamente em tela.

5.2.3 INTEGRAÇÃO THC HYDRA

A integração entre a tela principal e a ferramenta é realizada através de alterações na classe PermiteAbrirFrame.java que é responsável iniciar uma instância do Linux e executar a verificação de uma lista de logins padrões para o dispositivo informado em tela. Todo processo é executado de forma invisível ao usuário passando a impressão que o processo é todo executado pela tela principal. Ao final do processo um log e apresentado com o resultado dos testes conforme ilustrado na Figura 39.

Figura 39 – Simulação integração THC Hydra.



Fonte: Autoria própria.

5.2.4 INTEGRAÇÃO JOHN THE RIPPER

A integração entre a tela principal e a ferramenta foi realizada através de alterações na classe PermiteAbrirFrame.java que é responsável por criar um arquivo texto com a senha criptografada em tela e após isso iniciar a aplicação John the Ripper. Após executar os procedimentos necessários o retorno da descriptografia é retornado em tela. Todo processo é executado de forma invisível ao usuário sem a necessidade de interação com o John the Ripper.

5.3 CONSIDERAÇÕES FINAIS

O uso do mecanismo de comunicação através de uso de *sockets* mostrou-se eficiente e simples, facilitando a integração das ferramentas, mantendo seu código original sem uma grande quantidade de alterações, mantendo a sua estrutura de desenvolvimento original.

As funcionalidades agregadas são úteis e estão integradas de maneira que facilitará muito a utilização por administradores de redes sem fio em geral. Além da rápida verificação para dispositivos novos ou específicos a ferramenta para recuperação de senhas é útil para situações críticas relacionadas à perda de senhas.

A estrutura de desenvolvimento utilizada facilita o entendimento e desenvolvimento de novas funcionalidades, adição de outras ferramentas ou melhorias nas funcionalidades existentes.

6. TESTES E RESULTADOS

Durante o desenvolvimento dos incrementos no *software*, alguns testes foram realizados de acordo com o progresso da integração entre as ferramentas. Quando o desenvolvimento dos incrementos foi finalizado, a ferramenta foi testada inúmeras vezes a fim de garantir o correto funcionamento das funcionalidades originais e dos incrementos. Após os testes realizados em uma rede local doméstica para validação, foi realizado testes em dois ambientes reais. O primeiro deles foi uma rede sem fio (Wi-Fi) do Laboratório de Informática da Universidade de Caxias do Sul, no horário vespertino, que é um horário que não tem muito movimento é possível verificar a existência de mais de 400 dispositivos sem fio conectados. O segundo teste foi realizado em uma rede sem fio (Wi-Fi) de um ambiente corporativo, uma fábrica de *software* localizada em Caxias do Sul, em horário comercial, possibilitando verificar a existência de 239 dispositivos sem fio conectados. A documentação e resultados destes testes estão descritos a seguir.

Os testes foram divididos em duas etapas. A etapa inicial testou as principais funcionalidades originais das ferramentas integradas e também as funcionalidades que foram desenvolvidas e adicionadas no *software* de integração. A segunda etapa simulou o uso do *software* em ambientes reais utilizando o conjunto de funcionalidades para identificar dispositivos conectados vulneráveis à ataques DDoS e *malware*.

6.1 TESTES DE FUNCIONALIDADES DA INTEGRAÇÃO

Para a validação das funcionalidades da integração foram elaborados os seguintes casos de testes: Integração jNetMap, Integração Angry IP Scanner, Integração Port Tester, Integração THC Hydra e Integração John the Ripper. A seguir está o detalhamento para cada um dos testes realizados.

O primeiro teste realizado foi realizar uma avaliação das funcionalidades da integração em dispositivos mapeados no *software* jNetMap seguindo os passos descritos no Quadro 6.

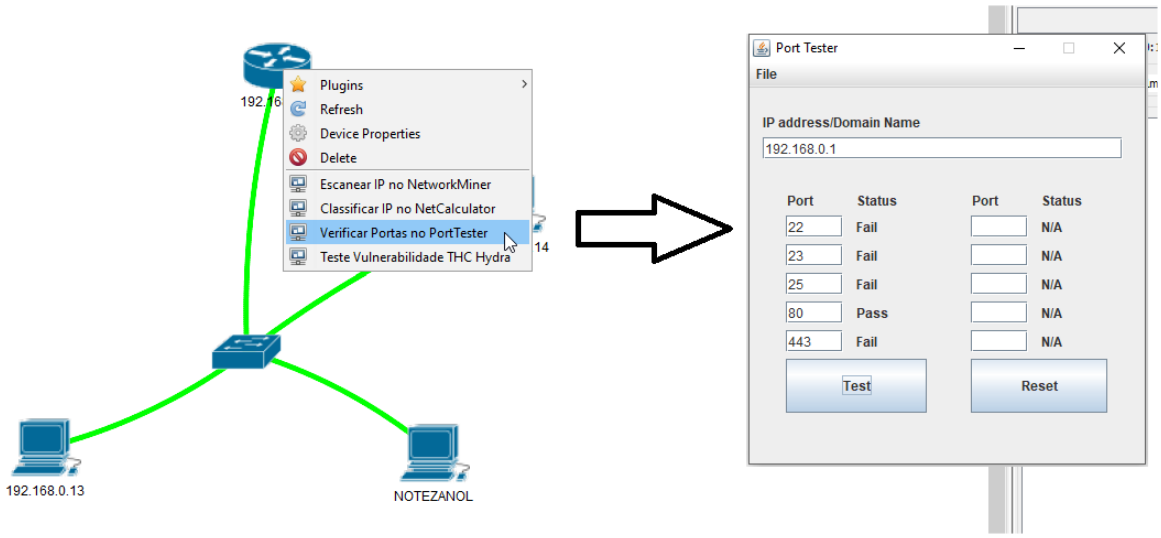
Quadro 6- Caso de Teste 01 – Integração jNetMap.

#	Ação	Resultado Esperado
1	Abrir ferramenta principal.	Apresentar tela principal da ferramenta.
2	Clicar no botão jNetMap.	Abrir a ferramenta jNetMap.
3	Carregar mapa de rede salvo.	Apresentar na tela o mapa de rede.
4	Clicar bom o botão direito em um dispositivo.	Apresentar menu de opções na tela.
5	Clicar na opção “Verificar portas no Port Tester”.	Abrir a ferramenta Port Tester com o endereço do dispositivo carregado e executar a verificação das portas pré-definidas exibindo o resultado ao lado do número das portas.
6	Clicar na opção “Teste Vulnerabilidade THC Hydra”.	Abrir a tela da ferramenta THC Hydra.
7	Clicar no botão “Iniciar” para executar o teste para o dispositivo selecionado.	Apresentar no campo “Log” o resultado do teste de vulnerabilidade.
8	Clicar na opção de fechar as ferramentas.	Fechar as ferramentas.

Fonte: Autoria própria.

Após abrir a ferramenta e clicar no botão principal jNetMap, a ferramenta foi iniciada e a partir do mapa de rede “redeApto.jnm” previamente criado para testes. A seguir foi utilizado as novas opções de menu desenvolvidas para abrir a ferramenta Port Tester, clicando na opção “Verificar Portas no Port Tester”. Automaticamente após abrir a ferramenta, foi realizado o teste de verificação de portas abertas para o dispositivo selecionado (IP 192.168.0.1) exibindo o resultado do teste de cada porta ao lado direito dela. Neste caso, apenas a porta 80 estava liberada apresentando o resultado “Pass”, enquanto as demais verificadas apresentavam o resultado “Fail” conforme ilustrado na figura 40. O mesmo procedimento foi relizado para os outros dispositivos da rede mapeada para validar que não seja aberta mais de uma vez a mesma ferramenta caso ela já esteja aberta. Clicando na opção “Verificar Portas no Port Tester” para o dispositivo 192.168.0.13 o endereço de IP dentro do Port Tester foi alterado para o solicitado sem abrir uma nova instância da ferramenta.

Figura 40 – Simulação integração jNetMap e Port Tester.



Fonte: Autoria própria.

Após verificado que a ferramenta Port Tester estava integrada com a ferramenta jNetMap e suas funcionalidades estavam conforme o planejado foi passado para a próxima etapa e validado a integração entre jNetMap e THC Hydra.

O teste foi realizado partindo da nova opção desenvolvida nomeada por "Teste Vulnerabilidade THC Hydra" para abrir a ferramenta THC Hydra previamente instalada no subsistema Linux. Ao clicar na opção, uma tela desenvolvida diretamente no *software* IoT Manager and Security Toolkit é exibida com o endereço de IP do dispositivo selecionado (192.168.0.13) preenchido no campo "IP Dispositivo IoT" e a opção "Telnet" no campo "Protocolo" para um teste de vulnerabilidade.

A lista de usuários e senhas criadas para a validação de vulnerabilidade foram extraídas do código fonte do Mirai disponibilizado pelo usuário Jerry Gamblin em seu repositório do Github[10]. Esta configuração vem pré-definida a partir de um arquivo de configuração criado para o THC Hydra.

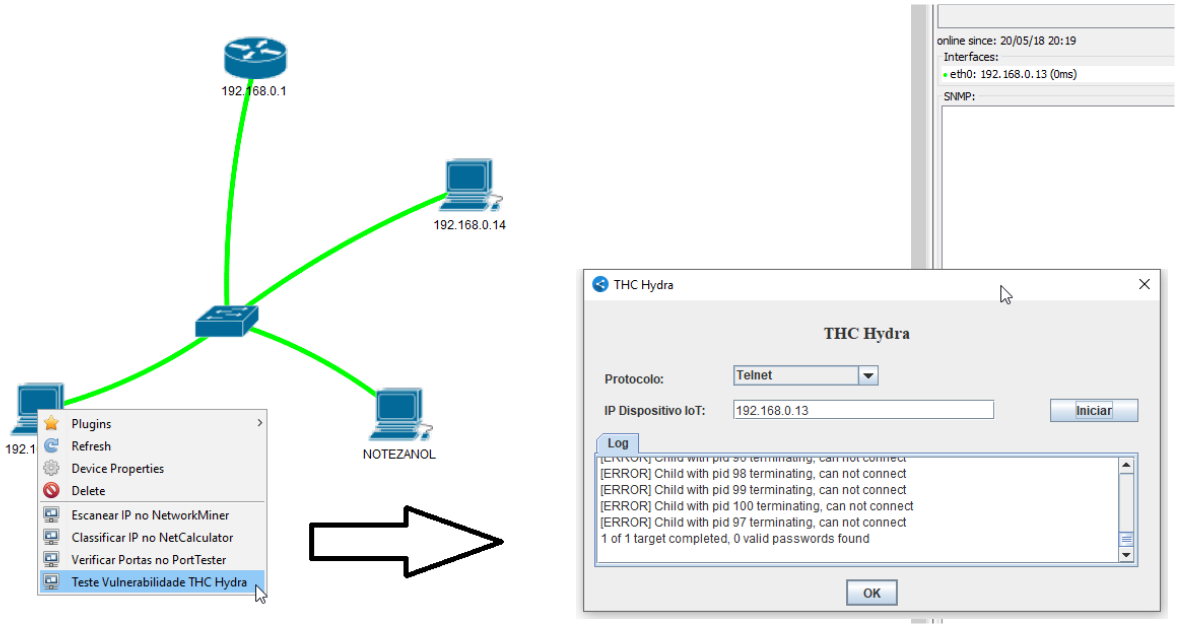
O resultado da verificação apontou que nenhum usuário e senha utilizado pelo Mirai *botnet* é válido para o protocolo Telnet (23), conforme ilustrado na figura 41. Outras portas suportadas pela ferramenta THC Hydra não foram validadas pois não estavam abertas. Neste teste também foi possível identificar que a velocidade de execução do teste de vulnerabilidade pode variar de uma rede para outra, em uma rede doméstica a verificação levou em torno de 3

[10] Código fonte do Mirai *botnet*. Disponível em: <https://github.com/jgamblin/Mirai-Source-Code>.

minutos, enquanto testes em uma rede corporativa levou em torno de 20 minutos e na rede da Universidade levou em torno de 40 minutos.

A funcionalidade de executar automaticamente o teste de vulnerabilidade ao abrir o THC Hydra foi retirada por que o processo pode demorar muito tempo em determinadas redes. Esta alteração foi realizada para evitar que o programa fique processando por muito tempo caso o usuário tenha clicado na opção por engano.

Figura 41 – Simulação integração jNetMap e THC Hydra.



Fonte: Autoria própria.

O próximo teste realizado foi realizar uma avaliação das funcionalidades da integração em dispositivos mapeados no *software* Angry IP Scanner seguindo os passos descritos no Quadro 7.

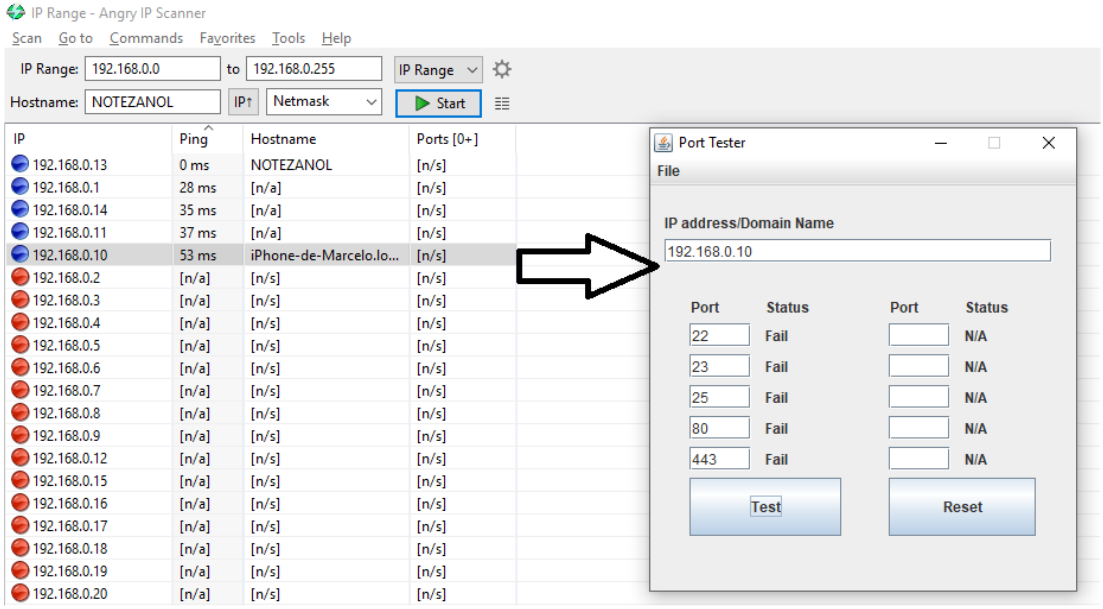
Quadro 7- Caso de Teste 02 – Integração Angry IP Scanner.

#	Ação	Resultado Esperado
1	Abrir ferramenta principal.	Apresentar tela principal da ferramenta.
2	Clicar no botão Angry IP Scanner.	Abrir a ferramenta Angry IP Scanner.
3	Clicar no botão Start.	Escanear dispositivos na rede e apresentar na lista.
4	Clicar bom o botão direito em um dispositivo/ Clicar no menu “Commands”.	Apresentar menu de opções na tela.
5	Clicar na opção “Open in Port Tester”.	Abrir a ferramenta Port Tester com o endereço do dispositivo carregado e executar a verificação das portas pré-definidas exibindo o resultado ao lado do número das portas.
6	Clicar na opção “Open in THC Hydra”.	Abrir a tela da ferramenta THC Hydra.
7	Clicar no botão “Iniciar” para executar o teste para o dispositivo selecionado.	Apresentar no campo “Log” o resultado do teste de vulnerabilidade.
8	Clicar na opção de fechar as ferramentas.	Fechar as ferramentas.

Fonte: Autoria própria.

Após clicar no botão principal Angry IP Scanner, a ferramenta foi iniciada sem exibir resultados de IPs da rede e com o campo “IP Range” preenchido com os valores de 192.168.0.0 até 192.168.0.255 automaticamente. O carregamento da faixa de IPs é uma funcionalidade original da ferramenta. Ao clicar no botão “Start” o escaneamento foi iniciado e após 20 segundos a lista de dispositivos foi exibida na tela principal, identificando apenas cinco dispositivos ativos. A seguir foi utilizado as novas opções de menu desenvolvidas para abrir a ferramenta Port Tester (“Open in Port Tester”) e automaticamente, após abrir a ferramenta, foi realizado o teste de verificação de portas abertas para o dispositivo selecionado (IP 192.168.0.10) exibindo o resultado do teste de cada porta ao lado direito dela. Neste caso, todas as portas apresentaram o resultado “Fail” indicando que nenhuma delas estava aberta conforme ilustrado na figura 42. O mesmo procedimento foi relizado para os outros dispositivos da rede mapeada para validar que não seja aberta mais de uma vez a mesma ferramenta caso ela já esteja aberta. Clicando na opção “Open in Port Tester” para o dispositivo 192.168.0.11 e 192.168.0.14 o endereço de IP dentro do Port Tester foi alterado para o solicitado sem abrir uma nova instância da ferramenta.

Figura 42 – Simulação integração Angry IP Scanner e Port Tester.

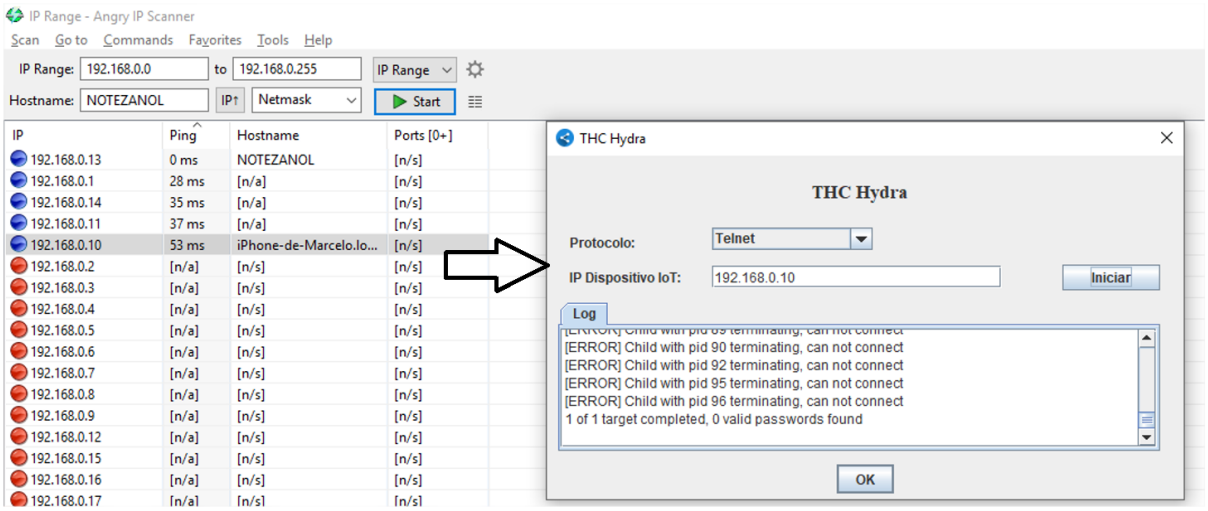


Fonte: Autoria própria.

Concluindo que as ferramentas do teste anterior estavam funcionais e a integração concluída, foi realizado o teste com as ferramentas Angry IP Scanner e THC Hydra.

O teste foi realizado partindo da nova opção desenvolvida nomeada por “Open in THC Hydra” para abrir a ferramenta THC Hydra. Ao clicar na opção, a tela desenvolvida diretamente no *software* IoT Manager and Security Toolkit é exibida com o endereço de IP do dispositivo selecionado (192.168.0.10) preenchido no campo “IP Dispositivo IoT” e a opção “Telnet” no campo “Protocolo” para o teste de vulnerabilidade. Como a funcionalidade de execução automática do teste foi retirada a fins de usabilidade e desempenho, foi necessário clicar no botão “Iniciar” para a execução do processo. O resultado da verificação apontou que nenhum usuário e senha utilizado pelo Mirai *botnet* é válido, conforme ilustrado na figura 43.

Figura 43 – Simulação integração Angry IP Scanner e THC Hydra.



Fonte: Autoria própria.

O próximo teste realizado foi executar a ferramenta Port Tester diretamente da tela principal do *software* IoT Manager and Security Toolkit, e validar suas funcionalidades conforme os passos descritos no Quadro 8.

Quadro 8- Caso de Teste 03 – Integração Port Tester.

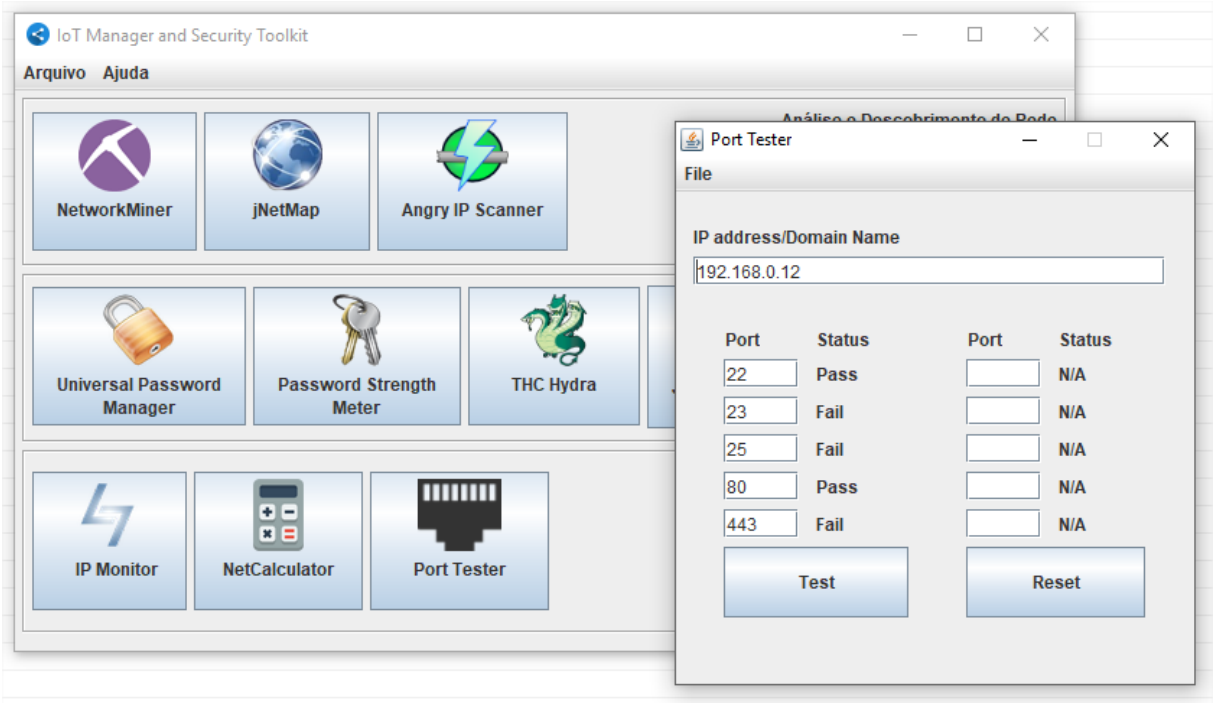
#	Ação	Resultado Esperado
1	Abrir ferramenta principal.	Apresentar tela principal da ferramenta.
2	Clicar no botão Port Tester.	Abrir a ferramenta Port Tester.
3	Informar endereço de IP e clicar em “Test”.	Retornar o resultado do teste para as portas informadas nos campos.
4	Clicar no botão “Reset”.	Limpar o conteúdo dos campos da tela.
5	Clicar na opção de fechar a ferramenta.	Fechar a ferramenta.

Fonte: Autoria própria.

Após abrir a ferramenta e clicar no botão principal “Port Tester” na seção Administração de Rede, a ferramenta foi iniciada com o campo “IP address/Domain Name” em branco e com as portas 22, 23, 25, 80 e 443 carregadas automaticamente. O campo do IP não veio carregado automaticamente pois partindo da tela inicial não se tem o endereço de um dispositivo como havia nos testes anteriores. Informando o endereço de IP 192.168.0.12 manualmente e clicado no botão “Test” o resultado da verificação foi exibido ao lado cada porta preenchida. Neste caso, apenas as portas 22 e 80 estavam liberadam apresentando o resultado “Pass”, enquanto as

demais verificadas apresentavam o resultado “Fail” conforme ilustrado na figura 44. Ao clicar no botão “Reset” todos campos foram apagados e o resultado da verificação foi alterado para “N/A” finalizando o caso de testes.

Figura 44 – Simulação integração Port Tester.



Fonte: Autoria própria.

O próximo teste realizado foi o da ferramenta THC Hydra chamada diretamente da tela principal do *software* IoT Manager and Security Toolkit, e validar suas funcionalidades conforme os passos descritos no Quadro 9.

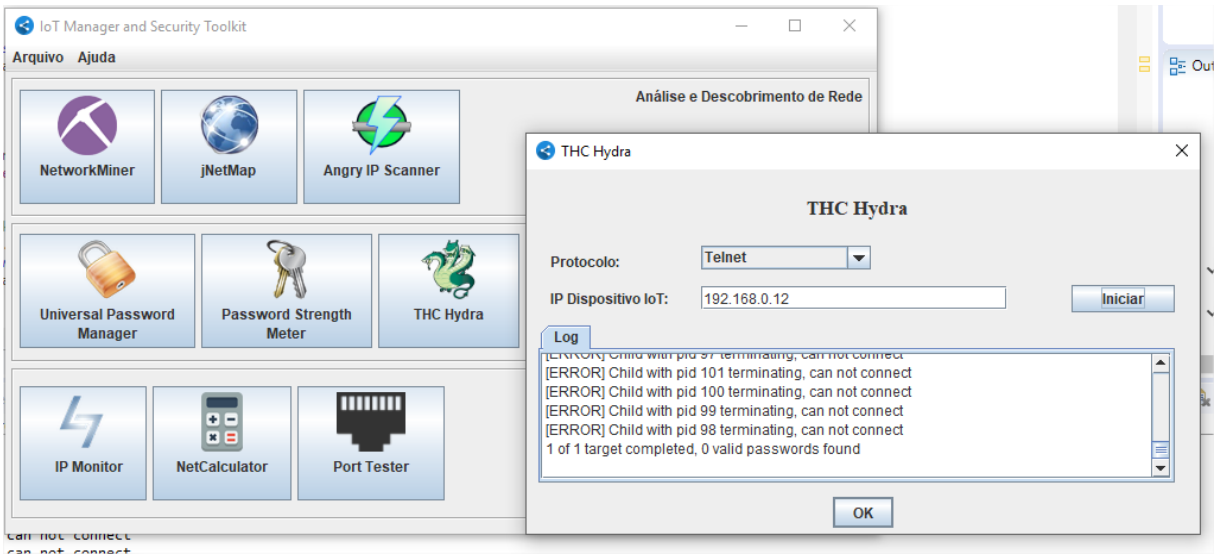
Quadro 9- Caso de Teste 04 – Integração THC Hydra.

#	Ação	Resultado Esperado
1	Abrir ferramenta principal.	Apresentar tela principal da ferramenta.
2	Clicar no botão THC Hydra.	Abrir a tela da ferramenta THC Hydra.
3	Informar endereço de IP, selecionar o protocolo e clicar em “Iniciar”.	Apresentar no campo “Log” o resultado do teste de vulnerabilidade.
4	Clicar na opção de fechar a ferramenta.	Fechar a tela da ferramenta.

Fonte: Autoria própria.

Após abrir a ferramenta e clicar no botão principal “THC Hydra” na seção Segurança e Senhas, a ferramenta foi iniciada com o campo “IP Dispositivo IoT” em branco. O campo do IP não veio carregado automaticamente pois partindo da tela inicial não se tem o endereço de um dispositivo como havia nos testes anteriores. Informando o endereço de IP 192.168.0.12 manualmente e clicando no botão “Iniciar”, após em torno de 40 segundos o resultado do teste de vulnerabilidade foi exibido no campo “Log” indicando que nenhum usuário e senha utilizado pelo Mirai *botnet* é válido, conforme ilustrado na figura 45, finalizando o caso de testes.

Figura 45 – Simulação integração THC Hydra.



Fonte: Autoria própria.

O próximo teste realizado foi a integração da ferramenta John The Ripper para validar a sua funcionalidade de quebra de senhas em *hash*. A ferramenta é chamada através da tela principal do software e o teste decorreu conforme os passos descritos no Quadro 10.

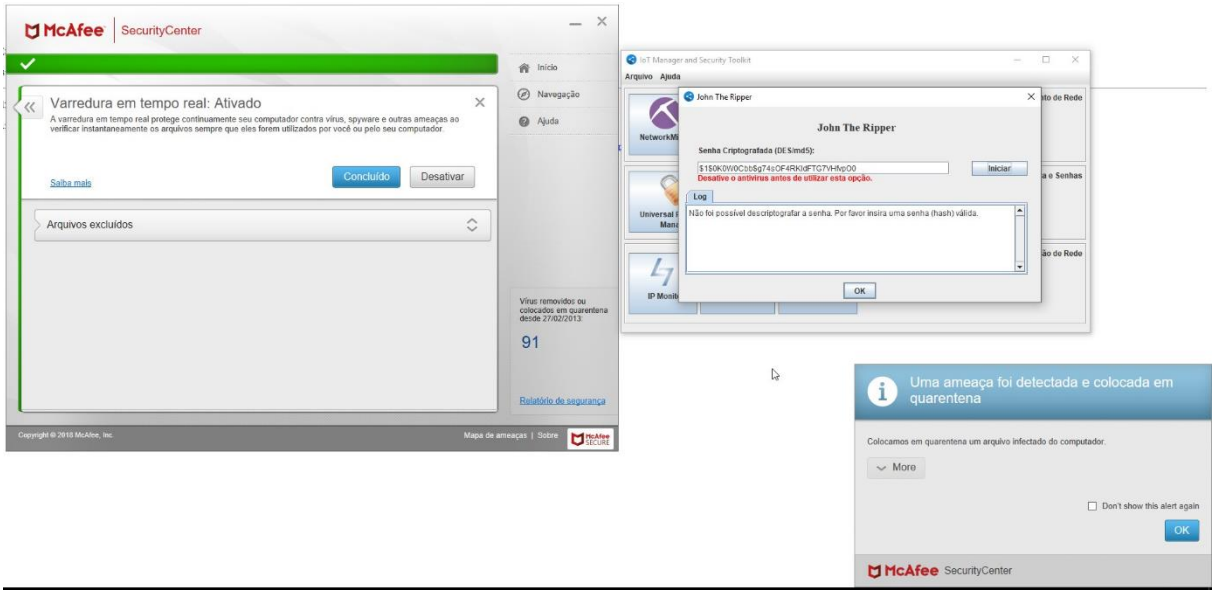
Quadro 10- Caso de Teste 05 – Integração John the Ripper.

#	Ação	Resultado Esperado
1	Abrir ferramenta principal.	Apresentar tela principal da ferramenta.
2	Clicar no botão John the Ripper.	Abrir a tela da ferramenta John the Ripper.
3	Informar uma senha criptografada e clicar em “Iniciar”.	Apresentar no campo “Log” o resultado da descriptografia.
4	Clicar na opção de fechar a ferramenta.	Fechar a tela da ferramenta.

Fonte: Autoria própria.

Após abrir a ferramenta e clicar no botão principal “John the Ripper” na seção Segurança e Senhas, a ferramenta foi iniciada com o campo “Senha Criptografada (DES/md5)” em branco. O próximo passo foi inserir a senha criptografada “1zck74fII$yNdKJXuxodQLzhMlFj3sC1” [11] para verificação. Como ao realizar o teste o antivírus McAfee SecurityCenter estava ativado, o executável do programa John the Ripper foi movido para a quarentena e foi exibida a mensagem de erro “Não foi possível descriptografar a senha. Por favor insira uma senha (hash) válida.” no campo “Log” da tela, conforme exibido na figura 46.

Figura 46 – Uso John the Ripper com antivírus ativado.

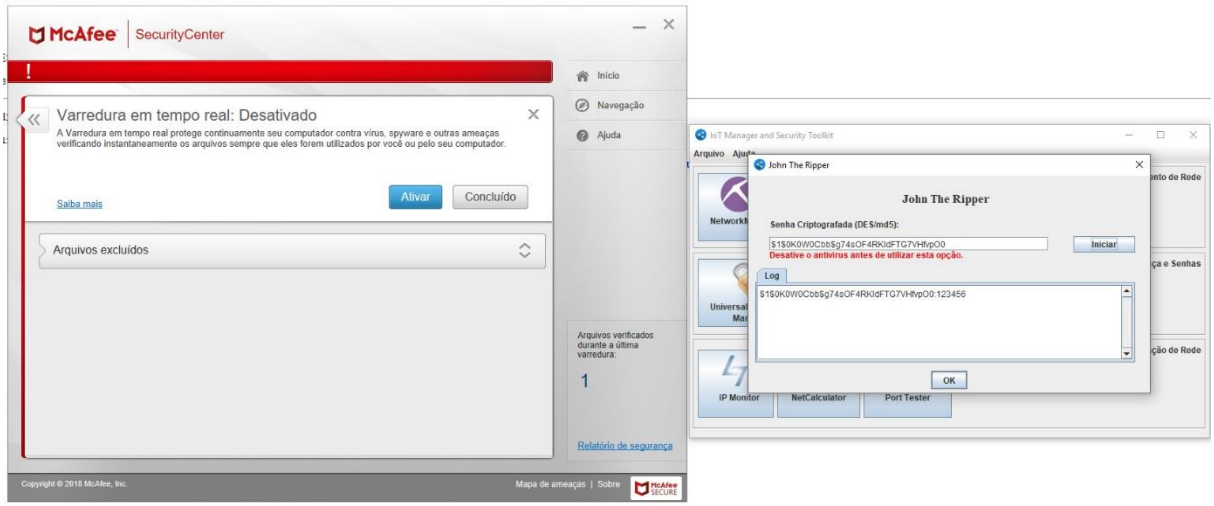


Fonte: Autoria própria.

Para continuar os testes, foi necessário desativar o antivírus e recolocar o executável da ferramenta na pasta do projeto. Ao clicar novamente no botão “Iniciar” foi exibido no campo “Log” o retorno da descriptografia com o formato <senha criptografada>:<senha descriptografada>. Foram executados os testes utilizando a criptografia md5 e DES, conforme ilustrado nas figuras 47 e 48, finalizando os casos de testes.

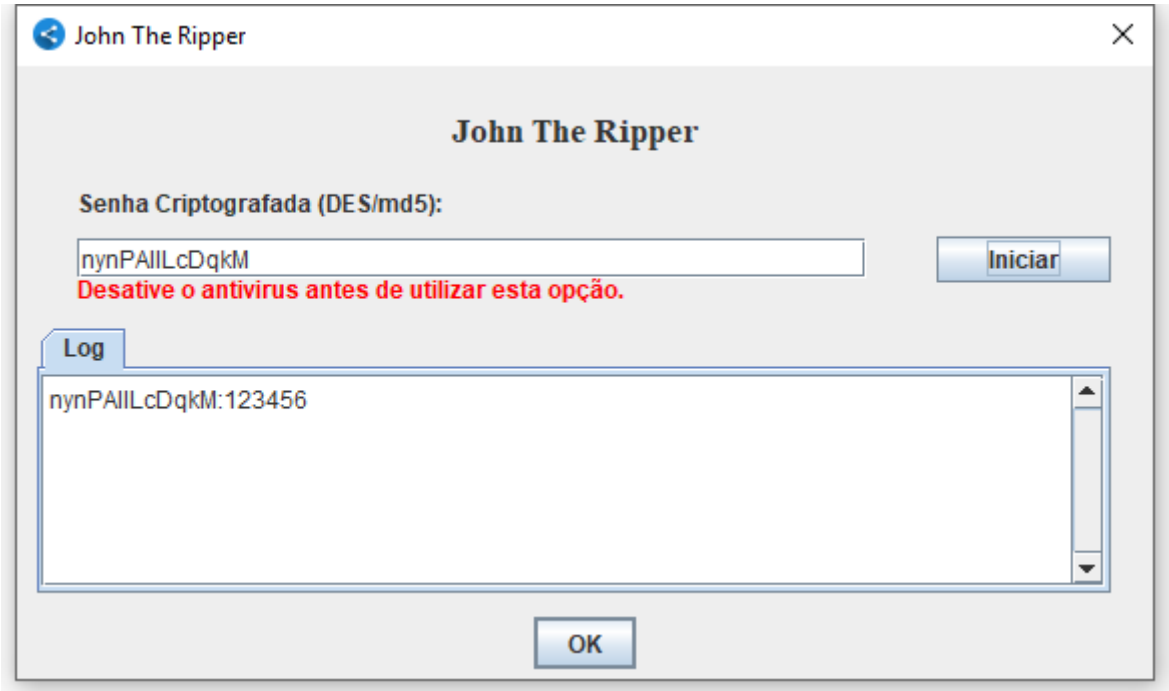
[11] Foi utilizado um site que permite criar um usuário e senha com ambos tipos de criptografia (DES e md5). Disponível em: <http://sherylcanter.com/encrypt.php>.

Figura 47 – Uso John the Ripper com antivírus desativado. Criptografia: md5



Fonte: Autoria própria.

Figura 48 – Simulação integração John the Ripper. Criptografia: DES.



Fonte: Autoria própria.

6.2 TESTES DE PREVENÇÃO DE ATAQUES DDOS E MALWARE

Os diferentes tipos de *malware* estudados neste trabalho em sua grande maioria dependem da cultura do usuário o que impossibilita a ação desta na prevenção. *Rootkits, Worms* e *Ramsomeware* são altamente disseminados através de *e-mails, sites* e mensagens aleatórias de amigos contendo *links* maliciosos. Cavalos de Tróia e *Spywares* em geral infectam os dispositivos através do *download* de *softwares* maliciosos que se passam por programas legítimos enganando o usuário.

Backdoors podem ser evitados através da utilização dos novos recursos incluídos no *software*, com o uso combinado das ferramentas Angry IP Scanner e Port Tester para escaneamento de portas abertas em diferentes dispositivos e a partir disto realizar uma configuração e bloquear portas que não são utilizadas pelos dispositivos IoT.

Ataques DDoS dos tipos UDP *flood* e ICMP *flood*, são ataques que consistem em enviar diversos pacotes para portas aleatórias dos dispositivos, sobrecarregando o sistema caso uma dessas portas esteja vulnerável. Estes ataques podem ser evitados utilizando a ferramenta Port Tester para fechar portas não utilizadas pela atividade principal dos dispositivos IoT.

As ferramentas para identificação de dispositivos com vulnerabilidade contra ataques DDoS, neste caso ataques do tipo HTTP *flood,* foram testadas em dois ambientes reais. O primeiro deles foi uma rede sem fio (Wi-Fi) de um ambiente corporativo, uma fábrica de *software* localizada em Caxias do Sul, com aproximadamente 200 funcionários é considerada uma das melhores empresas para trabalhar do país. Em segundo, foi uma rede sem fio (Wi-Fi) do Laboratório de Informática da Universidade de Caxias do Sul, uma das principais universidades do estado do Rio Grande do Sul com aproximadamente 948 professores titulados e 800 laboratórios para todas áreas de ensino. Possíveis situações reais foram criadas para que fosse possível analisar o comportamento das ferramentas.

Foi elaborado um caso de teste para validar a situação dos dispositivos quanto à vulnerabilidade contra ataques DDoS, conforme descrito no Quadro 11.

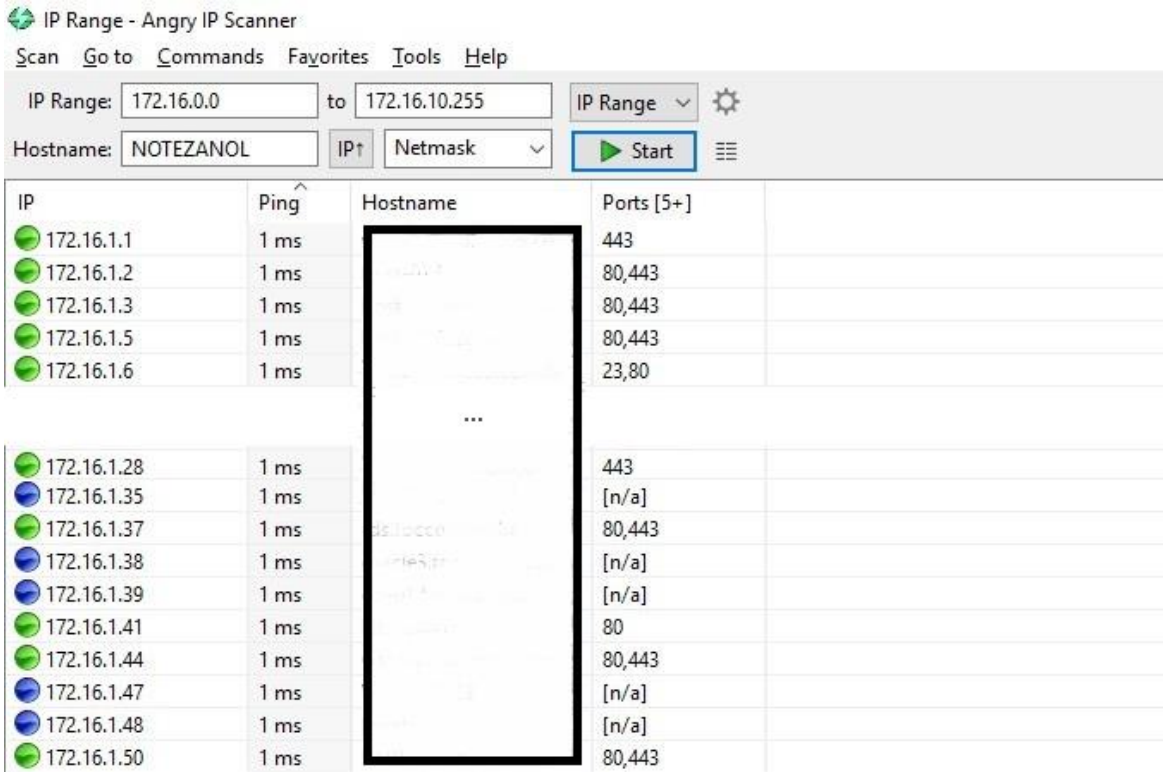
Quadro 11- Caso de Teste 06 – Vulnerabillidade DDoS e Malware.

#	Ação	Resultado Esperado
1	Abrir ferramenta principal.	Apresentar tela principal da ferramenta.
2	Clicar no botão Angry IP Scanner.	Abrir a ferramenta Angry IP Scanner.
3	Clicar no botão Start.	Escanear dispositivos na rede e apresentar na lista.
4	Clicar bom o botão direito em um dispositivo/ Clicar no menu “Commands”.	Apresentar menu de opções na tela.
5	Clicar na opção “Open in Port Tester”.	Abrir a ferramenta Port Tester com o endereço do dispositivo carregado e executar a verificação das portas pré-definidas exibindo o resultado ao lado do número das portas.
6	Clicar na opção “Open in THC Hydra”.	Abrir a tela da ferramenta THC Hydra.
7	Clicar no botão “Iniciar” para executar o teste para o dispositivo selecionado.	Apresentar no campo “Log” o resultado do teste de vulnerabilidade.
8	Clicar na opção de fechar as ferramentas.	Fechar as ferramentas.

Fonte: Autoria própria.

No teste das ferramentas na rede corporativa, após clicar no botão principal Angry IP Scanner, a ferramenta foi iniciada sem exibir resultados de IPs da rede e com o campo “IP Range” preenchido com os valores de 172.16.0.0 até 172.16.30.255 automaticamente. Ao clicar no botão “Start” o escaneamento foi iniciado e após aproximadamente 20 minutos a lista de dispositivos foi exibida na tela principal, identificando 239 dispositivos ativos, sendo que, 117 dispositivos possuíam portas ativas, representando o total de 48,9%. Como o teste de portas ativas utilizando a ferramenta Port Tester seria muito oneroso devido a quantidade de dispositivos, foi utilizado um recurso original da ferramenta. Este recurso possui a funcionalidade do Port Tester e exibe diretamente na lista de dispositivos quais portas estão abertas, porém é necessário configurar que portas devem ser verificadas antes de iniciar o escaneamento, conforme ilustrado na figura 49.

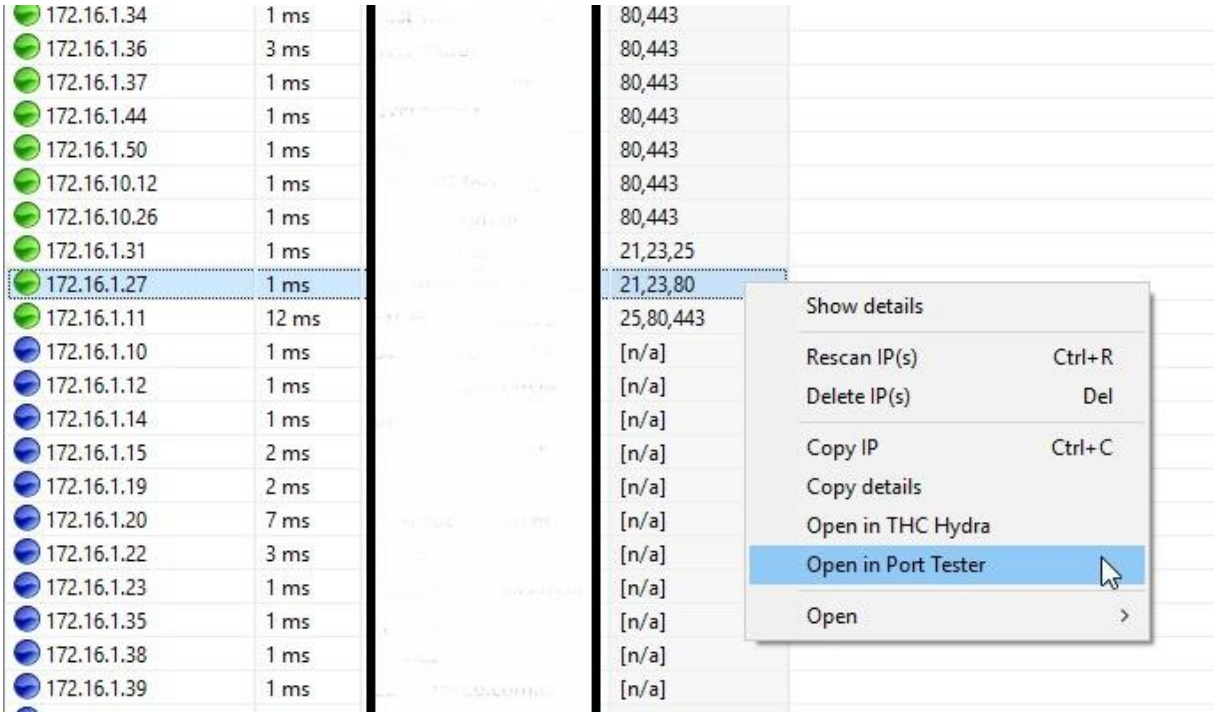
Figura 49 – Lista dispositivos conectado em uma Wi-fi corporativa.



Fonte: Autoria própria.

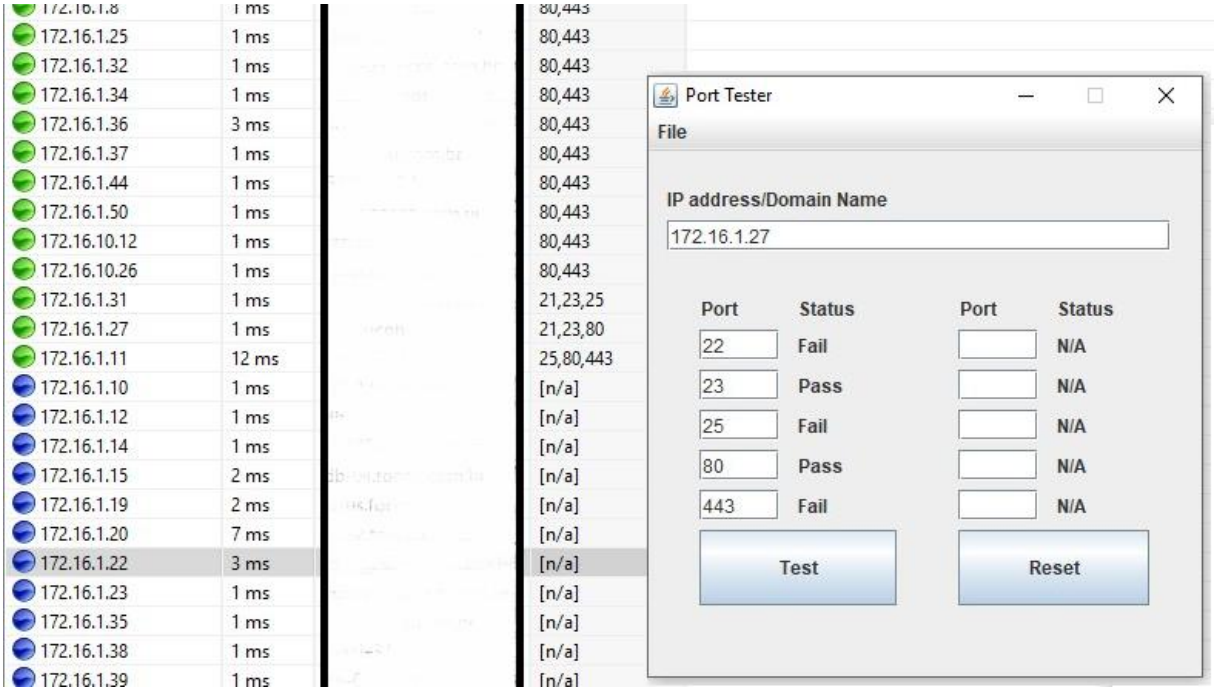
Dentre todos dispositivos IoT ativos na rede apenas dois, identificados pelos IPs 172.16.1.27 e 172.16.1.31, possuíam a porta Telnet (23) vulnerável. Para validação da informação foi utilizada a ferramenta Port Tester em ambos dispositivos e concluído que a informação estava correta. Na Figura 50 é apresentada a utilização do novo menu criado e na Figura 51 é apresentado o resultado do teste para as portas 22, 23, 25, 80, 443. As portas 22, 25 e 443 apresentaram o resultado "*Fail*" indicando que as portas não estavam vulneráveis e as portas 23 e 80 apresentaram o resultado "*Pass*" indicando que as portas estavam abertas. Nos campos em branco é apresentado o resultado "N/A" por não terem sido informadas.

Figura 50 – Detalhe menu aberto para teste de portas abertas.



Fonte: Autoria própria.

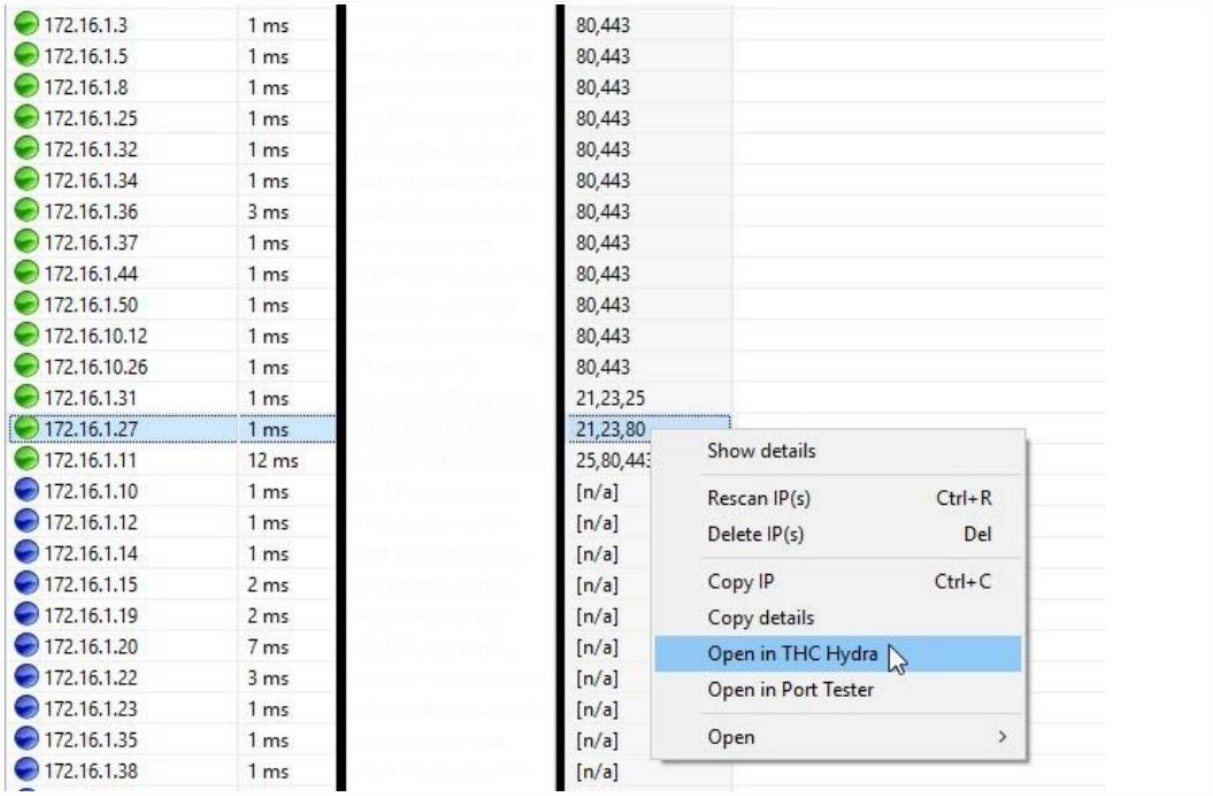
Figura 51 – Tela com o resultado do teste na ferramenta Port Tester.



Fonte: Autoria própria.

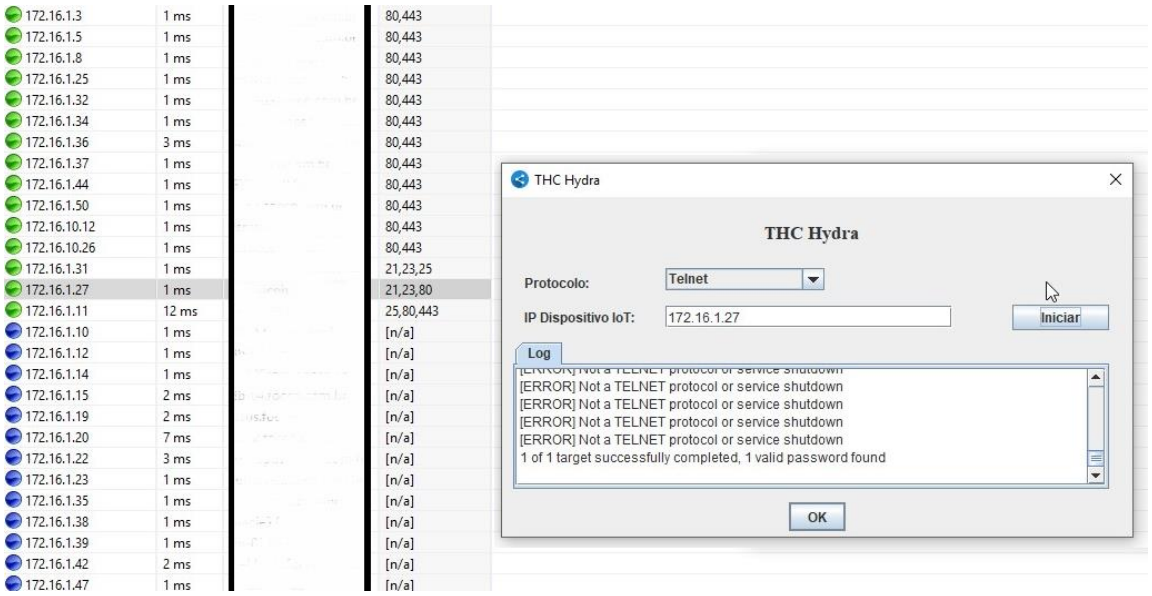
Com esta informação o próximo teste realizado na rede corporativa foi utilizar a ferramenta THC Hydra em conjunto com o Angry IP Scanner para verificar se os dispositivos possuíam um conjunto de usuário e senha utilizado pelo Mirai *botnet.* Clicando com o botão direito na nova opção desenvolvida “Open in THC Hydra”, a ferramenta foi aberta com o IP do dispositivo simulado e na sequência foi iniciado o teste de vulnerabilidade. Após pouco menos de 3 minutos de verificação para cada dispositivo foi constatado que eles possuíam dados de autenticação padrões, concluindo que os dispositivos estavam vulneráveis à ataques DDoS (figuras 52 e 53).

Figura 52 – Detalhe menu aberto para teste de vulnerabilidade DDoS.



Fonte: Autoria própria.

Figura 53 – Tela com resultado do teste de vulnerabilidade DDoS.



Fonte: Autoria própria.

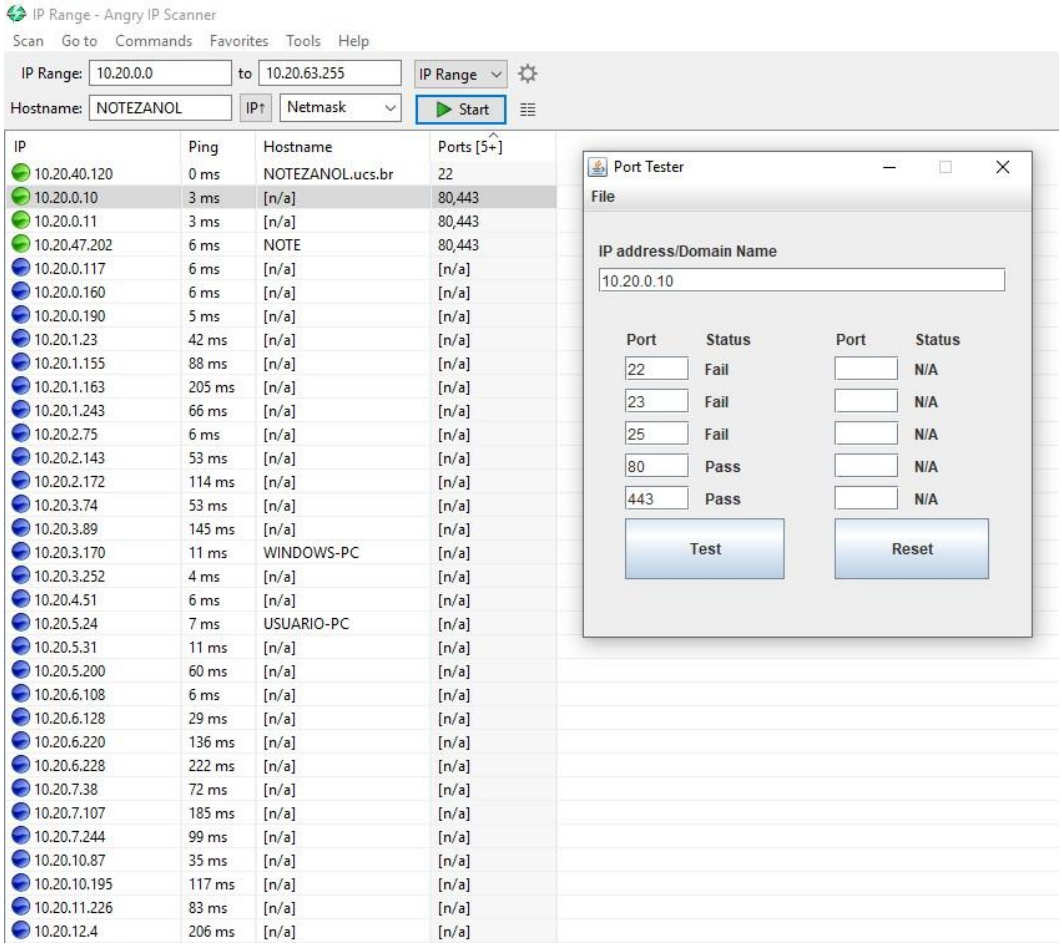
Após a conclusão dos testes na rede corporativa, foi gerada uma listagem com todos dispositivos que possuíam as portas FTP (21), SSH (22), Telnet (23) e HTTP/HTTPS (80/443) abertas, após isso, encaminhado para o responsável da rede para realizar uma intervenção manual e corrigir as falhas de segurança contra ataques DDoS e *malware*.

O próximo teste das ferramentas em ambiente real foi realizado na rede sem fio da Universidade. Após clicar no botão principal Angry IP Scanner, a ferramenta foi iniciada sem exibir resultados de IPs da rede e com o campo “IP Range” preenchido com os valores de 10.20.0.0 até 10.20.63.255 automaticamente. Ao clicar no botão “Start” o escaneamento foi iniciado e após aproximadamente 40 minutos a lista de dispositivos foi exibida na tela principal, identificando 266 dispositivos ativos, sendo que, apenas 4 dispositivos possuíam portas ativas, representando o total de 1,5%.

Dentre os dispositivos ativos as portas identificadas como abertas foram 22, 80 e 443 e para criar uma consistência das informações, foi utilizada a ferramenta integrada Port Tester, conforme ilustrado na figura 54.

Devido à falta de dispositivos com a porta Telnet (23) ativa, o teste de vulnerabilidade na rede sem fio (Wi-fi) através do uso da ferramenta THC Hydra não pôde ser concluído, chegando à conclusão de que a rede está segura contra os tipos de ataques DDoS verificados com o *software*.

Figura 54 – Teste da ferramenta Port Tester rede Wi-fi universidade.



Fonte: Autoria própria.

6.3 CONSIDERAÇÕES FINAIS

Os testes foram concluídos com sucesso em ambos ambientes. No ambiente corporativo foram identificado dois dispositivos com falhas graves de segurança, enquanto no ambiente da Universidade nenhum dispositivo com falhas de segurança contra ataques DDoS foi encontrado. No ambiente que havia problemas de segurança, foi entrado em contato com o responsável da rede para realizar os ajustes necessários. A integração das novas ferramentas e funcionalidades foram devidamente desenvolvidas e testadas agregando valor ao *software.*

É sempre válido manter a configuração de segurança em navegadores de *Internet* configurados com alta segurança a fim de evitar a tentativa de invasão e infecção por *malwares* que utilizem fraquezas através de navegadores.

7. CONCLUSÃO

O constante avanço e estudo em tecnologia garante uma evolução tecnológica que cada vez mais adentra o cotidiano através de dispositivos IoT. Comprar mantimentos em um mercado ou até analisar a imagem de câmeras de segurança em tempo real podem ser realizados através de um dispositivo móvel como um *smartphone* ou um computador conectado à internet independente da sua posição atual.

A fraqueza inerente da segurança em dispositivos IoT, combinada ao lançamento do código fonte do *botnet* Mirai motivou a pesquisa e o incremento de funcionalidades à uma ferramenta de código aberto, permitindo que o responsável pela segurança da rede possa mapear todos dispositivos conectados e identificar quais possuem portas desprotegidas e vulnerabilidades à ataques DDoS utilizando um *software* que concentre todas funcionalidades.

Ferramentas atuais para segurança contra ataques DDoS possuem funcionalidades limitadas ou muito específicas, como por exemplo, o Port Tester 1.0 permite apenas identificar portas desprotegidas, enquanto outras, como o Angry IP Scanner, permitem apenas escanear dispositivos conectados na rede, obrigando o responsável de segurança da rede a possuir diferentes aplicações, uma para cada funcionalidade específica.

A primeira etapa deste trabalho, portanto, visou um estudo para compreender os diferentes tipos de ataques à dispositivos IoT fazendo um detalhamento do conceito de ataques DoS e DDoS. Com isso em mente foi feito um levantamento de ferramentas open-source com funcionalidades que agregariam valor ao software e que combinadas, construíssem um software completo. Com isso, o objetivo de realizar a pesquisa e levantamento de ferramentas foi atingido.

A etapa seguinte foi utilizar o software BYOD Manager Kit como ferramenta base para o desenvolvimento da integração e adição de funcionalidades estudadas. O software teve a adição de quatro ferramentas diferentes, que possuem diferentes objetivos finais, aumentando a abrangência de atuação, originando a suíte de ferramentas chamada IoT Manager and Security Toolkit e atingindo ao objetivo de incrementar as novas funcionalidades e disponibilizar um novo *software* de segurança *open-source*.

Em alguns momentos foi necessário realizar uma pesquisa de outras ferramentas para substituir as inicialmente selecionadas, porém as dificuldades de integração entre Windows e Linux e compilação das ferramentas originais foram superadas viabilizando o plano de trabalho.

Em sua etapa de testes o novo software possibilitou a descoberta de dispositivos com falhas graves de segurança e que colocavam em risco a rede em que estavam conectados. Sem o uso

do software o mesmo processo poderia levar dias para ser executado, o tempo necessário é diretamente relacionado à quantidade de dispositivos conectados na rede. A ferramenta Angry IP Scanner, por exemplo, permitiu que uma faixa de 255 IPs diferentes fossem escaneados em questão de segundos e a integração com o Port Tester e o THC Hydra tornou a verificação de vulnerabilidade contra ataques DDoS extremamente acertiva.

O software desenvolvido pode auxiliar o profissional de TI na administração e segurança de redes IoT. As ferramentas do software são indispensáveis no dia a dia do administrador de rede. Elas possibilitam monitorar o que está sendo transmitido pela rede, incluindo formatos específicos de arquivos, um escaneamento rápido de todos dispositivos conectados, uma visualização gráfica da rede, verificação de dispositivos vulneráveis a diferentes tipos de ataques e um recurso para recuperação de senhas criptografadas. A solução proposta atendeu às diversas necessidades e as novas funcionalidades foram de grande valia para o projeto.

Algumas melhorias podem ser realizadas futuramente neste projeto. Em decorrência do curto espaço de tempo não foi possível adicionar outras funcionalidades à integração. Por exemplo, poderia ser incluída na integração a ferramenta Aircrack-ng, esta ferramenta poderia ajudar o profissional de TI a melhorar a segurança de seu ambiente antes que outra pessoa mal-intencionada lhe cause sérios problemas. Senhas de redes sem fio que utilizam padrões de segurança como WEP, WPA ou WPA2 podem representar uma grande vulnerabilidade à segurança da rede de computadores. Além disso poderia ser disponibilizada funcionalidades para análise de links em *e-mails* possibilitando uma prevenção mais completa ao usuário, aumentando a abrangência do *software*.

Com a escassez de ferramentas e artigos acadêmicos, este trabalho torna a análise e prevenção contra ataques DDoS e *malware* mais acessível e, talvez, propõe ideias para universidades e comunidades de programadores *open-source* para que novas ferramentas atendam à constante demanda de soluções modernas.

REFERÊNCIAS BIBLIOGRÁFICAS

ALECRIM, Emerson. **Malwares: o que são e como agem, 2017**. Disponível em: <https://www.infowester.com/malwares.php>. Acesso em: 19 setembro 2017.

ANTONAKAKIS M. et al. **Understanding the Mirai Botnet. Proceedings of the 26th USENIX Security Symposium, 2017.**

BEKERMAN, Dima. **New Mirai Variant Launches 54 Hour DDoS Attack against US College, 2017.** Disponível em: <https://www.incapsula.com/blog/new-mirai-variant-ddos-us-college.html>. Acesso em: 20 setembro 2017.

BERBERT, Lucia. **Especialistas advertem que dispositivos de IoT podem se transformar em armas de ataques cibernéticos, 2017.** Disponível em: <http://teletela.com.br/teletime/08/06/2017/especialistas-advertem-que-dispositivos-de-iot-podem-se-transformar-em-armas-de-ataques-ciberneticos>. Acesso em: 27 setembro 2017.

CELESTINO, André Luis. **O conceito e as dúvidas sobre o MVC, 2014.** Disponível em: <https://www.profissionaisti.com.br/2014/10/o-conceito-e-as-duvidas-sobre-o-mvc/>. Acesso em: 31 outubro 2017.

CIO. **Ataques de DDoS chegaram a 800 Gbps impulsionados por dispositivos de IoT, 2017.** Disponível em: <http://cio.com.br/noticias/2017/02/15/ataques-de-ddos-chegaram-a-800-gbps-impulsionados-por-dispositivos-de-iot>. Acesso em: 24 agosto 2017.

COMPUTERWORLD. **Rede Mirai: Ataques virtuais a dispositivos de IoT se tornam mais comuns.** Disponível em: < http://computerworld.com.br/rede-mirai-ataques-virtuais-dispositivos-de-iot-se-tornam-mais-comuns>. Acesso em: 16 de agosto de 2017

EASTWOOD, Gary. **4 critical security challenges facing IoT, 2017.** Disponível em: <https://www.networkworld.com/article/3166106/internet-of-things/4-critical-security-challenges-facing-iot.html>. Acesso em: 24 agosto 2017.

ECLIPSE. **Eclipse Wiki, 2017.** Disponível em: <https://wiki.eclipse.org/Eclipse/Installation>. Acesso em: 19 outubro 2017.

FIDELIS, Donizete. **Proxy Reverso**. São Paulo: Faculdade de Tecnologia Ourinhos, 2013.

GULZAR, Nadir. **Fast Track to Struts: What id Does and How**. p. 1-29, 2002. Disponível em: <http://media.techtarget.com/tss/static/articles/content/StrutsFastTrack/StrutsFastTrack.pdf>. Acesso em: 31 outubro 2017.

HERZBERG, Ben; BEKERMAN, Dima; ZEIFMAN, Igal. **Breaking Down Mirai: An IoT DDoS Botnet Analysis, 2016.** Disponível em: <https://www.incapsula.com/blog/malware-analysis-mirai-ddos-botnet.html>. Acesso em: 21 agosto 2017.

HILTON, Scott. **Dyn Analysis Summary Of Friday October 21 Attack, 2016.** Disponível em: <http://hub.dyn.com/dyn-blog/dyn-analysis-summary-of-friday-october-21-attack>. Acesso em: 21 agosto 2017.

INCAPSULA. **Malware Types.** Disponível em: <https://www.incapsula.com/web-application-security/malware-detection-and-removal.html>. Acesso em: 07 outubro 2017.

JNA. **Java Native Access, 2017.** Disponível em: < https://github.com/java-native-access/jna>. Acesso em: 02 outubro 2017.

KLABA, Octave. **Last days, we got lot of huge DDoS. Here, the list of "bigger that 100Gbps" only. You can see the simultaneous DDoS are close to 1Tbps !, 2016.** Disponível em: <https://twitter.com/olesovhcom/status/778830571677978624>. Acesso em: 21 agosto 2017.

KREBS, Brian. **KrebsOnSecurity Hit With Record DDoS, 2016.** Disponível em: <https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/>. Acesso em: 21 agosto 2017.

MANCINI, Mônica**. Internet das Coisas: História, Conceitos, Aplicações e Desafios, 2017.** Disponível em: <https://pmisp.org.br/documents/acervo-arquivos/241-internet-das-coisas-historia-conceitos-aplicacoes-e-desafios/file>. Acesso em: 24 agosto 2017.

MENEZES, Wander. **Rede Mirai: Ataques virtuais a dispositivos de IoT se tornam mais comuns, 2017.** Disponível em: <http://http://computerworld.com.br/rede-mirai-ataques-virtuais-dispositivos-de-iot-se-tornam-mais-comuns> Acesso em: 27 setembro 2017.

MEOLA, Andrew. **What is the Internet of Things (IoT)?, 2016.** Disponível em: <http://www.businessinsider.com/what-is-the-internet-of-things-definition-2016-8>. Acesso em: 24 agosto 2017.

ORACLE. **Windows System Requirements for JDK and JRE, 2016.** Disponível em: <https://docs.oracle.com/javase/8/docs/technotes/guides/install/windows_system_requirements.html#BABIFDGD>. Acesso em: 23 Outubro 2017.

OWASP. **OWASP Internet of Things Project.** Disponível em: <https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project>. Acesso em: 07 outubro 2017.

PERINI, Vinícius Lahm. **Integração de Ferramentas de Administração e Segurança BYOD.** 154f. Trabalho de conclusão de curso (Ciência da Computação) – Área do Conhecimento de Ciências Exatas e Engenharias, Universidade de Caxias do Sul, Caxias do Sul, 2017.

SECURITYLOCK. **O que é uma camada 7 Ataque DDoS?, 2016.** Disponível em: <http://securitylock.com.br/o-que-e-uma-camada-7-ataque-ddos/>. Acesso em: 19 setembro 2017.

STALLINGS, William. **Cryptography and Network Security: Principles and Practice.** 5 ed. Prentice Hall, 2013.

SYMANTEC. **Ransom.Wannacry, 2017.** Disponível em: <https://www.symantec.com/security_response/writeup.jsp?docid=2017-051310-3522-99>. Acesso em: 19 setembro 2017.

US-CERT. **Security Tip (ST04-015), Understanding Denial-of-Service Attacks.** Disponível em: <https://www.us-cert.gov/ncas/tips/ST04-015>. Acesso em: 19 setembro 2017.

VERISIGN. **Ataque de inundação SYN.** Disponível em: <https://www.verisign.com/pt_BR/security-services/ddos-protection/syn-flood/index.xhtml>. Acesso em: 19 setembro 2017.

W3II. **IPv4 Estrutura de Pacotes.** Disponível em: <http://www.w3ii.com/pt/ipv4/ipv4_packet_structure.html>. Acesso em: 20 setembro 2017.

APÊNDICE A

IOT MANAGER AND SECURITY TOOLKIT
SUÍTE DE FERRAMENTAS OPEN-SOURCE PARA ADMINISTRAÇÃO E SEGURANÇA DE REDES

MANUAL DO USUÁRIO

Desenvolvido por: Vinícius Perini e Marcelo Zanol

Universidade de Caxias do Sul

Área do Conhecimento de Ciências Exatas e Engenharias

Bacharelado em Sistemas de Informação

Caxias do Sul, 2018.

SUMÁRIO

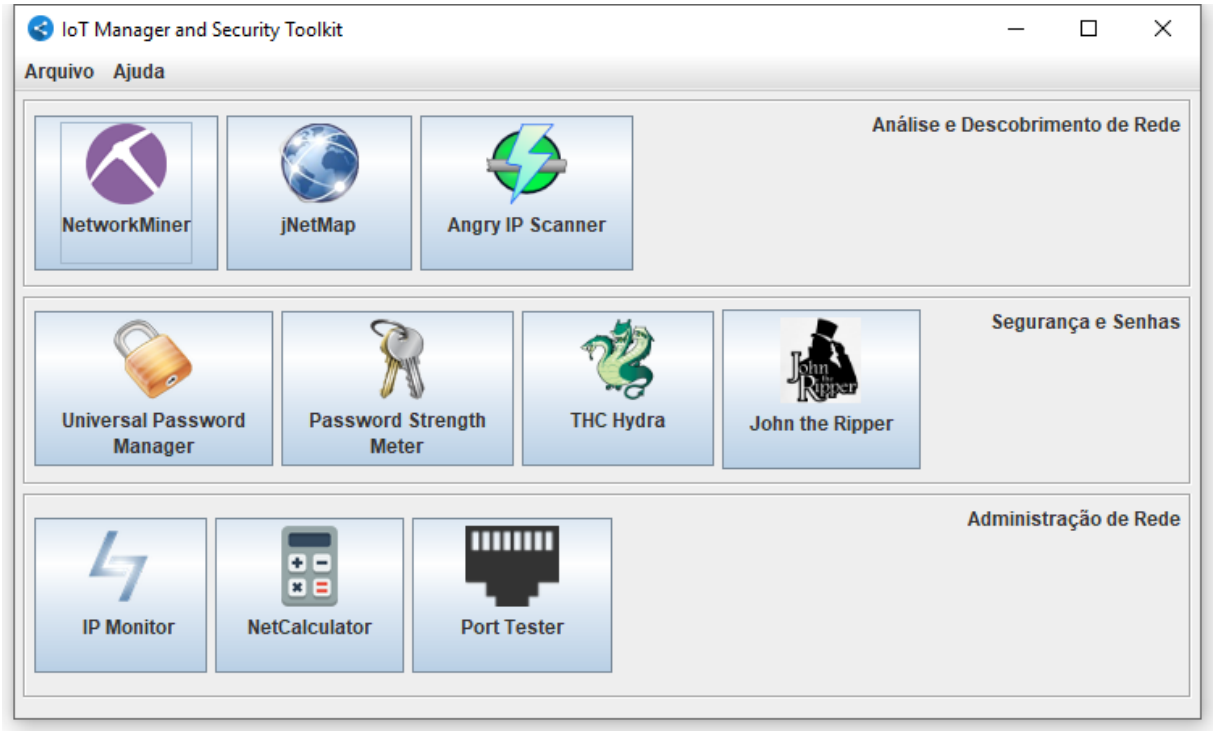
O que é o IoT Manager and Security Toolkit?

O IoT Manager and Security Toolkit é um software de integração de ferramentas open-source para administração e segurança de redes normais e BYOD que visa auxiliar as tarefas diárias do profissional de TI. Redes BYOD são redes corporativas nas quais os funcionários podem conectar dispositivos pessoais, como um notebook, por exemplo, para realizar tarefas profissionais.

O IoT Manager and Security Toolkit é uma suíte de ferramentas formada por um conjunto de vários outros softwares que foram alterados para funcionarem de maneira conjunta. Estes softwares foram divididos em categorias de acordo com a sua respectiva área de atuação.

Interface Principal

A interface principal do software é responsável por controlar e permitir o acesso às ferramentas integradas. O usuário[12] pode utilizar os menus ou os botões de atalho para abrir as ferramentas.



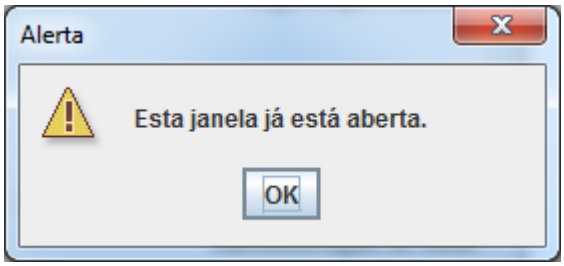
Interface Principal.

[12] Em todos os momentos em que este manual se refere ao termo “usuário” é com o intuito de identificar o usuário da ferramenta e não o usuário da rede corporativa. O usuário desta suíte possivelmente será um Administrador de Redes.

A interface é responsiva e funciona em diversas resoluções de tela. A ferramenta inicializa em um tamanho pré-fixado. Ao clicar na opção “Maximizar” a interface vai automaticamente se ajustar à resolução utilizada pelo usuário.

Detalhe do botão Maximizar.

Para garantir o correto funcionamento, o protótipo não permite que uma ferramenta seja aberta mais de uma vez, caso ela já esteja aberta.

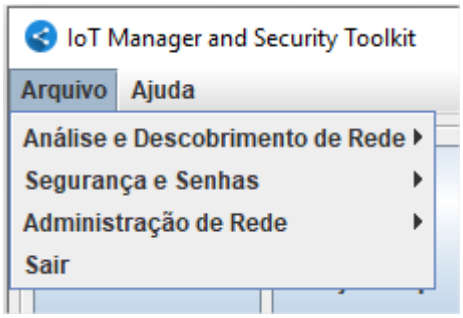


Caso o usuário tente abrir duas vezes a mesma ferramenta.

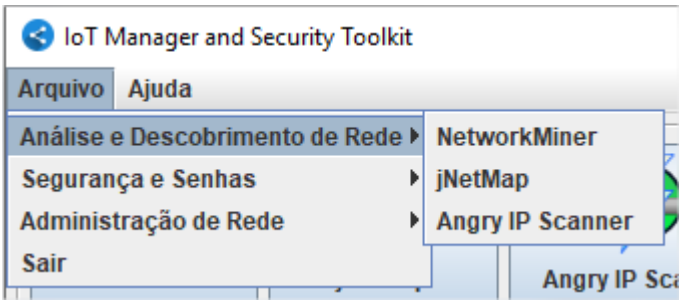
Além disso, o protótipo permite a inicialização de apenas uma ferramenta por vez. No momento em que o usuário clicar em um menu/botão, a ferramenta será carregada. Durante um curtíssimo intervalo de tempo os demais menus/botões ficarão desabilitados. Este processo garante o correto funcionamento das ferramentas.

Menu Arquivo

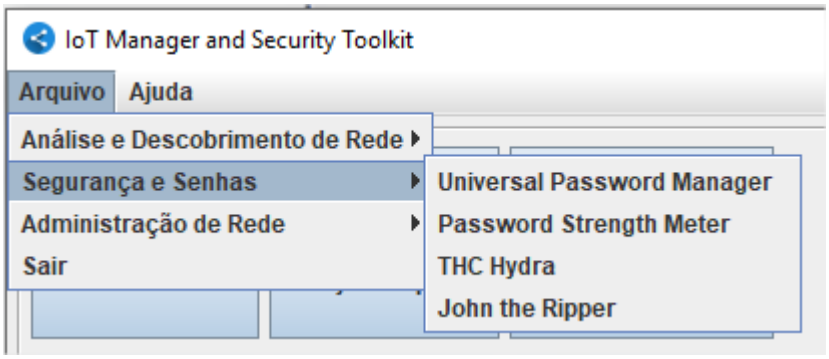
O menu Arquivo é responsável por disponibilizar o acesso às ferramentas que foram integradas de acordo com sua área de atuação. Na área “Análise e Descobrimento de Rede” é possível acessar as ferramentas: NetworkMiner, jNetMap e Angry IP Scanner. Na área “Segurança e Senhas” é possível abrir o Universal Password Manager, Password Strength Meter, THC Hydra e o John the Ripper. Na área “Administração de Rede” é possível abrir o IP Monitor, NetCalculator e o Port Tester. O menu Arquivo ainda possui a opção “Sair” que fecha o programa.



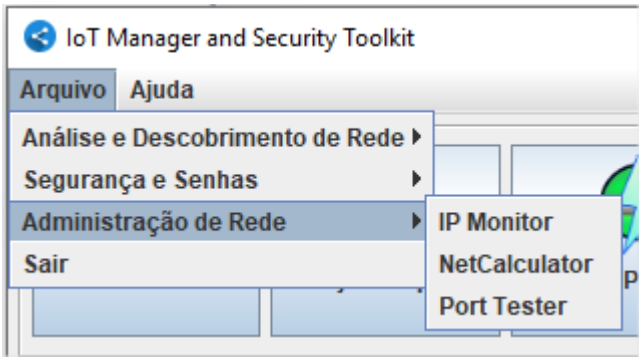
Menu Arquivo.



Menu Arquivo >> Análise e Descobrimento de Rede.



Menu Arquivo >> Segurança e Senhas.

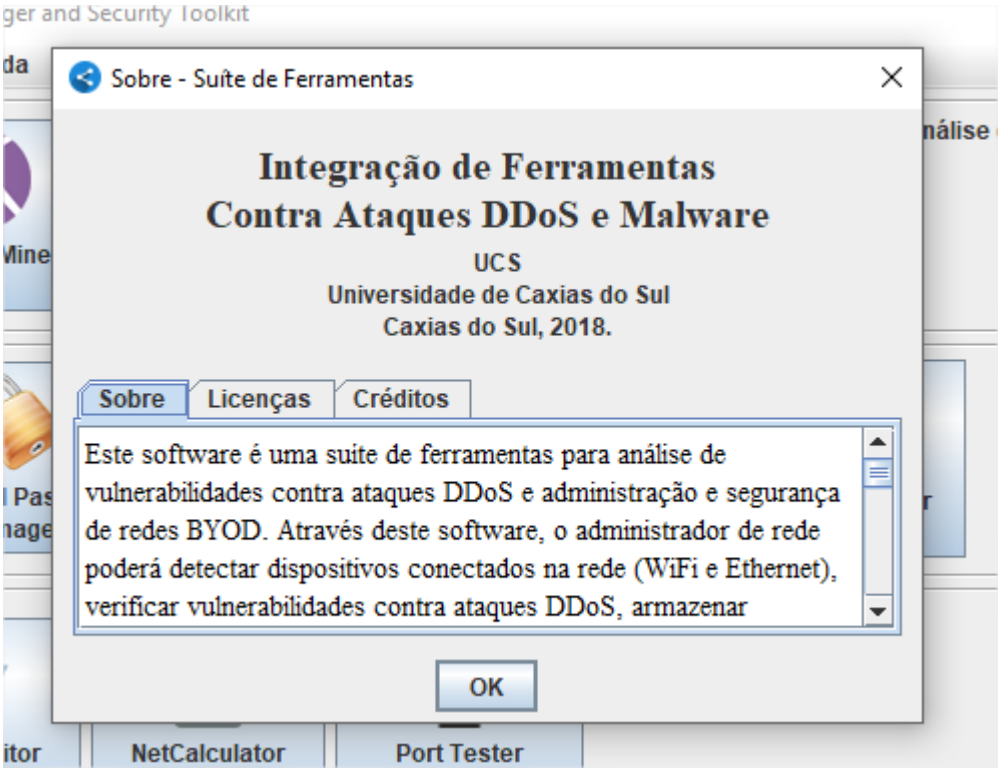


Menu Arquivo >> Administração de Rede.

Menu Ajuda

O menu Ajuda é responsável por permitir que o usuário acesse a documentação oficial de cada ferramenta com apenas um clique. A ferramenta NetworkMiner e a ferramenta jNetMap possuem manuais de usuário em arquivo PDF. As outras ferramentas disponibilizam sua documentação no site oficial. Além disso o menu Ajuda possui uma aba “Sobre” que abre um diálogo com informações adicionais sobre o projeto de integraão.

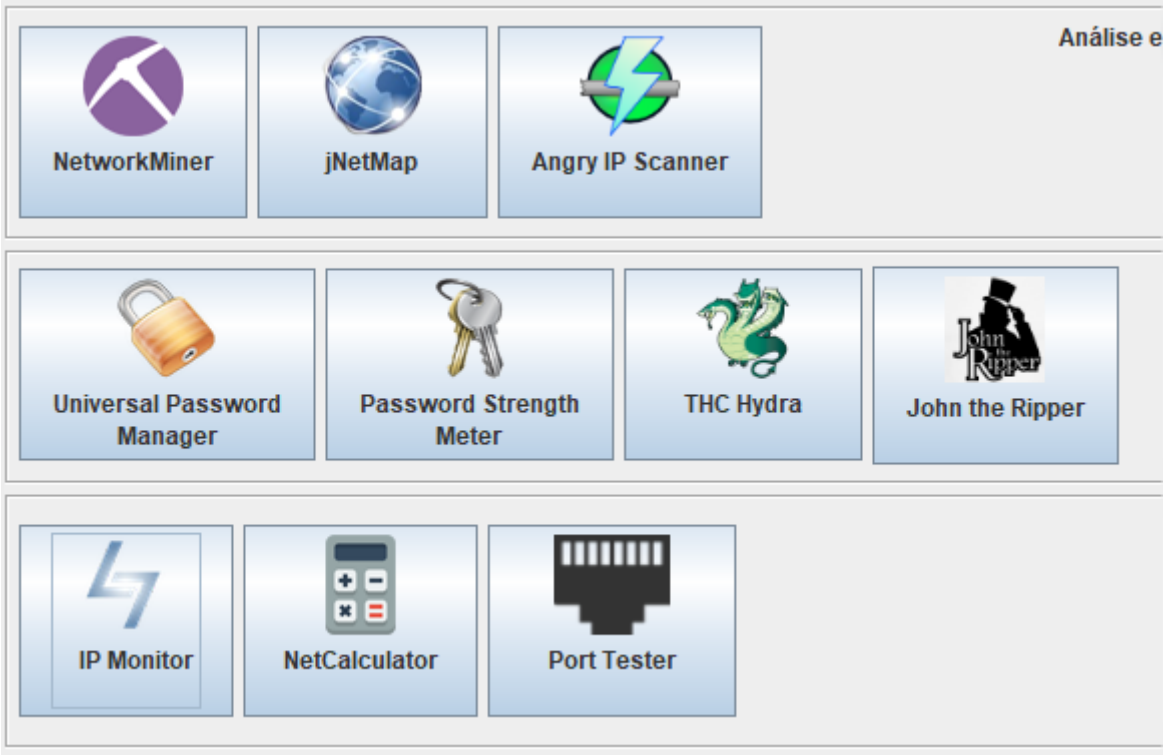
Menu Ajuda >> Documentação.



Diálogo Sobre.

Botões de Atalho

Ainda na interface principal do IoT Manager and Security Toolkit, existem os botões de atalho. Esta é a forma mais rápida de acessar as ferramentas que foram integradas.



Botões de Atalho.

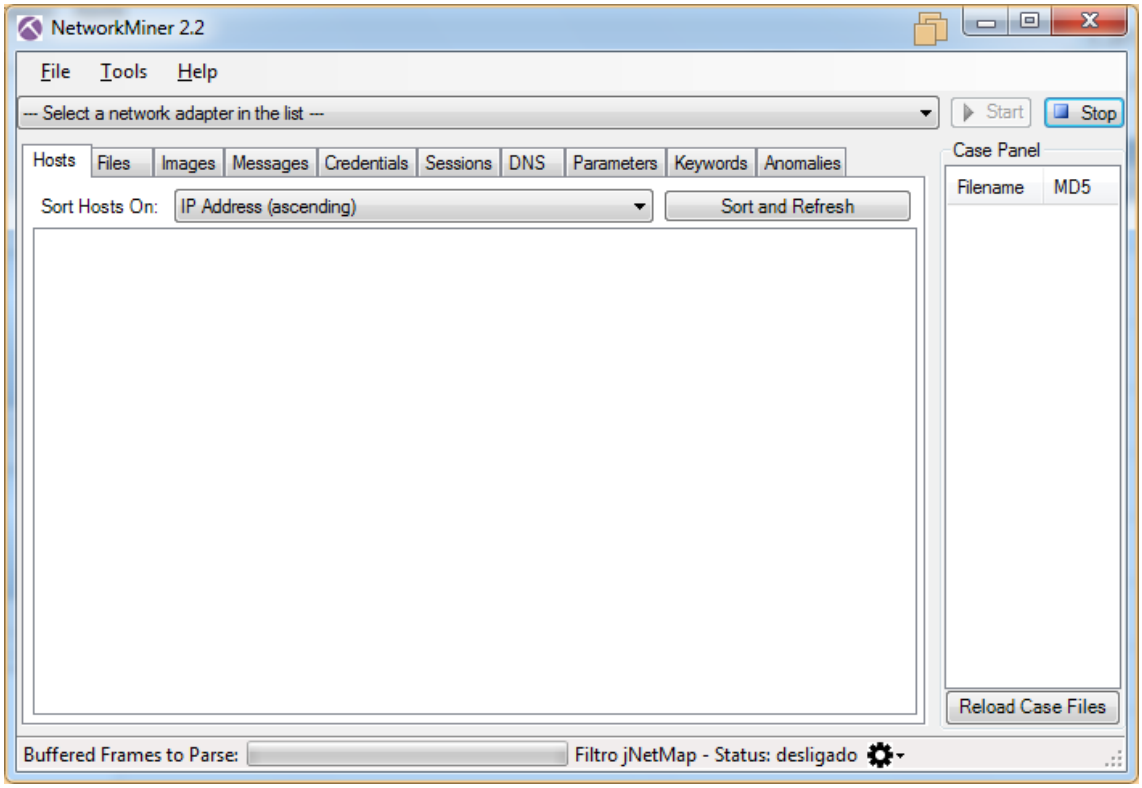
Análise e Descobrimento de Rede

A categoria Análise e Descobrimento de Rede engloba três ferramentas: o NetworkMiner, jNetMap e o Angry IP Scanner. O funcionamento delas é descrito a seguir. Após, as funcionalidades de integração são detalhadas.

NetworkMiner

O NetworkMiner é um sniffer de rede passivo, open-source e desenvolvido em C#. Ele pode ser acessado tanto pelo Menu Arquivo quanto pelo botão “NetworkMiner”.

Em detalhe as formas de acessar o NetworkMiner.

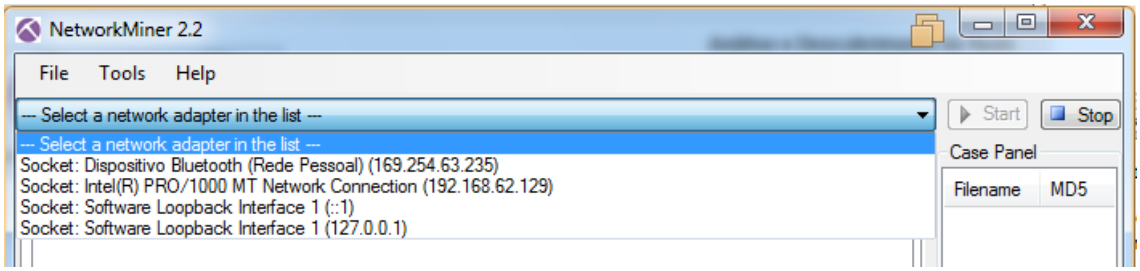


Interface principal do NetworkMiner integrado.

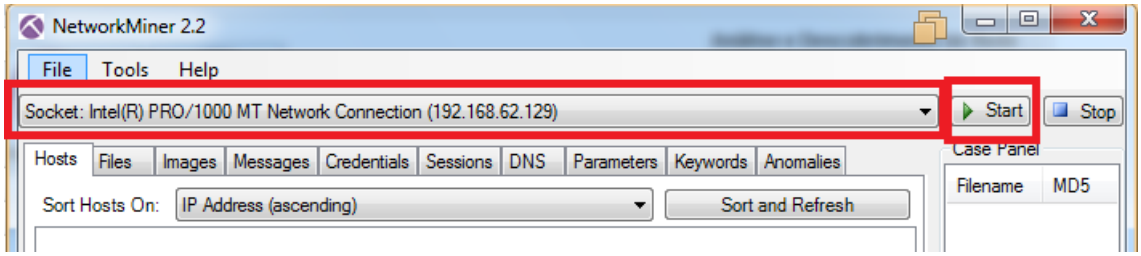
Para utilizar o NetworkMiner é necessário:

- Escolher um adaptador de rede dentre os disponíveis na lista (passo 1).
- Clicar no botão “Start” (passo 2).
- O software vai listar os resultados na tela, em suas diferentes abas.
- Caso haja o desejo de interromper o escaneamento de rede, clicar em “Stop” (passo 3).

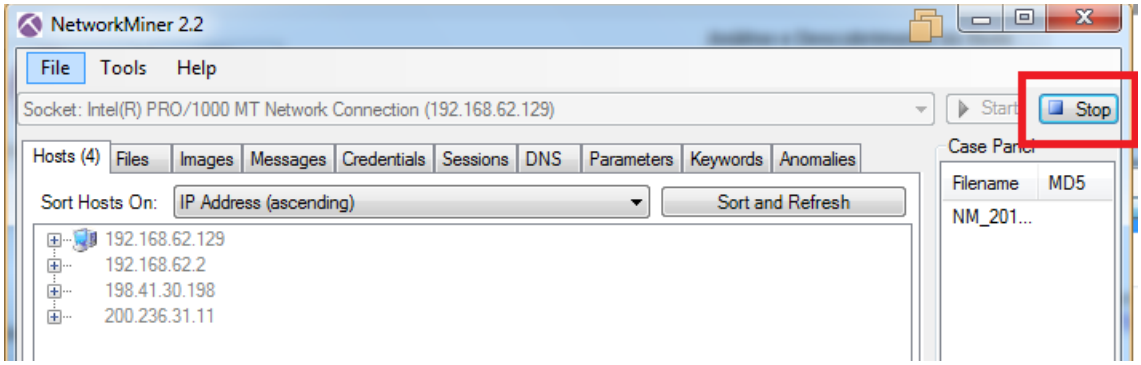
Maiores informações estão disponíveis na documentação oficial, que pode ser acessada no menu Ajuda na interface principal do IoT Manager and Security Toolkit.



Passo 1.



Passo 2.



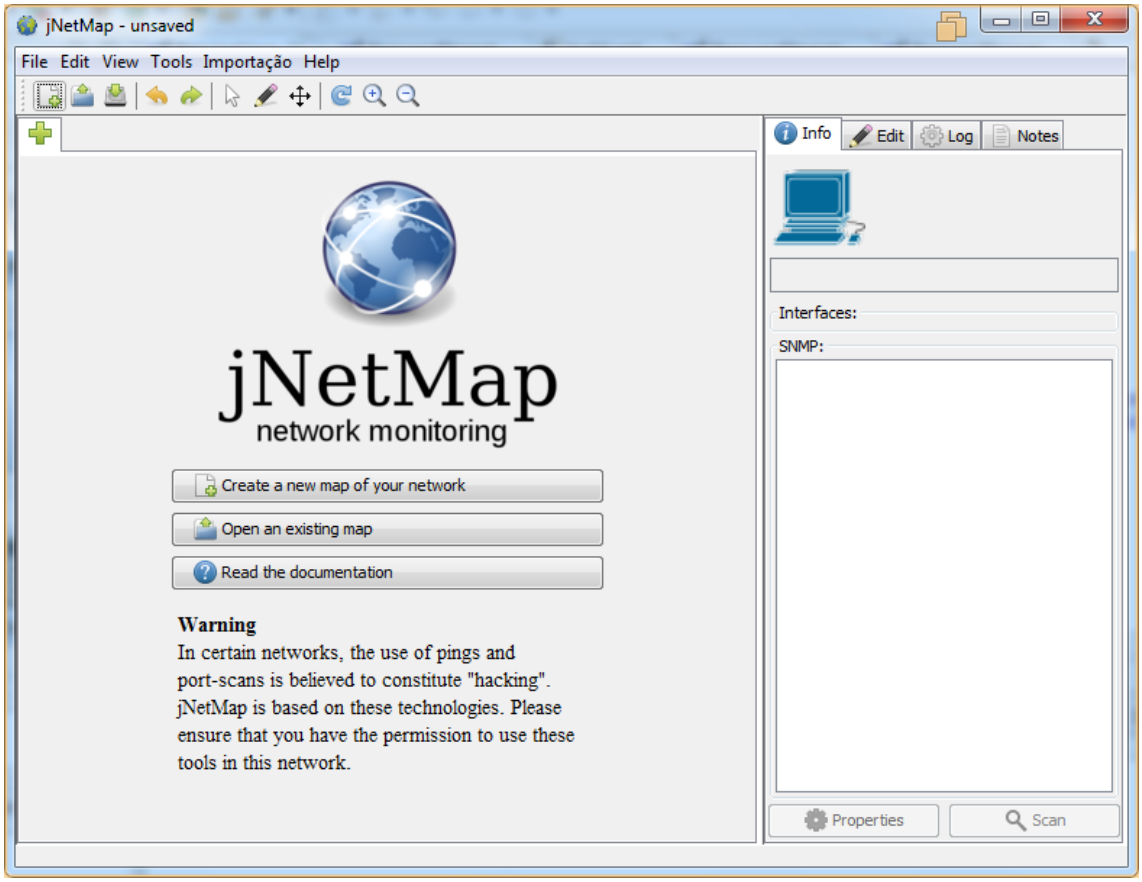
Passo 3.

---

jNetMap

O jNetMap é uma ferramenta gráfica para monitoramento e documentação de redes, open-source e desenvolvido em Java. Ela pode ser acessada tanto pelo Menu Arquivo quanto pelo botão “jNetMap”.

Em detalhe as formas de acessar o jNetMap.

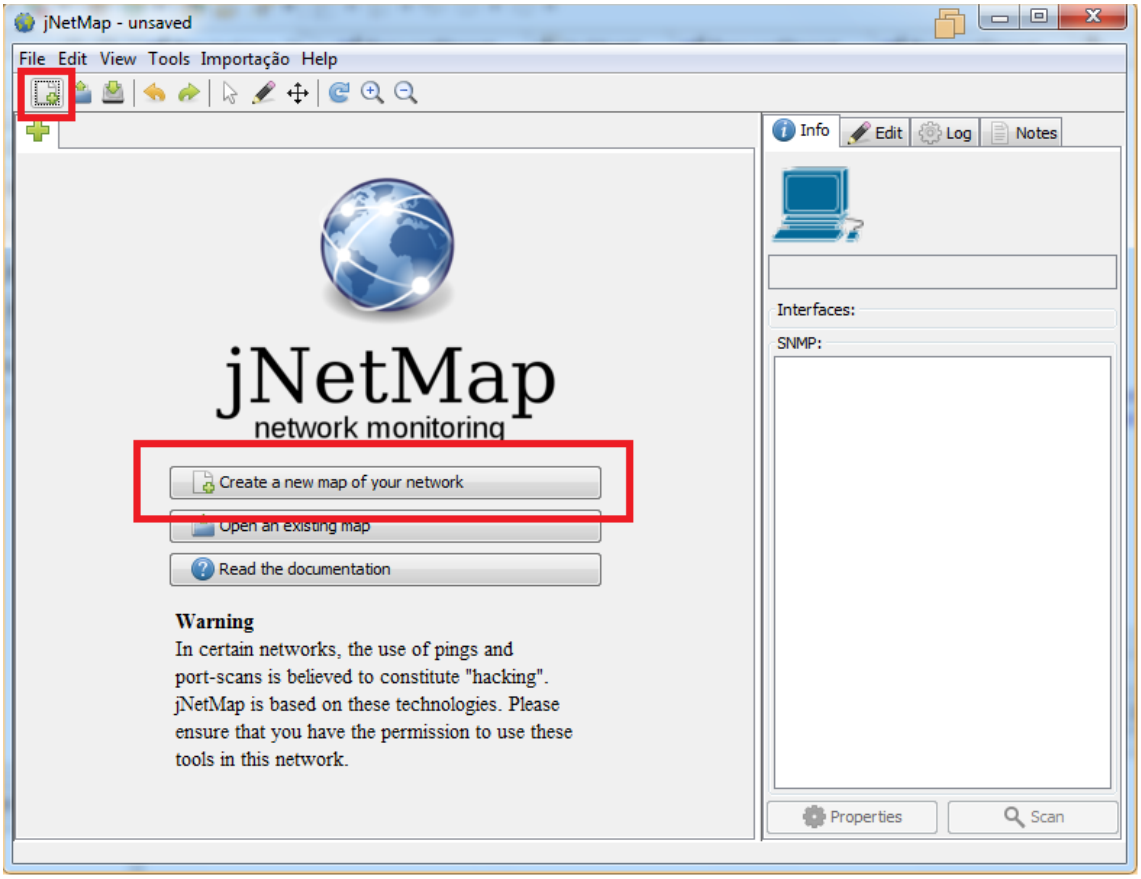


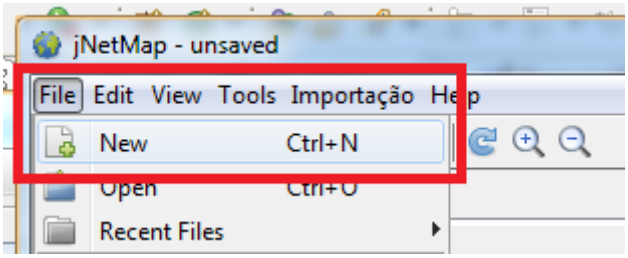
Interface principal do jNetMap integrado.

O jNetMap permite criar um novo mapa, abrir um mapa existente, adicionar manualmente dispositivos ou buscá-los na rede.

Para adicionar um novo mapa é necessário:

- Clicar no ícone novo mapa ou no botão “Create a new map of your network” ou através do menu “Arquivo” (passo 1).

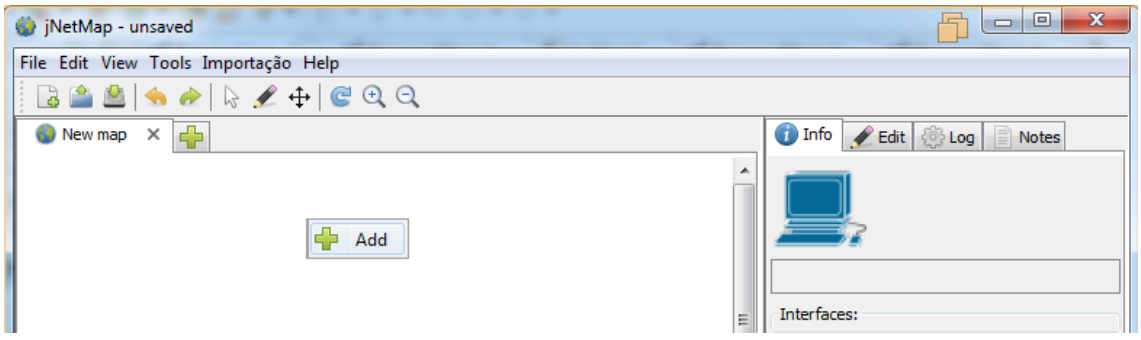




Como criar um novo mapa (passo 1).

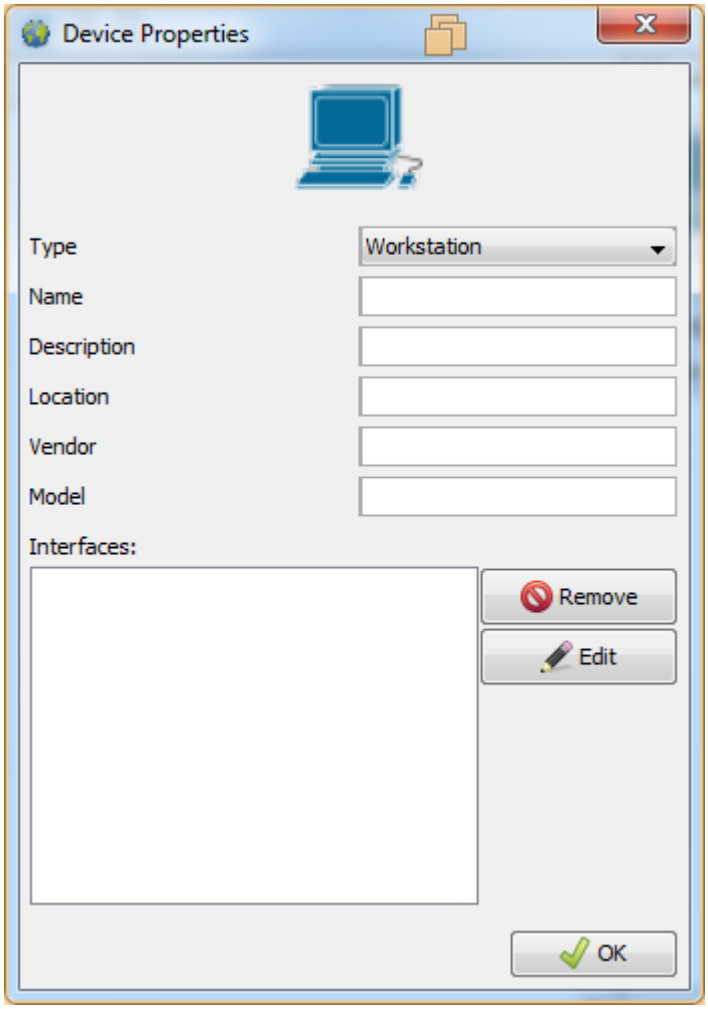
Para adicionar dispositivos manualmente:

- Clique com o botão direito do mouse em uma área em branco do mapa e então clicar no botão “Add” (passo 2).



Adicionar dispositivo (passo 2).

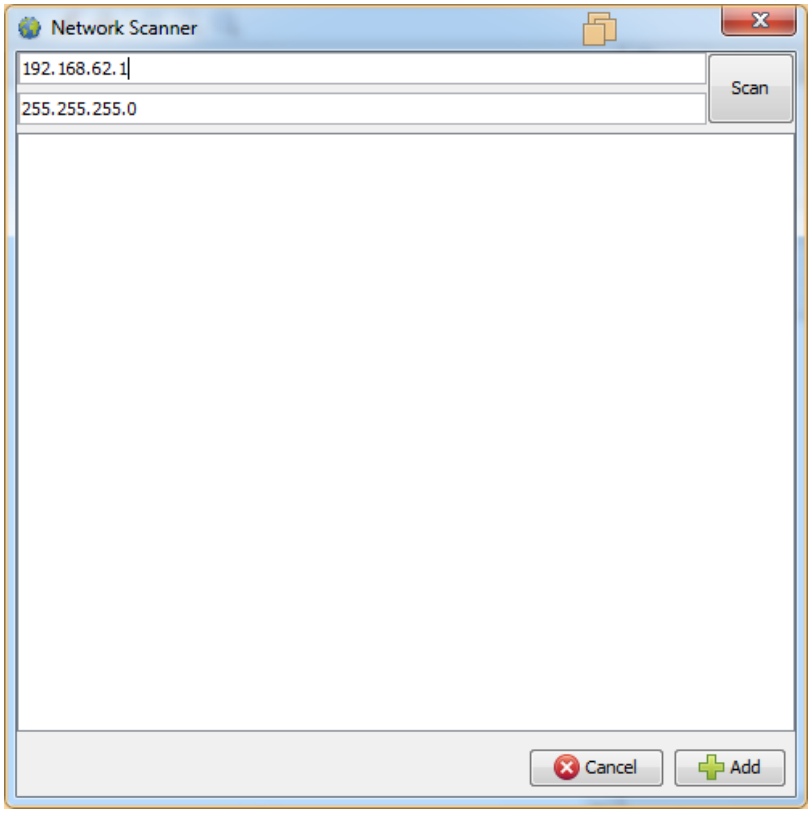
- Complete os campos de acordo com o dispositivo escolhido. Após clicar no botão “OK”.



Informando os dados do dispositivo (passo 2).

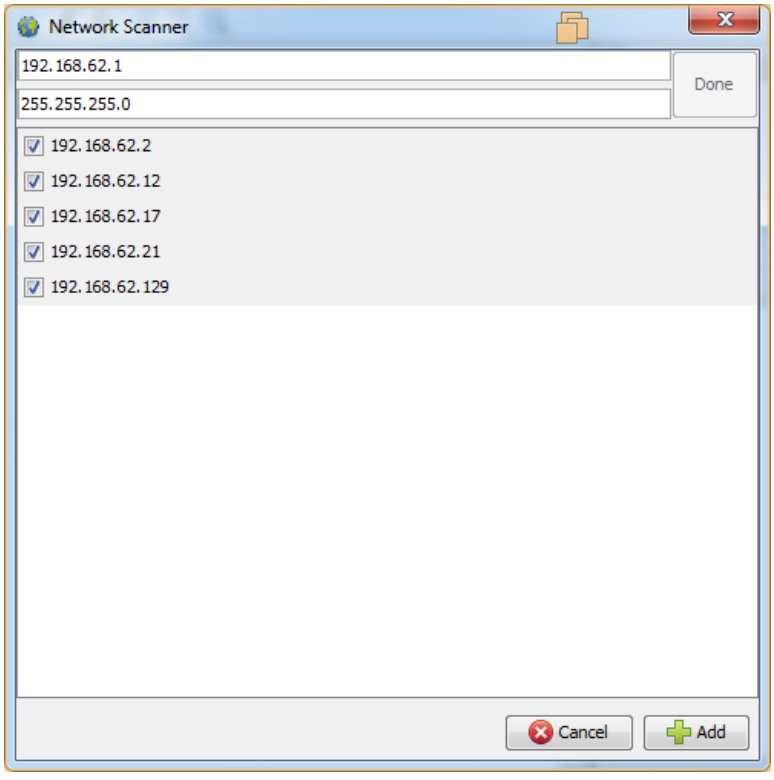
Para adicionar dispositivos de forma automática:

- Clique no menu “Tools” (Ferramentas) na interface principal do jNetMap.
- Clique no botão “Network Scanner”, que abrirá uma nova janela.
- Informe qual a faixa de IP deseja escanear e qual a máscara de rede.

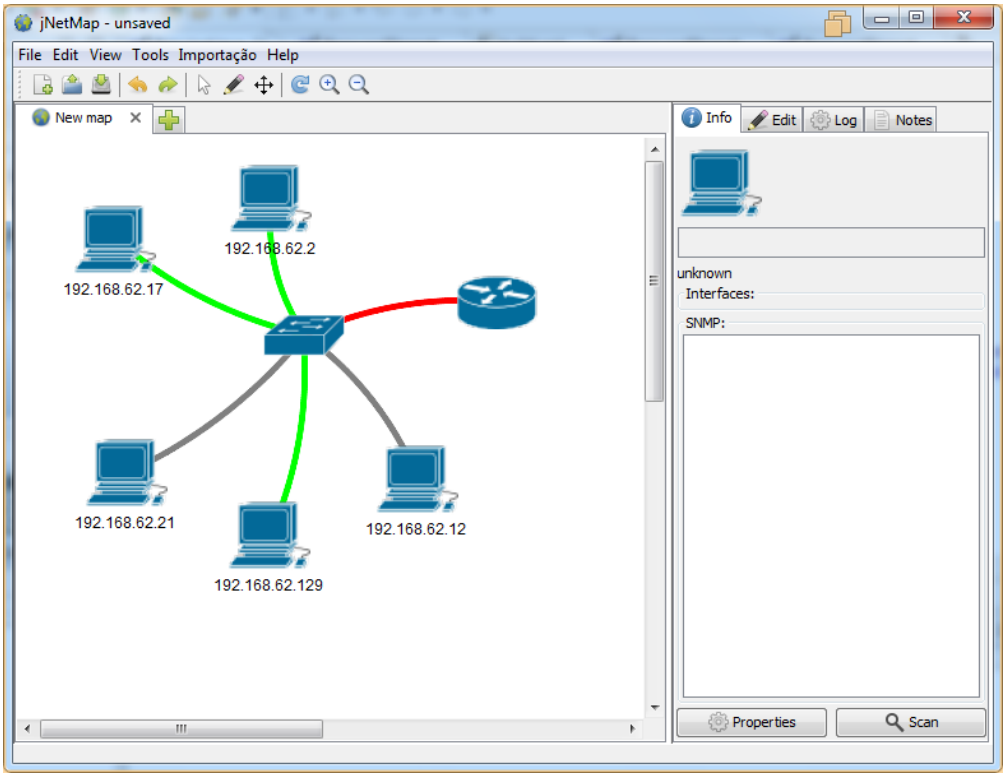


Escaneamento automático de rede (passo 3).

Após o escaneamento ser concluído, o usuário poderá escolher quais dispositivos ele deseja adicionar ao mapa. Basta selecioná-los na lista e clicar no botão “Add”.



Escaneamento concluído.



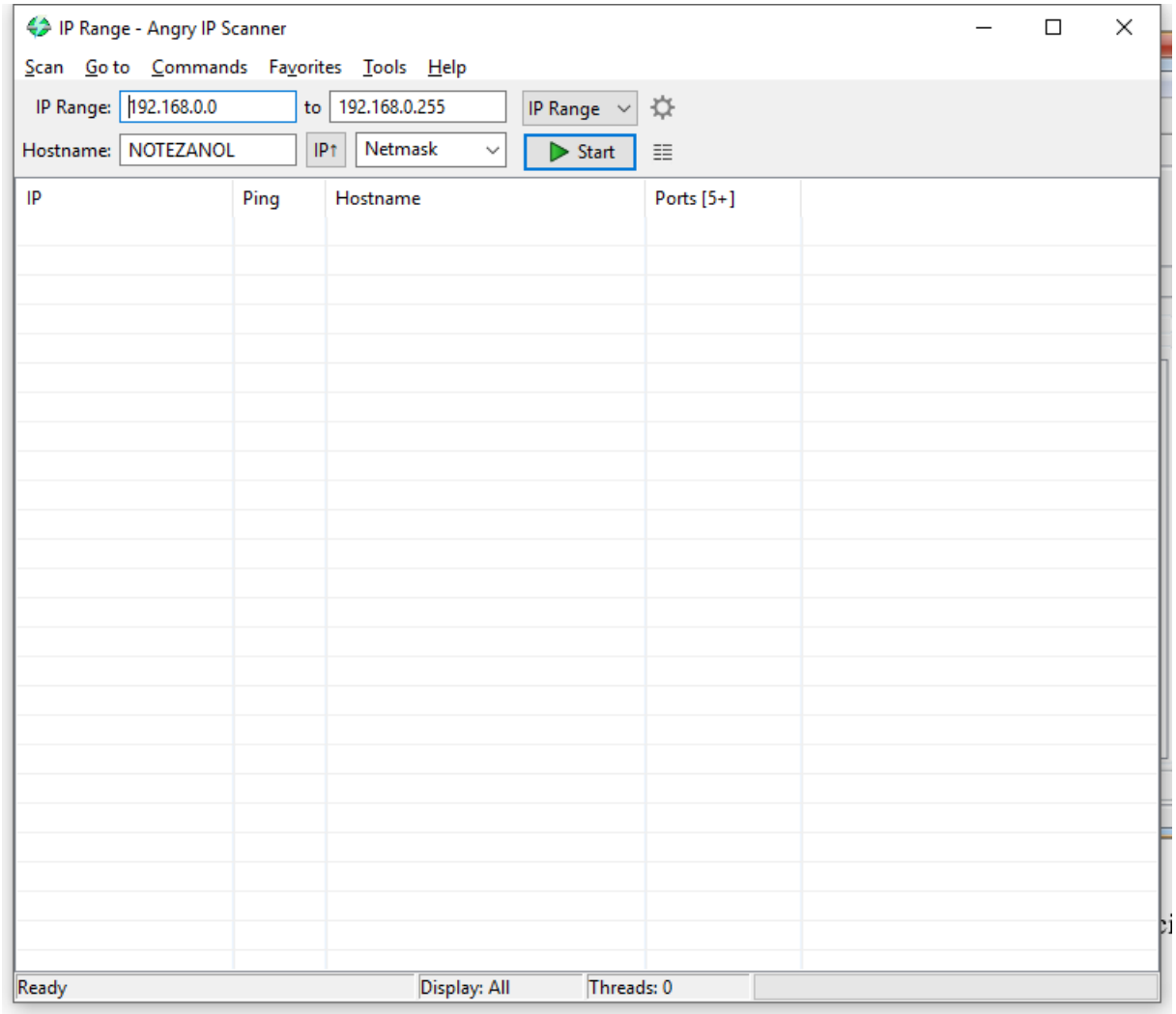
Dispositivos inseridos no mapa.

Maiores informações estão disponíveis na documentação oficial (que pode ser acessada no menu Ajuda).

Angry IP Scanner

O Angry IP Scanner tem como objetivo escanear IPs e portas em dispositivos conectados em uma rede local, open-source e desenvolvido em Java. Ela pode ser acessada tanto pelo Menu Arquivo quanto pelo botão “Angry IP Scanner”.

Em detalhe as formas de acessar o Angry IP Scanner.

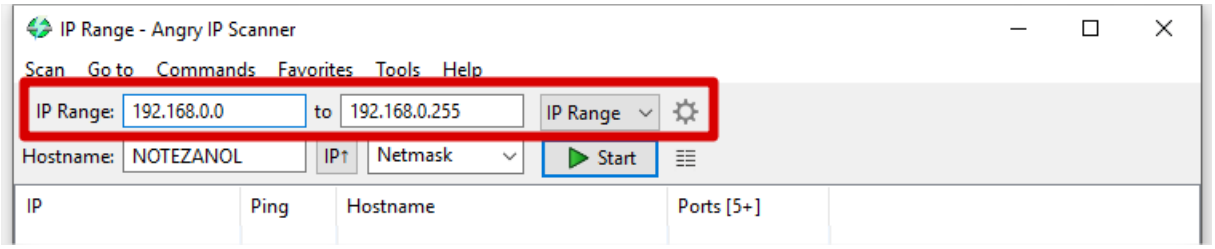


Interface principal do Angry IP Scanner integrado.

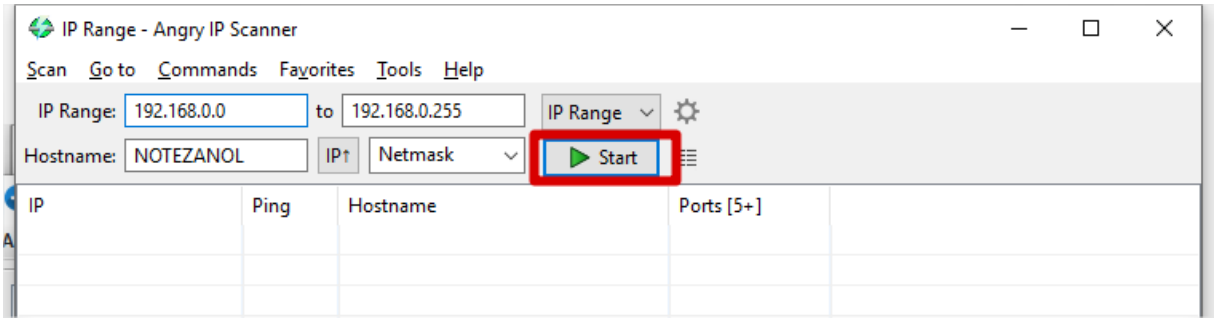
O Angry IP Scanner permite verificar todos dispositivos conectados na rede para a faixa de IPs informada nos campos “IP Range”.

Para utilizar o software é necessário:

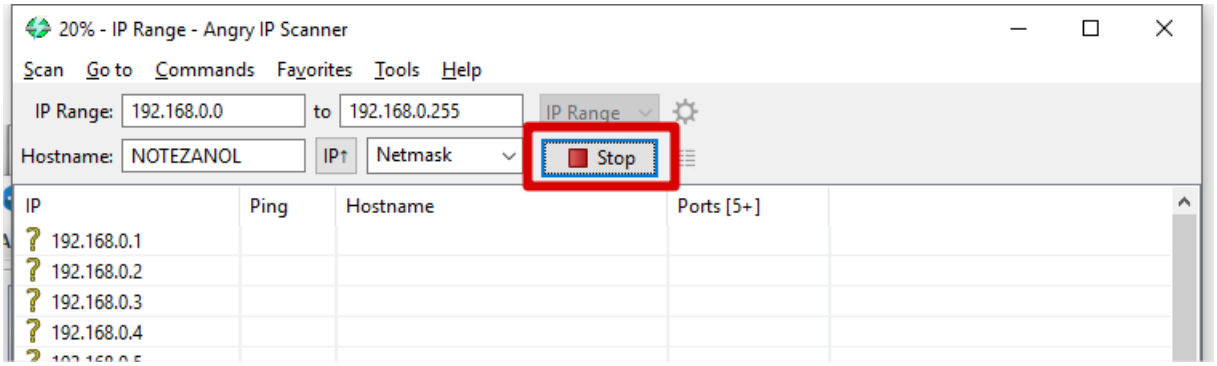
- Modificar a faixa de IPs automaticamente carregada caso necessário (passo 1).
- Clicar no botão “Start” (passo 2).
- O software vai listar os resultados na tela.
- Caso haja o desejo de interromper o escaneamento de rede, clicar em “Stop” (passo 3).



Passo 1.



Passo 2.



Passo 3.

Funcionalidades de Integração

Os tópicos descritos a seguir são funcionalidades que foram adicionadas às ferramentas originais.

Importar para o jNetMap

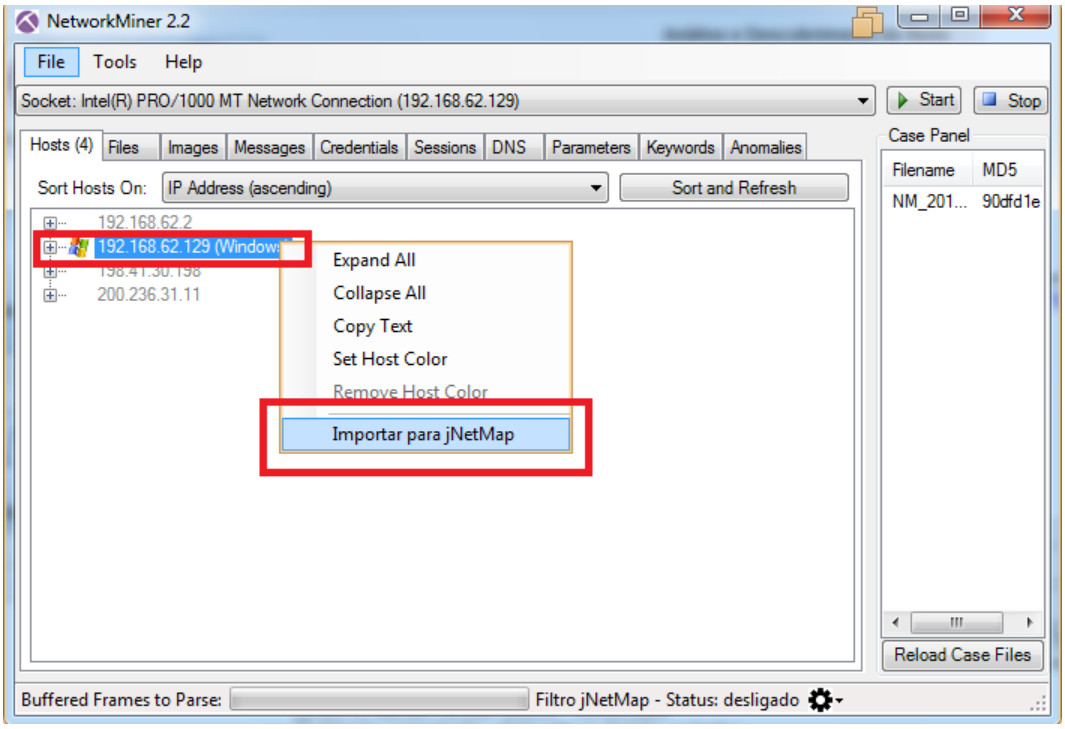
O NetworkMiner é capaz de reunir algumas informações sobre os hosts que estão na rede. Entre elas é possível destacar o endereço IP e o sistema operacional do host. A importação de hosts para o jNetMap é uma funcionalidade que envia algumas destas informações do host selecionado para o jNetMap.

Para realizar a importação, basta seguir estes passos:

- Clicar com o botão direto em um host da lista (passo 1)

- Clicar na opção “Importar para jNetMap” (passo 2).
  Neste momento, o jNetMap vai receber as informações enviadas do NetworkMiner.

- Para finalizar a importação é necessário utilizar a ferramenta jNetMap. O usuário deverá escolher em qual mapa deseja importar os dados.



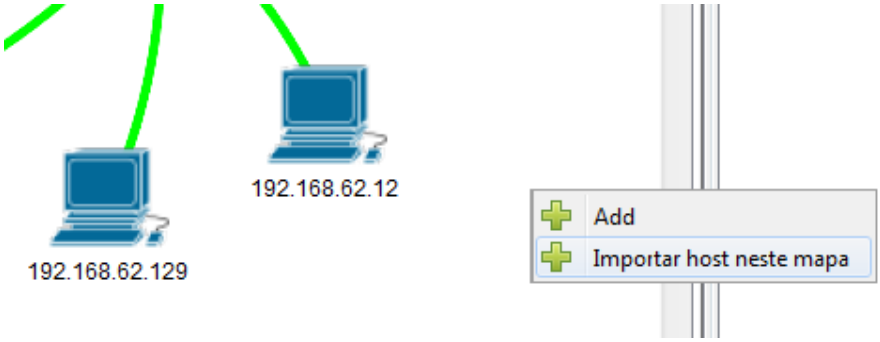
Passo 1 e Passo 2.

Importar host no jNetMap

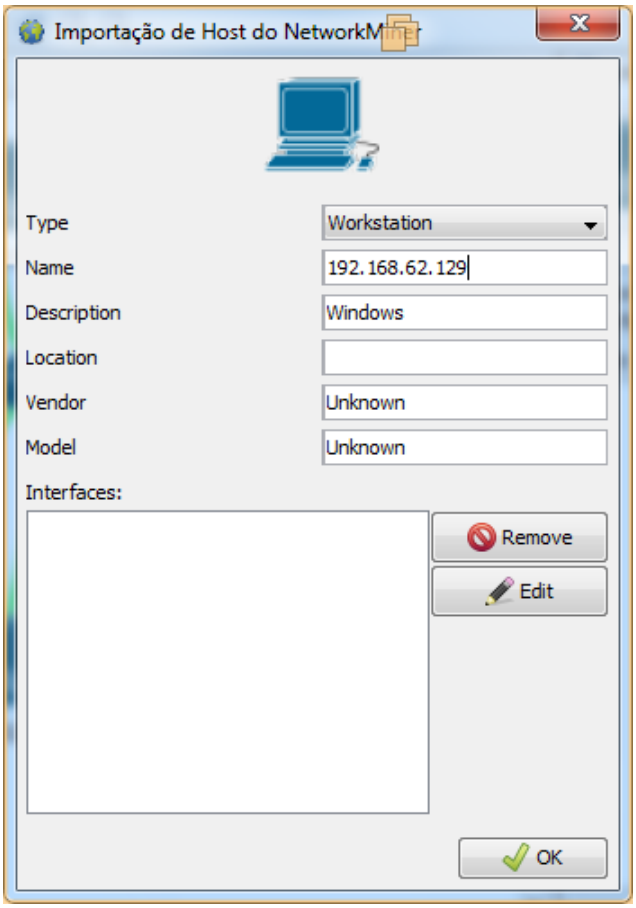
Para importar um dispositivo no jNetMap é preciso previamente enviá-lo do NetworkMiner.

Uma vez que o tutorial “Importar para o jNetMap” (pág 15) tenha sido realizado, os hosts recebidos poderão ser importados para qualquer mapa de rede do jNetMap. O usuário deverá:

- Clicar com o botão direito em qualquer área branca do mapa e escolher a opção “Importar host neste mapa” (passo 1).
- Confirmar e/ou editar as informações recebidas e então confirmar a adição do dispositivo no mapa (passo 2).

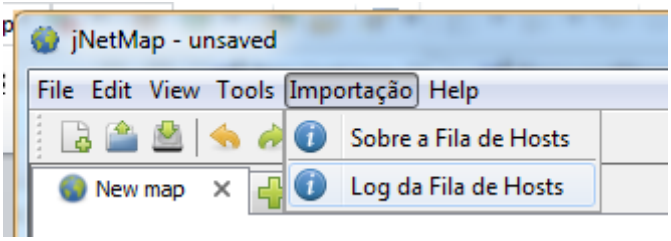


Passo 1.



Passo 2.

O usuário ainda pode utilizar o menu “Importação” para saber mais detalhes sobre este sistema de importação.



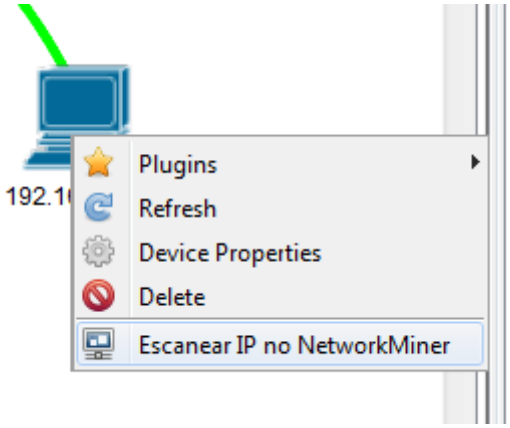
Menu Importação.

Escanear IP no NetworkMiner

O usuário pode enviar um endereço IP para ser escaneado pelo NetworkMiner. Este IP servirá como filtro dos resultados.

Para enviar um endereço IP para o NetworkMiner é necessário:

- Clicar com o botão direito em qualquer dispositivo do mapa (passo 1).
- Clicar na opção “Escanear IP no NetworkMiner”(passo 2).



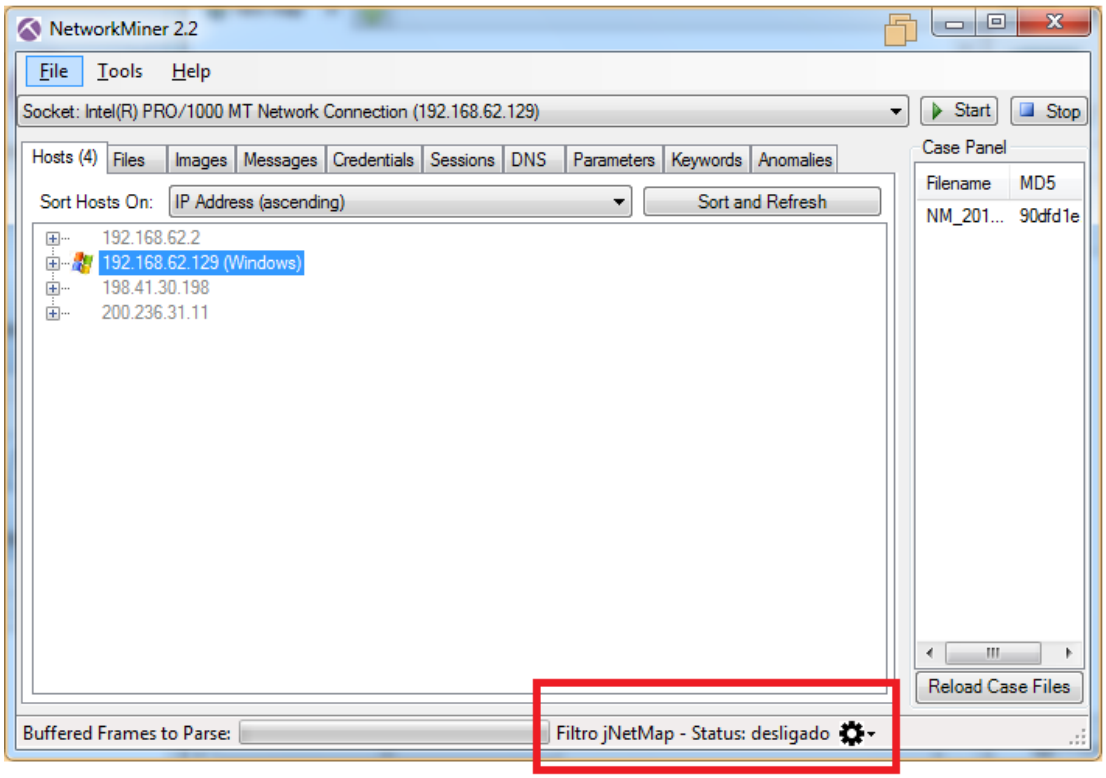
Passo 1 e passo 2.

Automaticamente, o endereço será recebido pelo NetworkMiner e o filtro será ligado.

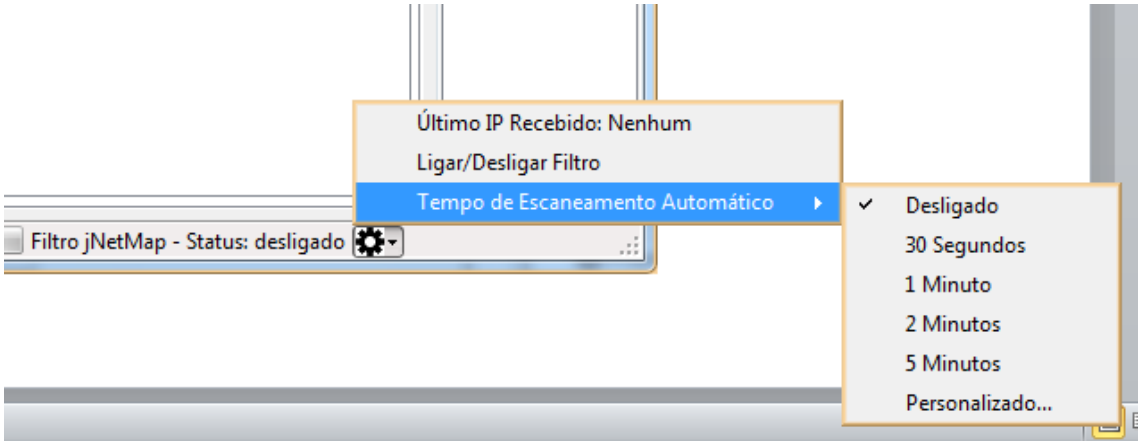
Filtragem de Resultados

A filtragem de resultados é consequência do recebimento de um endereço IP previamente enviado pelo jNetMap. O filtro é automaticamente ativado quando um endereço IP é recebido. Através do menu “Filtro jNetMap” é possível verificar qual IP está sendo utilizado no filtro, ligar/desligar o filtro e definir um intervalo de tempo para o escaneamento automático.

Com o filtro ligado, todos os resultados mostrados na tela serão relacionados com o endereço IP selecionado. Caso o filtro esteja desligado, o escaneamento mostrará todos os resultados obtidos normalmente.



Posição do menu Filtro jNetMap.



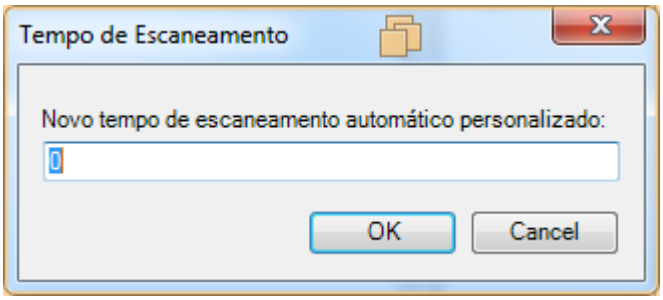
Menu Filtro jNetMap.

Escaneamento Automático

O escaneamento automático remove a necessidade do usuário clicar nos botões “Start” e “Stop” para realizar o escaneamento com filtro ao receber um endereço IP do jNetMap.

O escaneamento automático é configurado por padrão como “0 Segundos”. Ou seja, desligado. O usuário pode selecionar entre intervalos pré-definidos ou inserir um intervalo

personalizado. Caso queira desativar, basta inserir um intervalo personalizado de 0 (zero) segundos ou clicar na opção “Desligado” dentro do menu “Filtro jNetMap”.



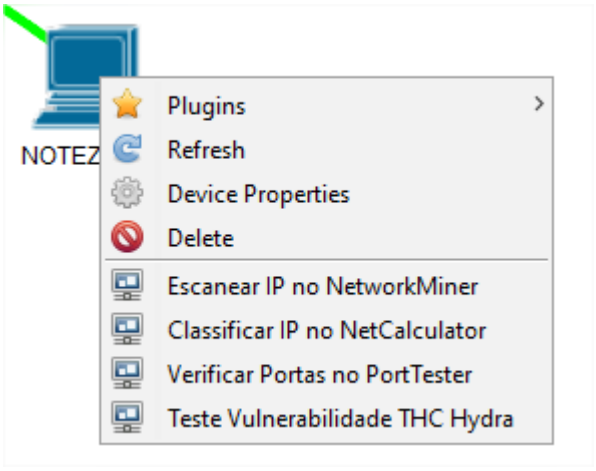
Intervalo personalizado de tempo.

Escanear IP no Port Tester no jNetMap

O usuário pode enviar um endereço IP para ser escaneado pelo Port Tester.

Para enviar um endereço IP para o Port Tester é necessário:

- Clicar com o botão direito em qualquer dispositivo do mapa (passo 1).
- Clicar na opção “Verificar Portas no PortTester”(passo 2).



Passo 1 e passo 2.

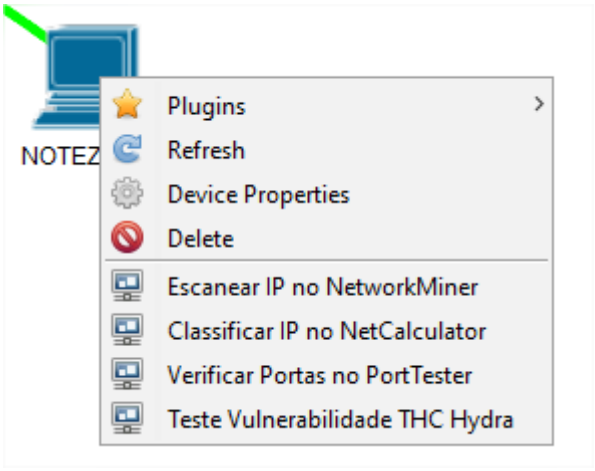
Automaticamente, o endereço será recebido pelo Port Tester e o campo do endereço do dispositivo será carregado.

Verificar vulnerabilidade para IP no THC Hydra no jNetMap

O usuário pode enviar um endereço IP para ser verificado se o dispositivo possui vulnerabilidades contra ataques DDoS pelo THC Hydra.

Para enviar um endereço IP para o THC Hydra é necessário:

- Clicar com o botão direito em qualquer dispositivo do mapa (passo 1).
- Clicar na opção “Teste Vulnerabilidade THC Hydra”(passo 2).



Passo 1 e passo 2.

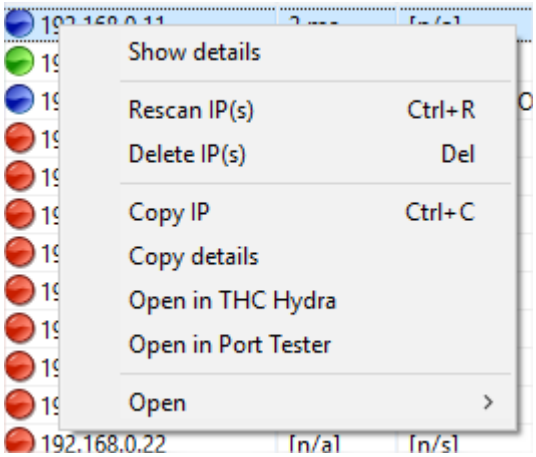
As mesmas opções foram disponibilizadas no Menu >> Commands. Automaticamente, o endereço será recebido pelo THC Hydra e o campo do endereço do dispositivo será carregado.

Escanear IP no Port Tester no Angry IP Scanner

O usuário pode enviar um endereço IP para ser escaneado pelo Port Tester.

Para enviar um endereço IP para o Port Tester é necessário:

- Clicar com o botão direito em qualquer dispositivo listado (passo 1).
- Clicar na opção “Open in Port Tester”(passo 2).



Passo 1 e passo 2.

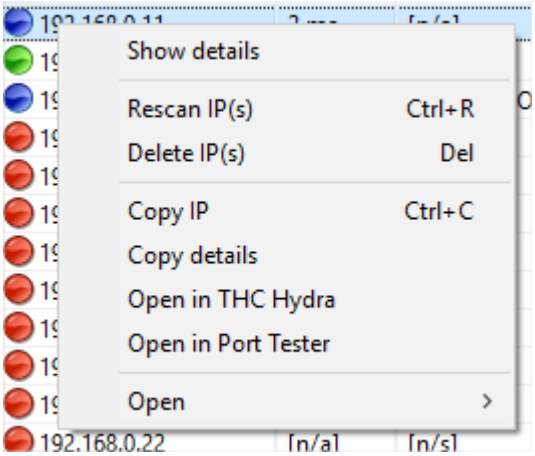
Automaticamente, o endereço será recebido pelo Port Tester e o campo do endereço do dispositivo será carregado.

Verificar vulnerabilidade para IP no THC Hydra no Angry IP Scanner

O usuário pode enviar um endereço IP para ser verificado se o dispositivo possui vulnerabilidades contra ataques DDoS pelo THC Hydra.

Para enviar um endereço IP para o THC Hydra é necessário:

- Clicar com o botão direito em qualquer dispositivo listado (passo 1).
- Clicar na opção “Open in THC Hydra”(passo 2).



Passo 1 e passo 2.

As mesmas opções foram disponibilizadas no Menu >> Commands.

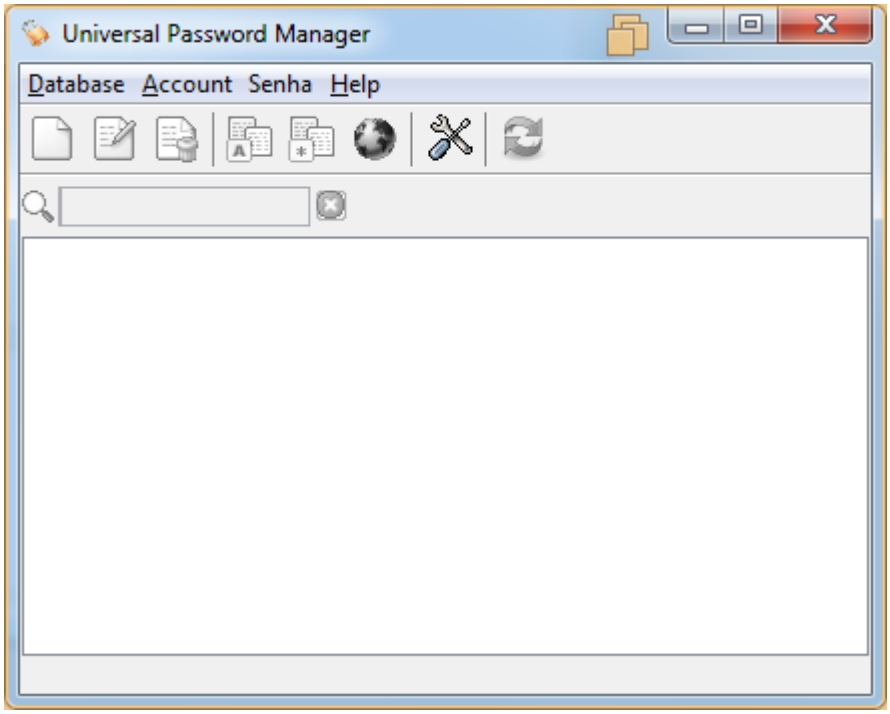
Automaticamente, o endereço será recebido pelo THC Hydra e o campo do endereço do dispositivo será carregado.

Segurança e Senhas

A categoria Segurança e Senhas engloba duas ferramentas: o Universal Password Manager, Password Strength Meter, THC Hydra e o John the Ripper. O funcionamento e as funcionalidades de integração dessas ferramentas são detalhados abaixo.

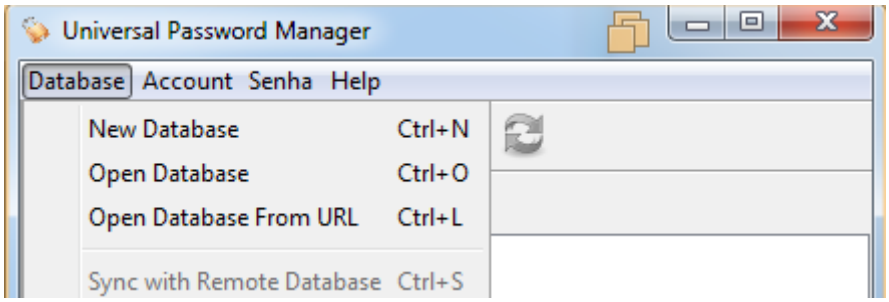
Universal Password Manager

O Universal Password Manager é uma ferramenta para gerenciamento de usuários e senhas, desenvolvida em Java. Ele pode ser acessado pelo menu “Arquivo” ou pelo botão “Universal Password Manager”.



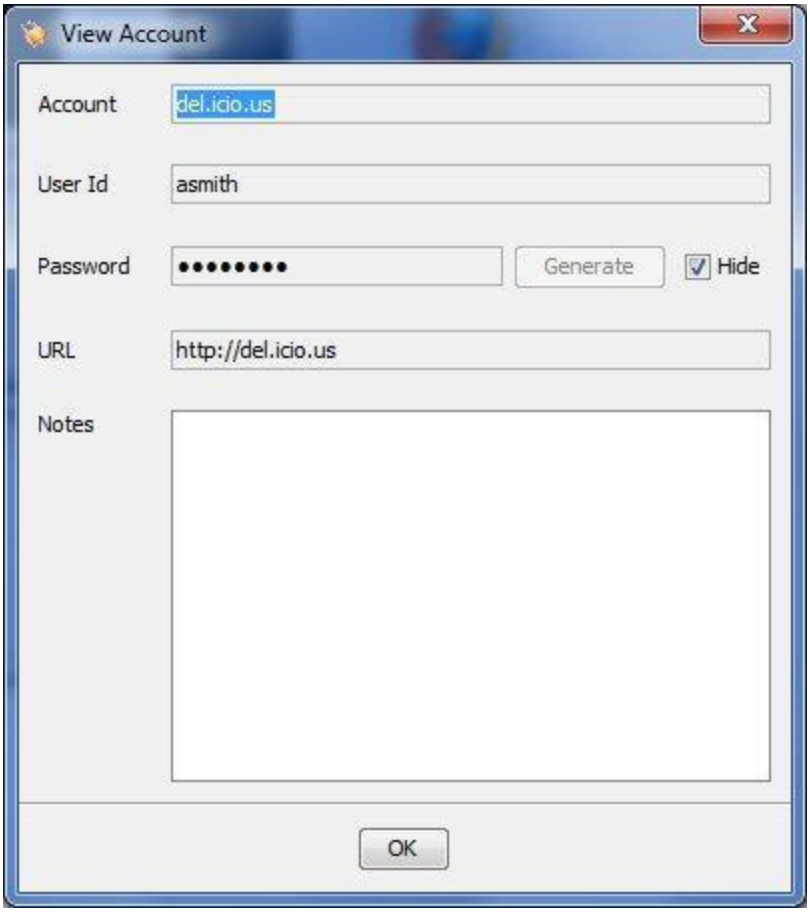
Interface principal do Universal Password Manager Integrado.

Para armazenar e gerenciar as credenciais, é necessário ter uma base de dados. O usuário pode criar uma base nova, ou abrir uma já existente utilizando o menu “Database”.



Menu Database.

No Menu “Account” é possível adicionar novas credenciais de usuário e ver detalhes das já existentes (opção “View Account”). Na interface “View Account” é possível ver detalhes das credenciais, como por exemplo, o usuário, a senha e o endereço de acesso.



Interface “View Account”.

Maiores informações sobre o Universal Password Manager estão disponíveis na documentação oficial, disponível no menu Ajuda da interface principal do protótipo.

Password Strength Meter

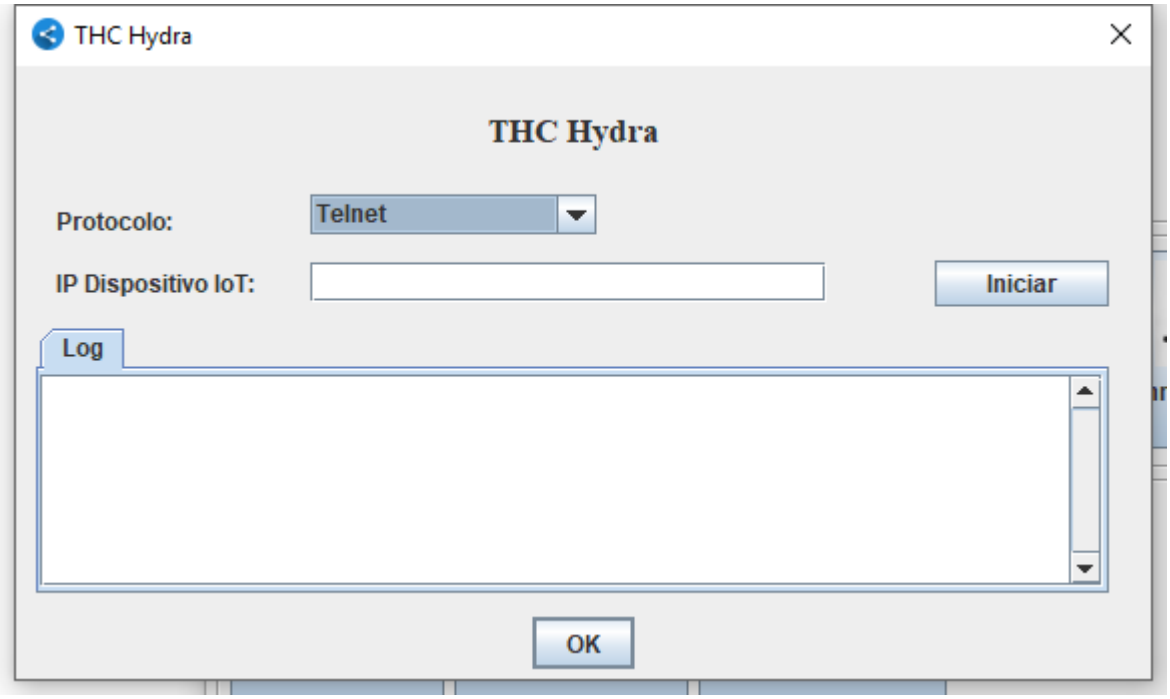
O Password Strength Meter é uma biblioteca desenvolvida em Java. Essa ferramenta é capaz de testar a força das senhas e ainda disponibiliza um número aproximado de iterações necessárias para a quebra da senha. Essa ferramenta pode ser acessada tanto pelo menu “Arquivo”, quanto pelo botão “Password Strength Meter”.

THC Hydra

O Hydra é uma ferramenta executada em um subsistema Linux previamente configurado na estação. Essa ferramenta é capaz de verificar se o dispositivo selecionado possui vulnerabilidade contra ataques DDoS validando um conjunto de senhas utilizadas pelo *botnet*

Mirai. Essa ferramenta pode ser acessada tanto pelo menu “Arquivo”, quanto pelo botão “THC Hydra”. Ela pode ser acessada tanto pelo Menu Arquivo quanto pelo botão “THC Hydra”.

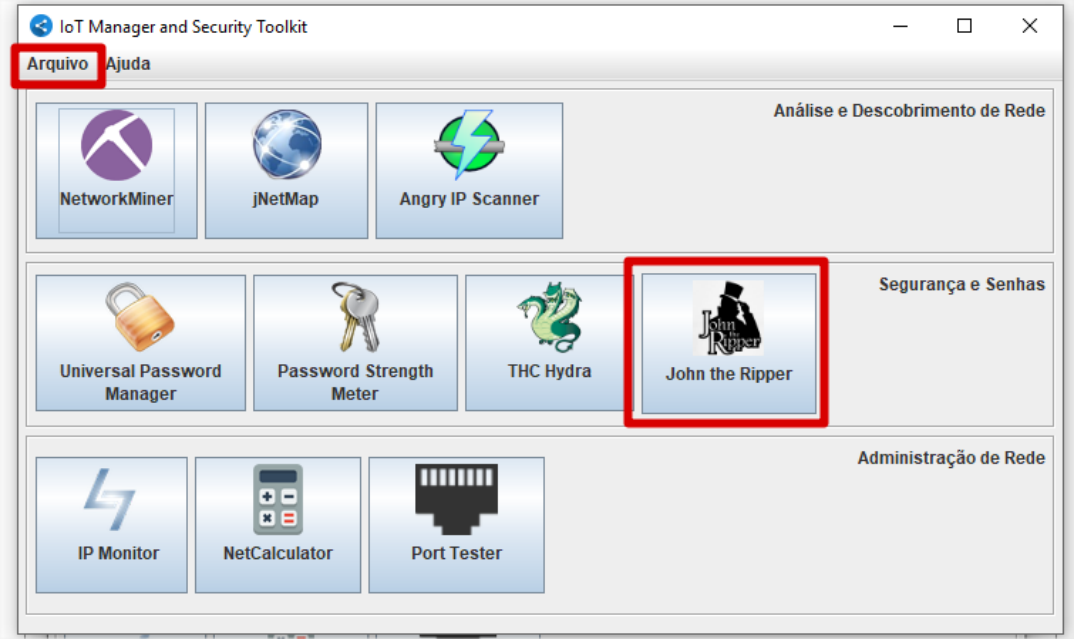
Em detalhe as formas de acessar o THC Hydra.



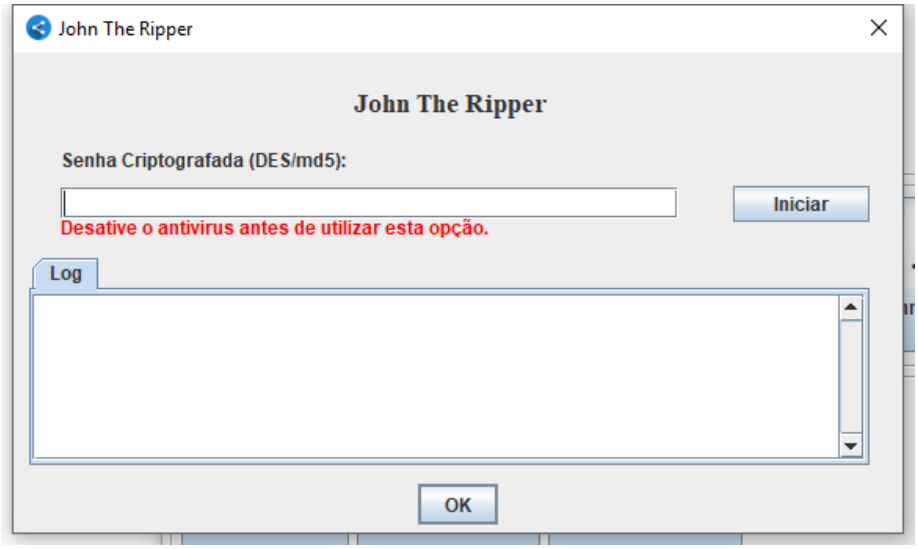
Interface principal do THC Hydra integrado.

John the Ripper

O John the Ripper é uma ferramenta capaz de descriptografar senhas criptogradas no formado md5 e DES. É necessário que *softwares* antivírus estejam desativados para o correto funcionamento. Essa ferramenta pode ser acessada tanto pelo menu “Arquivo”, quanto pelo botão “John the Ripper”. Ela pode ser acessada tanto pelo Menu Arquivo quanto pelo botão “THC Hydra”.



Em detalhe as formas de acessar o John the Ripper.



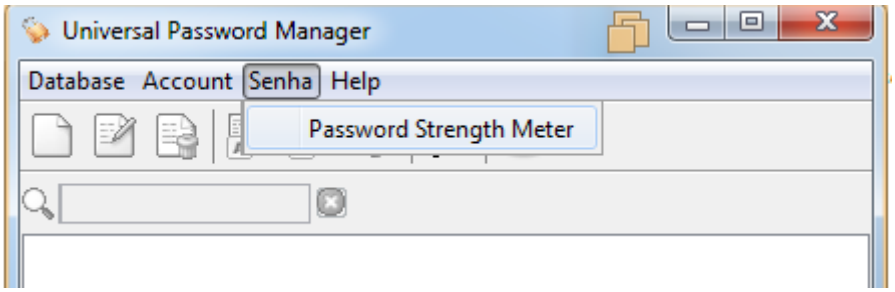
Interface principal do John the Ripper integrado.

Funcionalidades de Integração

Os tópicos descritos a seguir são funcionalidades que foram adicionadas às ferramentas originais.

Testando uma senha com o Universal Password Manager

O usuário do Universal Password Manager poderá testar a força de suas credenciais de rede e/ou outras senhas quaisquer salvas através do menu Senha. Neste menu é possível acessar a ferramenta “Password Strength Meter” e realizar o teste das senhas.



Menu Senha.

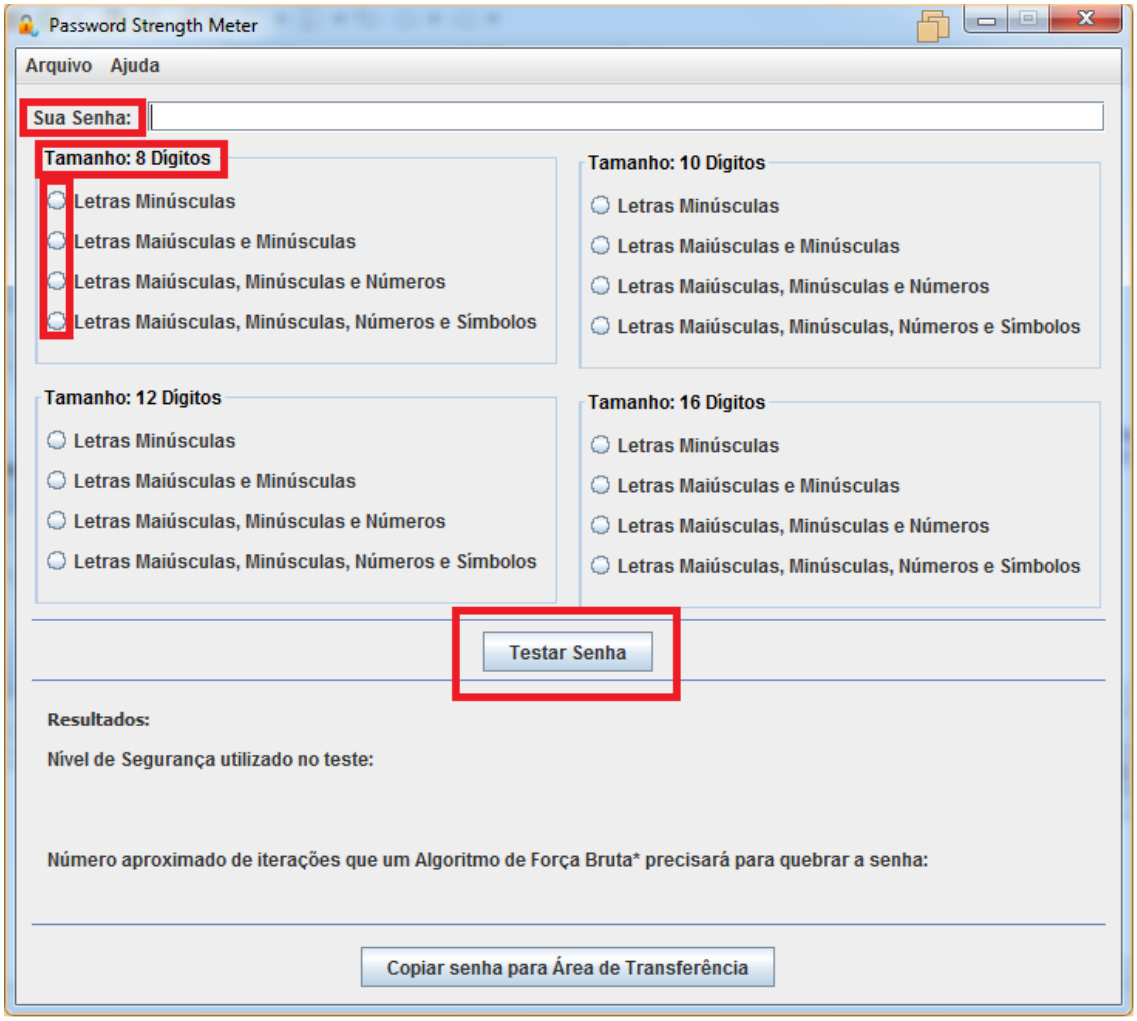
* Observação: Esta pode ser aberta também pelo botão “Password Strength Meter” na interface principal do IoT Manager and Security Toolkit.

As informações de como realizar o teste de força de senha são descritas no tutorial “Testando uma senha com o Password Strength Meter”.

Testando uma senha com o Password Strength Meter

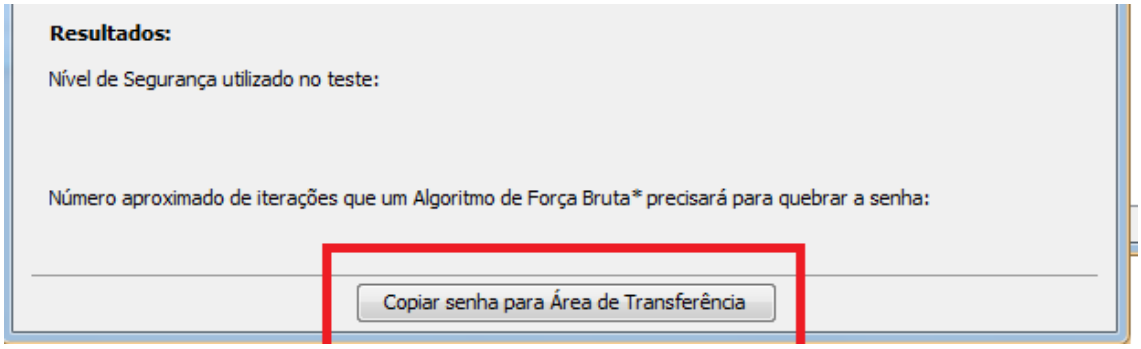
Para testar uma senha é necessário:

- Informar no campo “Sua Senha” a senha que será testada.
- Escolher qual padrão de senha ela deve atender.
- Clicar no botão “Testar Senha”. Os resultados serão exibidos na parte inferior da interface.



Interface principal do Password Strength Meter.

O usuário poderá também copiar a senha para a área de transferência facilmente. Basta clicar no botão “Copiar senha para Área de Transferência”.



Botão “Copiar senha para Área de Transferência”.

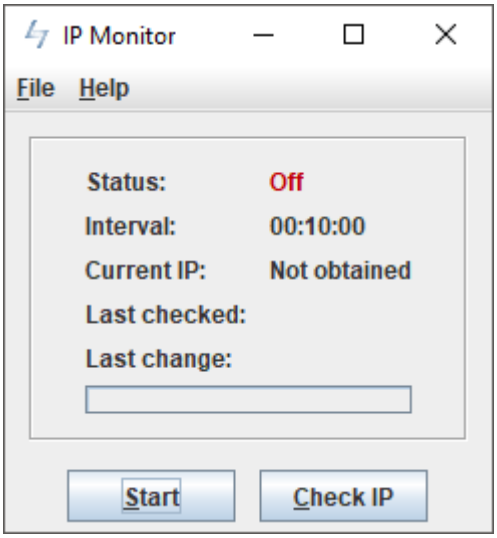
Maiores informações estão disponíveis na documentação oficial do Password Strength Meter.

Administração de Rede

A categoria Administração de Rede engloba duas ferramentas: o IP Monitor, NetCalculator e o Port Tester. O funcionamento e as funcionalidades de integração dessas ferramentas são detalhados abaixo.

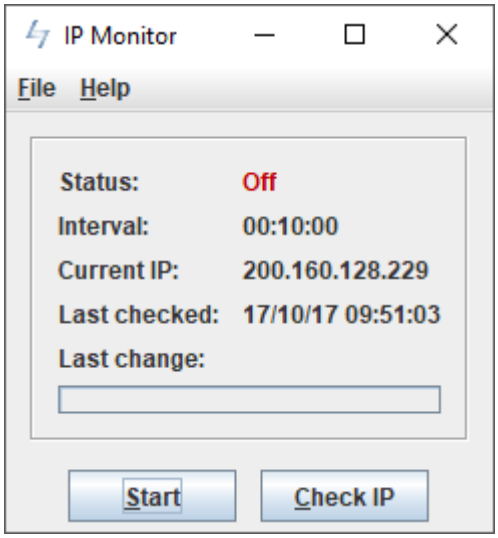
IP Monitor

O IP Monitor é uma ferramenta para monitoramento do endereço de IP público. Ela é desenvolvida em Java e pode ser acessada pelo botão “IP Monitor” ou pelo menu “Arquivo” na interface principal do IoT Manager and Security Toolkit.



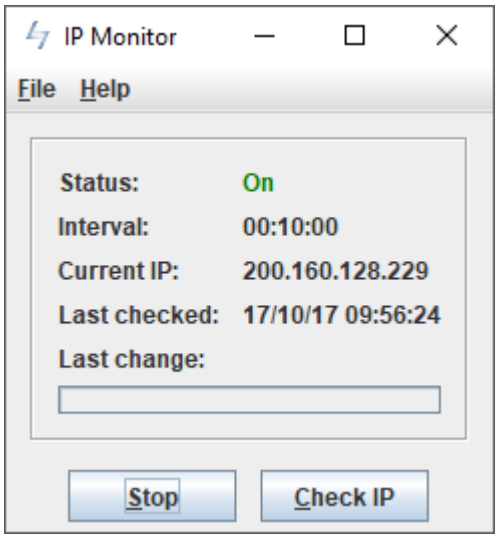
Interface Principal do IP Monitor Integrado.

Para checar o Endereço IP público manualmente, basta clicar no botão “Check IP” na interface principal. O software verificará o endereço e mostrará como resultado o IP atual, o horário da última verificação e o horário da última mudança, caso tenha ocorrido.



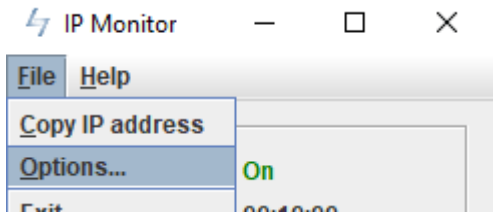
Endereço IP Público verificado às 09:51 do dia 17 de Outubro de 2017.

O usuário poderá iniciar um monitoramento automático do IP público clicando no botão “Start”. Com isso um processo de verificação será inicializado. Este processo consiste em realizar uma verificação periodicamente. Esta verificação informará o IP atual, o horário da última verificação e o horário da última mudança, caso tenha ocorrido.

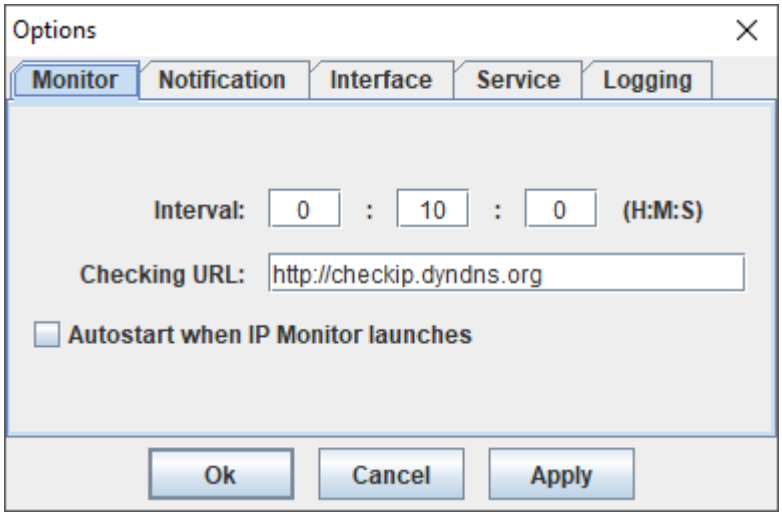


Processo de monitoramento ligado.

O usuário poderá personalizar o intervalo de tempo entre uma verificação e outra, além de definir qual a melhor forma de notificação em caso de mudança de endereço de IP público. Para realizar essa personalização é necessário clicar no menu “File” e então no botão “Options”.



Menu File.



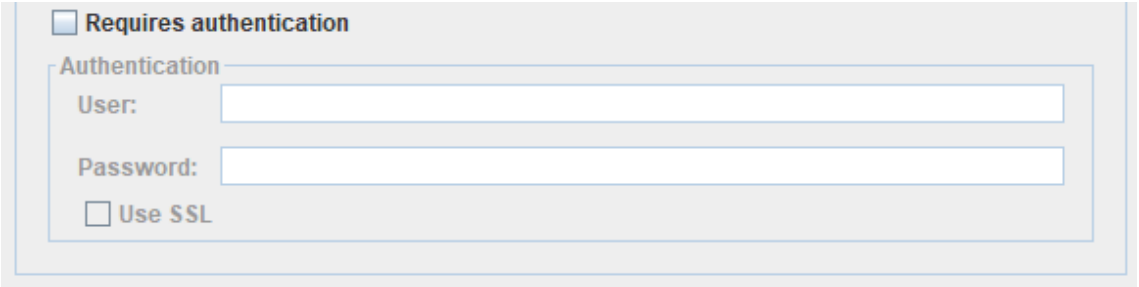
Janela de Opções: Definindo o Intervalo de Escaneamento.

O usuário poderá escolher entre quatro tipos diferentes de notificações. Na aba "Notification" é possível selecionar notificação de áudio, via e-mail, notificação visual ou via linha de comando.

Para efetuar uma notificação por e-mail é necessário:

- Configurar o servidor de e-mail utilizado (passo 1).
- Caso seja necessário, o usuário deve informar usuário e senha (passo 2).
- Completar os campos da Mensagem com Nome, Destinatário, Assunto, etc (passo 3);
- Escrever o corpo da Mensagem (passo 4).

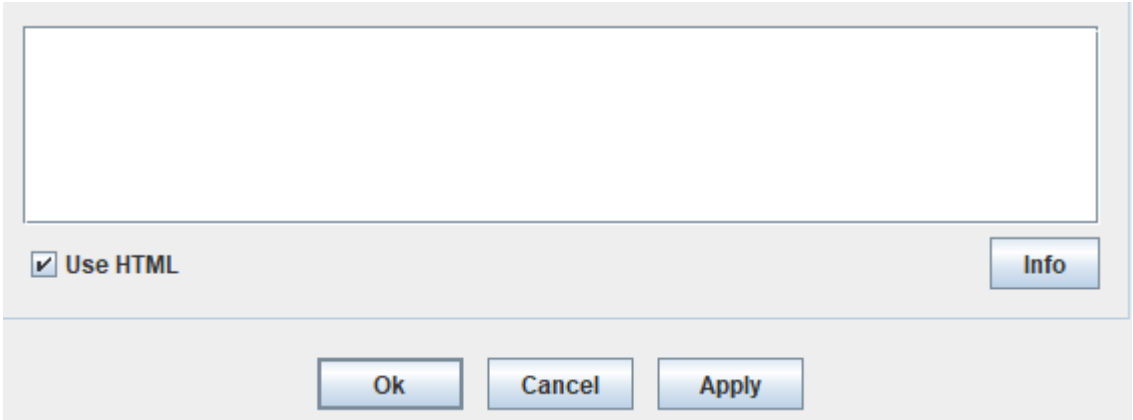
Informar o endereço do servidor de e-mail (passo 1).



Caso necessário, informar o usuário e senha (passo 2).

Dados do e-mail (passo 3).

O Botão “Info” ilustrado na imagem abaixo informa o que o usuário deve escrever no corpo da mensagem para que os dados obtidos pelo IP Monitor sejam corretamente enviados ao destinatário. Estes dados podem ser: IP público antigo, IP Público atual, data e horário do escaneamento, etc...

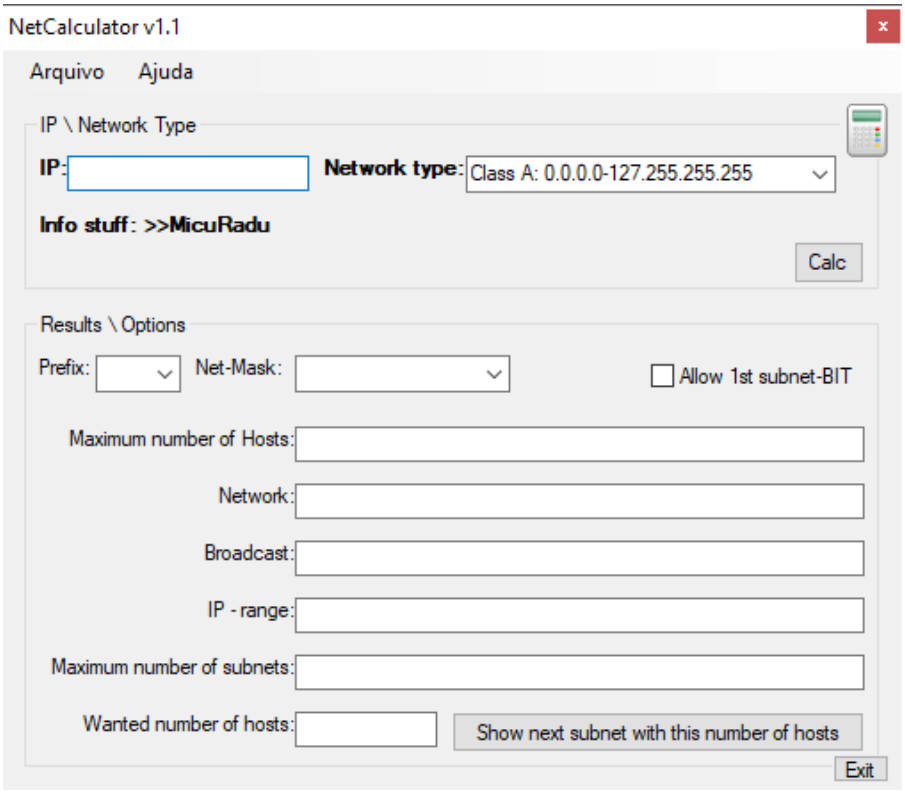


Corpo da Mensagem (passo 4).

Maiores informações sobre a ferramenta IP Monitor estão disponíveis na documentação oficial que pode ser acessada pela interface principal do IoT Manager and Security Toolkit.

NetCalculator

O NetCalculator é uma ferramenta desenvolvida em C# para realizar cálculos de rede e subredes. Ela pode ser acessada pelo menu “Arquivo” ou pelo botão “NetCalculator” na interface principal do IoT Manager and Security Toolkit.

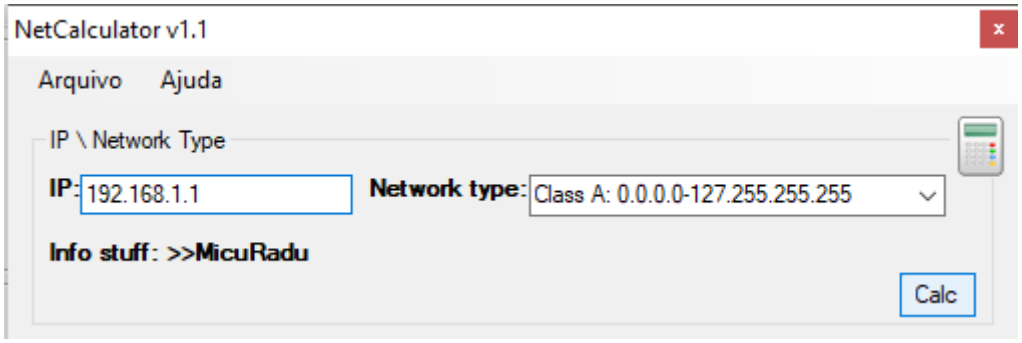


Interface Principal do NetCalculator Integrado.

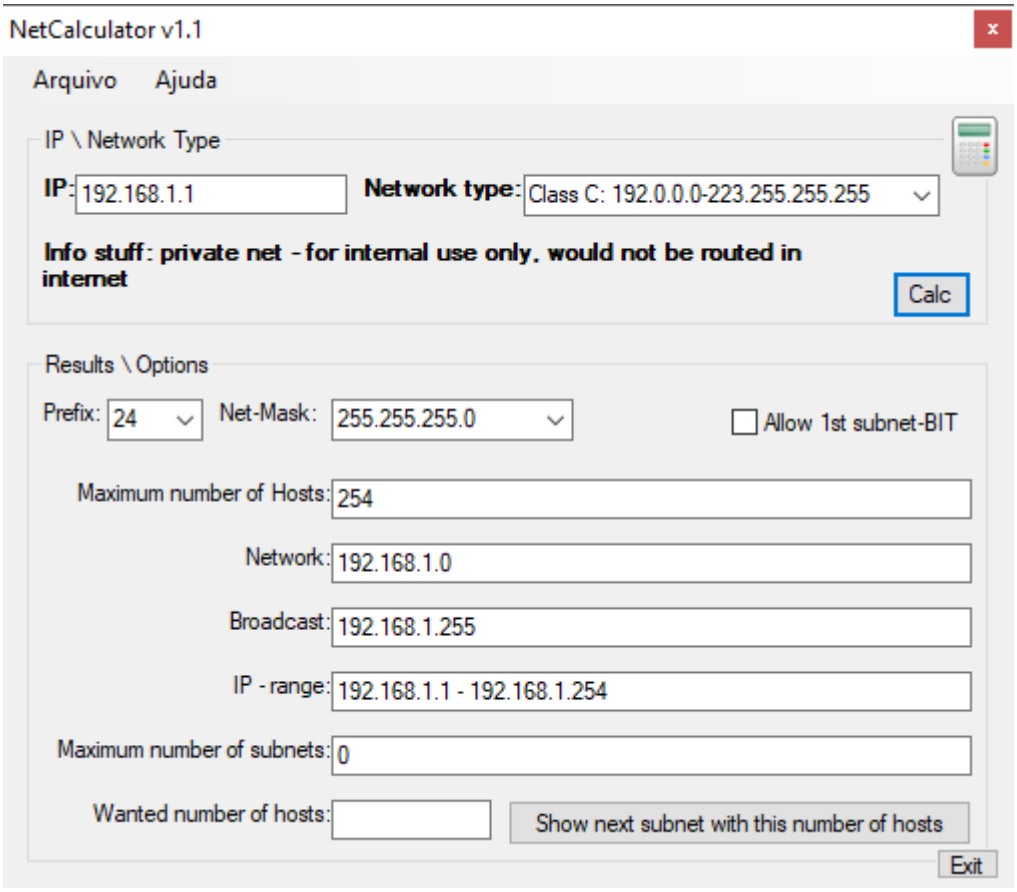
Para realizar o cálculo de rede, é necessário:

- Informar um endereço base no campo IP (passo 1).
- Clicar no botão “Calc” para realizar os cálculos (passo 2).

O NetCalculator vai automaticamente detectar a qual classe o IP pertence e mostrar os resultados na interface.

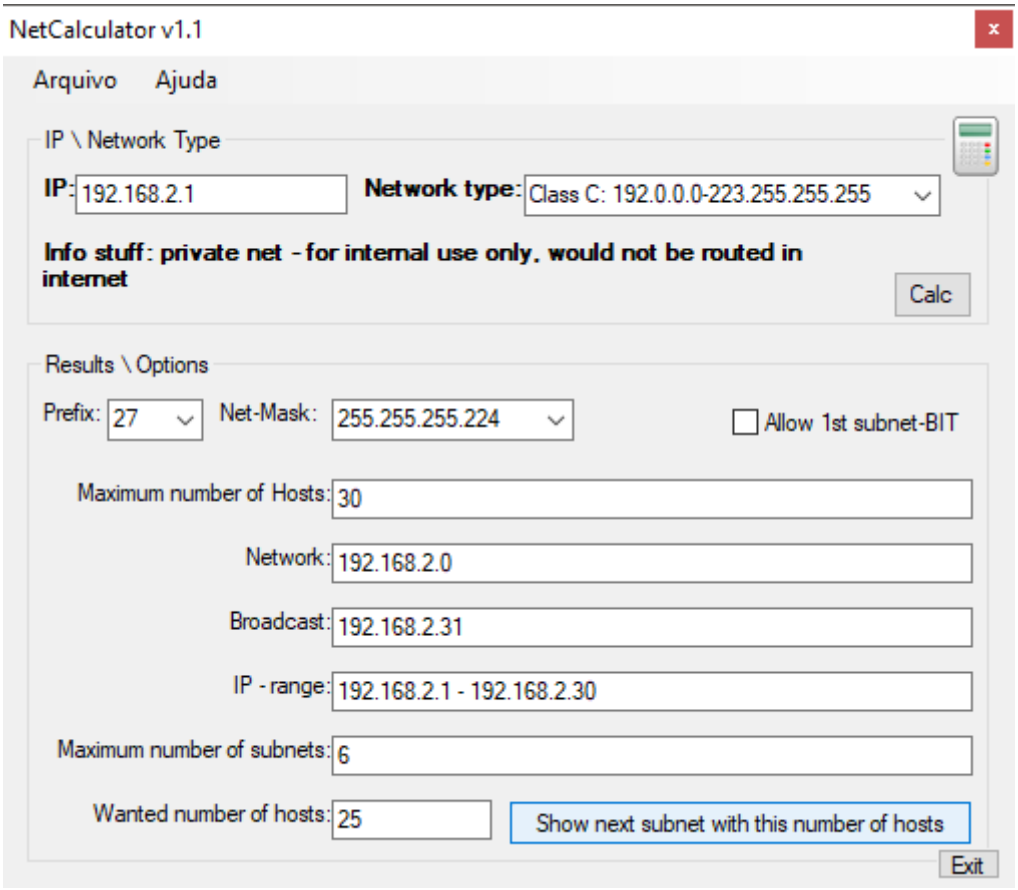


Endereço IP inserido (passo 1).



Cálculo de Rede realizado (passo 2).

Para realizar o cálculo de subredes, o usuário deve informar a quantidade desejada de hosts por subrede e clicar em “Show next subnet with this number of hosts”.

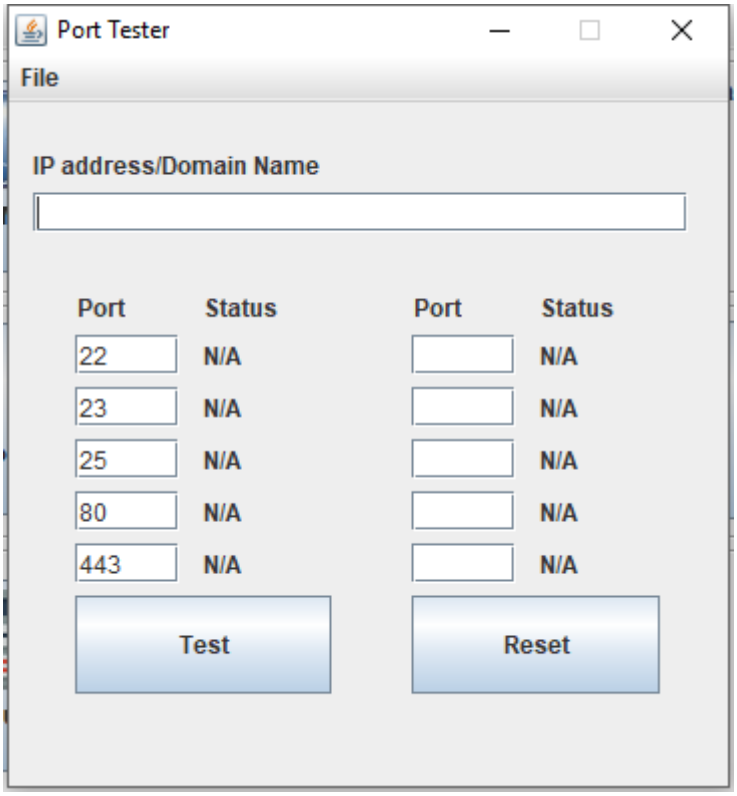


Subrede com 25 hosts.

Mais informações sobre o NetCalculator estão disponíveis na documentação oficial que pode ser acessada pela interface principal do IoT Manager and Security Toolkit.

Port Tester

O Port Tester tem como objetivo verificar se portas de dispositivos conectados em uma rede local estão abertas, open-source e desenvolvido em Java. Ela pode ser acessada tanto pelo Menu Arquivo quanto pelo botão “Port Tester”.



Interface principal do Port Tester integrado.

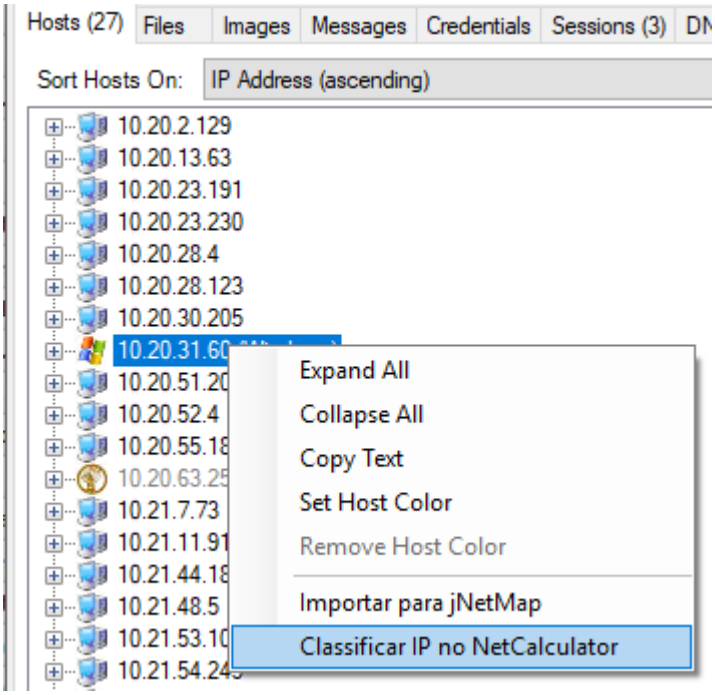
Funcionalidades de Integração

Os tópicos descritos a seguir são funcionalidades que foram adicionadas às ferramentas originais.

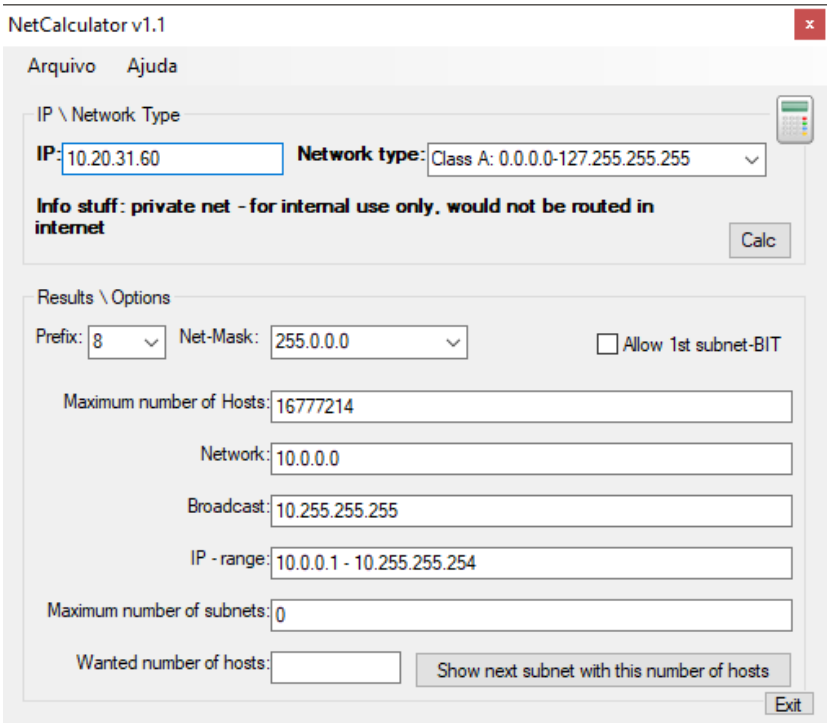
Envio de IP para o NetCalculator

O envio de um endereço IP para o NetCalculator pode ser feito através do NetworkMiner e/ou do jNetMap. Ao receber este endereço IP o NetCalculator irá exibir automaticamente os resultados na interface principal, realizando os cálculos de rede e classificando o IP. Como não é possível descobrir o prefixo de rede utilizando o NetworkMiner ou o jNetMap, o campo “Prefix” precisa ser alterado manualmente.

Para enviar um endereço IP do NetworkMiner para o NetCalculator é necessário clicar com o botão direito do mouse sobre um host previamente listado e então clicar no botão “Classificar IP no NetCalculator”.



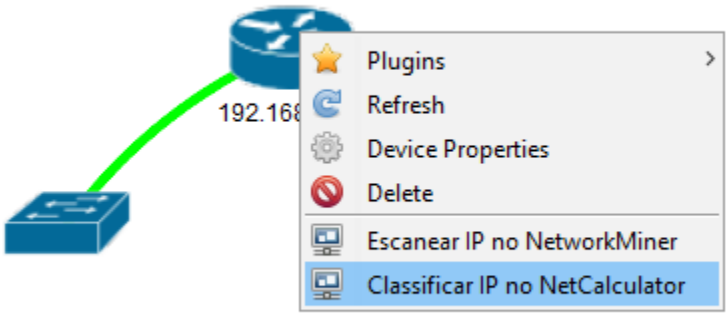
Enviando IP do NetworkMiner para o NetCalculator.



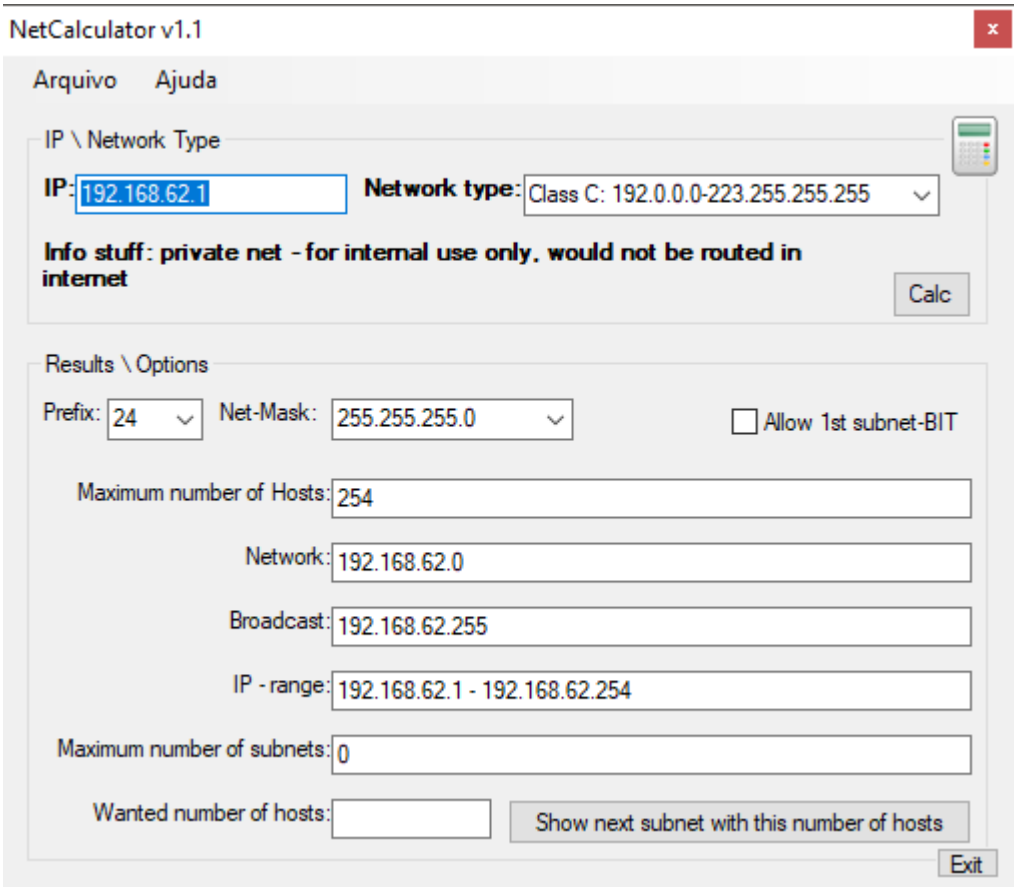
Endereço IP recebido e calculado no NetCalculator.

OBS: O prefixo pode ser alterado manualmente pelo usuário no campo “Prefix”.

Para enviar um endereço IP do jNetMap para o NetCalculator o usuário deve clicar com o botão direito do mouse sobre um dispositivo presente no mapa e então clicar no botão “Classificar IP no NetCalculator”.



Enviando IP do jNetMap para o NetCalculator.



Endereço IP recebido e calculado no NetCalculator.

OBS: O prefixo pode ser alterado manualmente pelo usuário no campo “Prefix”.