

## **3.1 – Falhas de Injeção**

### **3.1.1 – SQL Injection**

#### **O que é SQL Injection**

Segundo a definição da OWASP (2012) um ataque de SQL Injection consiste na inserção ou “injeção” de uma requisição SQL através de uma entrada de dados de um cliente em uma aplicação. Ataques SQL Injection são um tipo de ataque de injeção nos quais comandos de SQL são injetados em entradas de dados em texto plano para efetuar a execução de comandos SQL pré-definidos.

Diferentemente de um ataque Cross Site Scripting (XSS) que tem como alvo os usuários, um ataque SQL Injection tem como alvo o banco de dados a que uma aplicação está ligada.

#### **Causas de um ataque SQL Injection**

Uma ataque SQL Injection é possível quando uma aplicação utiliza dados que podem ser controlados por um atacante como parte de uma requisição SQL. O atacante pode enviar uma entrada de dados maliciosa de forma que a requisição enviada ao banco de dados será interpretada de uma forma diferente da forma prevista pelo programador da aplicação (DASWANI; KERN; KESAVAN, 2007, pág. 124).

Clarke (2009, pág. 13) completa: Vulnerabilidades de injeção SQL devem ocorrer comumente quando o desenvolvedor de uma aplicação Web não se certifica de que os valores recebidos dos formulários da Web, cookies, parâmetros de entrada, entre outros, sejam validados antes de serem passados para requisições SQL que serão executadas no banco de dados do servidor.

Qualquer processo que construa uma instrução SQL pode estar potencialmente vulnerável. A natureza diversa do SQL e os métodos disponíveis para construí-lo, proporcionam opções riquíssimas de codificação. A primeira forma de injeção SQL consiste na injeção direta de código nos parâmetros que são concatenados com comandos SQL e executados. Um ataque menos direto injeta código malicioso em Strings que estão destinadas ao armazenamento em uma tabela ou em Strings que servirão de metadados. Quando as Strings armazenadas são subseqüentemente concatenadas com um comando SQL dinâmico, o código malicioso é executado. Quando uma aplicação Web falha em filtrar adequadamente os parâmetros que são passados para instruções SQL dinamicamente criadas, (mesmo utilizando técnicas de parametrização) é possível a um atacante alterar a construção de instruções SQL. Quando um atacante é capaz de modificar uma instrução SQL, a instrução irá ser executada com os mesmos privilégios da aplicação do usuário. Quando se usa o servidor SQL para executar comandos que interagem com o sistema operacional, o processo irá executar com as mesmas permissões do componente que executou o comando, o qual freqüentemente tem boas permissões (CLARKE, 2009, pág. 7).

### **Conseqüências de um ataque SQL Injection**

Se um atacante pode controlar a entrada que é enviada para uma requisição SQL e manipula essa entrada de forma que o dado seja interpretado como código ao invés de dado, então o atacante pode executar código no banco de dados (CLARKE, 2009, pág. 13).

De acordo com a OWASP (2012) Um ataque de injeção SQL realizado com sucesso pode ler dados importantes de um banco de dados, modificá-los, inserir novos dados e até apagá-los.

Levando em consideração as informações citadas por Clarke e pela OWASP, inferimos que um atacante diante de um sistema vulnerável a falhas de injeção SQL, pode, entre outras ações, utilizar-se dos meios de entrada de dados para injetar código SQL malicioso junto aos dados, a fim de alterar a semântica das requisições SQL e conseqüentemente manipular informações no banco de dados da aplicação. As informações podem ser inseridas, modificadas ou até apagadas. A ocorrência deste fato pode resultar na incoerência das informações contidas no banco de dados e significar em grandes perdas para instituições e organizações.

### **Como evitar ataques SQL Injection**

Segundo Clarke, (2009, pág. 13) existem algumas situações as quais podem ser pontos de vulnerabilidade em sistemas que utilizam a linguagem SQL:

- **Construção de Strings Dinâmicos:** É uma técnica de programação que permite ao desenvolvedor construir instruções SQL dinamicamente em tempo de execução. Desta forma é possível criar aplicações de propósito geral mais flexíveis. Entretanto a mesma flexibilidade que pode facilitar a criação de instruções dinâmicas, pode se tornar um problema caso as requisições não sejam parametrizadas. Requisições parametrizadas são requisições que possuem um ou mais parâmetros embutidos na instrução SQL. Parâmetros podem ser passados para estas requisições em tempo de execução. Parâmetros contendo entradas do usuário embutidas não são interpretados como comandos a serem executados, não havendo assim oportunidade para injeção de código. Este método de embutir parâmetros na instrução SQL é mais eficiente e muito mais seguro do que criar dinamicamente e executar instruções SQL utilizando técnicas de junção de Strings.

- **Manipulação Incorreta dos Caracteres de Escape:** Banco de dados SQL interpretam o caractere (') como o divisor entre código e dados. Sendo assim, toma-se que tudo o que segue este caractere é código a ser executado e tudo o que está encapsulado entre estes caracteres é dado. Desta forma, este caractere é utilizado em ataques de injeção SQL para “escapar” do contexto da requisição do desenvolvedor para que o atacante possa construir suas próprias requisições e possa executá-las. Existem ainda outros caracteres de escape, como por exemplo no Oracle, o espaço em branco, pipe duplo (||), vírgula, ponto, (\*/) e aspas duplas.
- **Manipulação Incorreta de Tipos:** Como visto no caso anterior, dados e código são separados por um caractere ('), entretanto nem todo dado de fato precisa estar entre aspas simples, como é o caso de números inteiros por exemplo. Se colocarmos um inteiro entre aspas simples, o dado será tratado como String (texto) e não como um número inteiro propriamente dito. Dependendo do tipo de dado esperado pela entrada em uma aplicação que utiliza SQL, pode-se criar uma vulnerabilidade, como no caso de uma entrada de código de usuário, a qual o usuário deve entrar com um código numérico. Assim o código digitado não acompanhará caracteres de escape e pode ser um potencial risco para o banco de dados caso a entrada não for filtrada.
- **Manipulação Incorreta de Requisições Assembly:** Algumas aplicações complexas precisam ser codificadas com sentenças de SQL dinâmicas, pois as tabelas ou campos que precisam ser acessados ainda não são conhecidos no estágio de desenvolvimento ou ainda não existem. O fato de uma aplicação gerar uma entrada de dados em tempo de execução faz, muitas vezes, com que o desenvolvedor confie na entrada de dados gerada.

Entretanto ela pode ser controlada pelo usuário, se for, por exemplo, submetida por uma requisição do tipo GET. O atacante pode submeter os seus dados no lugar dos dados gerados pela aplicação, realizando assim um ataque de injeção SQL.

- **Manipulação Incorreta de Erros:** Lidar com erros de forma inapropriada pode acarretar uma série de problemas de segurança para um Web site. O problema mais comum ocorre quando mensagens de erro internas detalhadas, como as que envolvem estruturas de tabelas ou código de erros são mostrados ao usuário ou atacante. Essas mensagens revelam detalhes de implementação que nunca deveriam ser revelados. Esses detalhes podem fornecer informações importantes referentes à estrutura do Web site para o atacante, facilitando a descoberta de falhas. Mensagens de erro de tabelas podem ser usadas para extrair informações das tabelas e auxiliar na construção de injeções com o intuito de escapar da requisição do desenvolvedor ou manipular as informações contidas na tabela.
- **Manipulação Incorreta de Submissões Múltiplas:** Desenvolvedores de aplicações tendem a projetar aplicações as quais guiarão os usuários por um fluxo esperado do processo, imaginando que o usuário irá seguir os passos lógicos que foram projetados. Por exemplo, eles esperam que se um usuário chegou até o terceiro formulário em uma série de formulários, então o usuário deve ter preenchido o primeiro e o segundo formulários. Na realidade, freqüentemente é muito simples desviar do fluxo de dados esperado, requisitando os recursos fora de ordem diretamente através de seus URLs. Um atacante pode acessar o segundo formulário diretamente, sem acessar o primeiro, ou ele pode simplesmente enviar dados válidos como entrada para

o primeiro formulário e depois manipular estes dados quando eles forem enviados ao segundo formulário. Desta forma a aplicação irá “pensar” que os dados do primeiro formulário que acompanham o segundo formulário, já foram validados, quando na verdade estes dados podem ter sido alterados.

Tendo como base os pontos de vulnerabilidade listados acima, de forma geral, podemos evitar ataques de injeção SQL dedicando atenção a estes pontos e desenvolvendo aplicações que se esquivam ao máximo destes problemas.

Abaixo estão listadas algumas recomendações de Clarke, (2009, pág. 373):

- Use instruções parametrizadas
  - Use instruções parametrizadas ao invés de SQL dinâmico na construção de requisições SQL.
  - Use instruções parametrizadas somente quando você está fornecendo dados; você não pode utilizá-las para fornecer palavras-chaves SQL ou identificadores (como nomes de tabelas ou colunas).
- Valide as entradas
  - Sempre utilize validação de entrada do tipo “White list” (aceitando somente entradas esperadas) onde puder.
  - Certifique-se de ter validado o tipo, tamanho, intervalo e conteúdo de todas as entradas controladas por usuário na aplicação.
  - Utilize validação de entrada do tipo “Black list” (aceitando tudo, exceto entradas contidas na Black list) somente quando você não puder utilizar validação de entrada do tipo “White list”.
  - Nunca utilize apenas “Black list”. Ao menos combine-a com a codificação de saída ou outra validação.

- Codifique a saída
  - Certifique-se de que requisições SQL contendo entradas controladas por usuário são codificadas corretamente para prevenir que caracteres especiais alterem a requisição.
  - Se você está utilizando cláusulas do tipo LIKE, certifique-se de que os “wild cards” da cláusula LIKE estão propriamente codificados.
  - Certifique-se de que os dados recebidos da base de dados estão em um contexto apropriado da entrada de validação.
- Normalize (Padronize)
  - Os filtros de validação de entrada e codificação da saída devem ser executados depois que a entrada foi decodificada ou está na forma canônica.
  - Esteja ciente de que existem múltiplas representações de qualquer caractere e múltiplas formas de codificá-lo.
  - Onde possível, utilize validação de entradas do tipo “White list” e rejeite formas não canônicas de entrada.
- Projete visando evitar os perigos de Injeção SQL
  - Use procedimentos armazenados para que você possa ter permissões mais granulares no nível da base de dados.
  - Você pode usar uma camada de abstração de acesso à dados para reforçar a segurança do acesso a dados através de uma aplicação inteira.
  - Considere controles adicionais sobre informações importantes em tempo de projeção e desenvolvimento.

### 3.1.2 – Cross Site Scripting (XSS)

#### O que é Cross Site Scripting

Segundo a definição da OWASP (2012) ataques de Cross Site Scripting são um tipo de problema de injeção, no qual scripts maliciosos são injetados em web sites confiáveis.

A OWASP categoriza os ataques XSS em 2 categorias:

- **Armazenados (Stored):** São os ataques nos quais o código injetado é permanentemente armazenado nos servidores alvo, como em um banco de dados, fórum de mensagens, log de visitantes ou em um campo de comentários, etc. Assim, a vítima recebe o código malicioso do servidor quando ela solicita a informação armazenada.
- **Refletidos (Reflected):** São os ataques nos quais o código injetado é refletido dos servidores web, como em uma mensagem de erro, resultado de pesquisa, ou qualquer outra resposta que inclua alguma parte ou toda a entrada enviada ao servidor como parte da requisição. Ataques refletidos de XSS são entregues às vítimas através de outros caminhos, como uma mensagem de e-mail por exemplo. Quando o usuário clica em um link malicioso ou submete um formulário malicioso, o código injetado viaja até o servidor web vulnerável, que reflete o ataque de volta ao navegador do usuário. O navegador então executa o código, porque este veio de um servidor confiável.

Existe ainda um outro tipo de ataque XSS pouco conhecido que é baseado em DOM e modifica o ambiente DOM do navegador da vítima.



Um ataque XSS é um ataque de injeção bem como o SQL Injection, entretanto a principal diferença entre esses ataques é que um ataque Cross Site Scripting tem como alvo o usuário enquanto que um ataque SQL Injection tem como alvo o banco de dados de um servidor.

### **Causas de um ataque XSS**

Para a OWASP (2012) ataques de Cross Site Scripting ocorrem quando:

- Dados entram em uma aplicação web através de uma fonte não confiável. Frequentemente uma requisição web.
- O dado está incluso em um conteúdo dinâmico que é enviado por um usuário da web sem ser validado previamente e verificada a existência de código malicioso.

Para a OWASP (2012) ataques de Cross Site Scripting ocorrem quando um atacante usa uma aplicação web para enviar código malicioso, geralmente na forma de browser side script, para um usuário final diferente. Falhas que permitem o sucesso desses ataques são comuns e ocorrem em qualquer ambiente que utilize uma aplicação web a qual requeira uma entrada do usuário e utilize essa entrada na saída gerada pela aplicação sem antes validar e ou codificar a entrada do usuário.

O conteúdo malicioso enviado para o navegador do usuário frequentemente estão na forma de um segmento de JavaScript, mas podem também incluir HTML, Flash ou outros tipos de códigos que o navegador possa executar. A variedade de ataques baseados em XSS é quase ilimitada, mas comumente incluem a transmissão de dados privados como cookies ou outras informações de sessões ao atacante, redirecionando a vítima ao conteúdo da web controlado pelo atacante ou ainda

executando ações maliciosas na máquina do usuário guiadas pelo site vulnerável (OWASP, 2012).

Para o CGI Security (2012) um ataque de Cross Site Scripting ocorre quando uma aplicação coleta dados maliciosos de um usuário. O dado é normalmente coletado na forma de hyperlink, o qual contém conteúdo malicioso em si mesmo. O usuário na maior parte das vezes clicará neste link em um outro web site, em uma mensagem instantânea ou até mesmo em um e-mail. É comum que o atacante codifique a parte maliciosa do hyperlink em HEX (ou outros métodos de codificação) para evitar levantar suspeitas de que o link é fraudulento. Depois que os dados são coletados pela aplicação web, uma página de saída dos dados é criada para o usuário contendo os dados maliciosos que foram originalmente enviados, mas de uma maneira que a apresentação dos dados seja similar à apresentada pelo web site original.

### **Consequências de um ataque XSS**

Um atacante pode usar XSS para enviar um script malicioso ao browser de um usuário confiável sem que o browser do usuário final saiba que o script não é confiável. Desta forma, o browser irá executar o script e irá acreditar que o script veio de uma fonte confiável. O script malicioso pode acessar qualquer cookie, sessão de tokens ou qualquer outra informação importante armazenada pelo browser do usuário e utilizada naquele site. Alguns scripts são até capazes de reescrever páginas HTML (OWASP, 2012).

Eventualmente atacantes irão injetar JavaScript, VBScript, ActiveX, HTML ou Flash em uma aplicação vulnerável para enganar um usuário e coletar informações dele. Pode haver diversas consequências tais como: seqüestro de sessão, alteração nas

configurações do usuário, roubo ou envenenamento de cookies e até propagandas falsas (CGI Security, 2012).

Esse tipo de ataque não pode ser evitado pelo fato de se usar criptografia, como por exemplo, uma conexão SSL (Secure Socket Layer). A aplicação web com SSL funciona da mesma forma e os dados continuarão sendo criptografados, entretanto os dados em si é que são maliciosos e ao serem decodificados em seu destino final e executados farão o ataque acontecer.

### **Como saber se você está vulnerável e como se proteger**

Se você é o dono da aplicação, nunca confie em conteúdos digitados por usuários e sempre filtre metadados. Essa prática irá eliminar a maior parte de ataques XSS. Se você é um usuário final a maneira mais fácil de se proteger é seguir apenas links encontrados dentro da página oficial que você deseja visitar. Se você estiver lendo uma mensagem e a mensagem contiver um link para um web site conhecido, prefira digitar você mesmo o endereço em seu navegador ao invés de clicar no link para acessar a página. Isso provavelmente irá eliminar 90% dos riscos de você ser vítima de um ataque XSS. As vezes, códigos maliciosos podem ser executados simplesmente ao abrir um e-mail. Se você desconhece o remetente, seja cuidadoso ao abri-lo. Uma outra opção para se proteger de ataques XSS é desabilitar a execução de JavaScript em seu navegador, porém esta estratégia pode prejudicar a sua navegação em sites que utilizam JavaScript (CGI Security, 2012).

## **3.2 – Vírus**

### **Definição**

A OWASP (2012) define um vírus de computador como: Um pequeno programa projetado para causar algum tipo de dano no computador infectado, apagando dados, capturando informações ou alterando o comportamento normal de operação do computador.

Como os vírus humanos têm diferentes níveis de gravidade, os vírus de computador também variam de leves a totalmente destrutivos. Um vírus de computador não se espalha sem a atividade humana, precisamos enviar a alguém um arquivo ou e-mail para espalhá-lo (diferentemente dos worms). Os vírus podem assumir diversas formas, como imagens, áudios, vídeos entre outros anexos (OWASP, 2012).

### **Propósitos**

Vírus podem ser projetados para diversos propósitos. Vírus mais antigos eram tipicamente projetados para gerar alertas de mensagens, promover seus desenvolvedores como habilidosos em programação ou destruir dados (GREGG, 2008, pág. 261).

Vírus são geralmente maliciosos em suas intenções e são criados por seus autores para se replicarem em todos os arquivos executáveis e macros encontrados em um hospedeiro. Posteriormente eles dependem do compartilhamento dos arquivos infectados do usuário com outros usuários em diferentes computadores para que possam se propagar (BELAPURKAR; CHAKRABARTI; PONNAPALLI; VARADARAJAN, 2009, p. 58).

Grande parte dos vírus mais antigos tinha como alvo sistemas operacionais Microsoft Windows. Apesar de computadores com sistemas Linux não serem imunes, é mais difícil um vírus causar no Linux os mesmos danos que pode causar no Windows. Para um vírus ter sucesso no Linux, ele precisa infectar arquivos de

posse do usuário. Programas de posse do usuário root são freqüentemente acessados por um usuário normal através de uma conta não privilegiada. Vírus para Linux também precisam de um meio ou mecanismo com o qual possam atacar (GREGG, 2008, pág. 262).

Para Gregg (2008, pág. 262) o conceito dirigente para os vírus mais antigos foi a replicação. Isso significou que para um vírus ter sucesso, ele precisava se reproduzir rápido, antes que fosse descoberto e erradicado.

### **Características**

Como as empresas de antivírus evoluíram nas formas de detecção de vírus, os criadores de vírus revidaram tentando desenvolver vírus que são mais difíceis de serem detectados. Uma das técnicas é a criação de vírus multipartidos. Um vírus multipartido pode utilizar mais de um método de propagação. Por exemplo, o vírus NATAS (Satan lido de trás pra frente) infecta setores de boot e arquivos de programa. A idéia é que esta técnica daria ao vírus mais chances de sobrevivência. Outra técnica que os desenvolvedores de vírus criaram foi tornar o vírus polimórfico. Vírus polimórficos podem trocar a sua assinatura toda vez que eles se replicam e infectam um novo arquivo. Esta técnica torna a detecção de vírus por sistemas antivírus muito mais difícil (GREGG, 2008, pág. 263).

De acordo com Gregg (2008, pág. 263) existem 3 componentes principais que compoem um vírus polimórfico: um corpo do vírus cifrado, uma rotina de decifragem e um mecanismo de mutação. O processo de infecção polimórfico é o seguinte:

1. A rotina de decifragem primeiro ganha controle do computador e depois decifra o corpo do vírus e o mecanismo de mutação.

2. A rotina de decifragem transfere o controle do computador para o vírus, o qual localiza um novo programa para infectar.
3. O vírus faz uma cópia de si e do mecanismo de mutação na RAM.
4. O vírus invoca o mecanismo de mutação, o qual randomicamente gera uma nova rotina de decifragem capaz de decifrar o vírus ainda contendo pouca ou nenhuma semelhança com qualquer rotina de decifragem anterior.
5. O vírus cifra a nova cópia do corpo do vírus e do mecanismo de mutação.
6. O vírus acrescenta a nova rotina de decifragem, juntamente com o novo vírus cifrado e o mecanismo de mutação a um novo programa.

Como resultado, não apenas o corpo do vírus é cifrado, mas a rotina de decifragem também muda de infecção para infecção. Desta forma, duas infecções do mesmo vírus aparentam ser diferentes, o que confunde o scanner de vírus, que procura por uma seqüência de bytes que identifica uma rotina de decifragem específica (GREGG, 2008, pág. 263).

Vírus ocultos tentam esconder sua presença de ambos, sistema operacional e sistema antivírus, utilizando-se das seguintes técnicas:

- Escondendo a alteração na data e hora do arquivo.
- Escondendo o aumento de tamanho do arquivo infectado.
- Auto cifrando-os.

(GREGG, 2008, pág. 264).

### **Formas de Propagação**

Desde os primeiros vírus de computador, esse tipo de malware teve como pilar alguns métodos básicos de propagação. A propagação de vírus requer atividade

humana, como inicializar um computador, executar um arquivo auto-executável em um CD ou abrir um anexo de e-mail. Existem três formas básicas pelas quais os vírus de computador se propagam:

- **Infecção da MBR (Master Boot Record):** Este é o método original de ataque. Ele funciona através do ataque à MBR de disquetes e discos rígidos. Este método era eficiente nos dias em que havia grande movimentação de disquetes.
- **Infecção de arquivo:** Essa forma de vírus baseia-se na execução de um arquivo pelo usuário. Extensões como .com e .exe são tipicamente utilizadas. Usualmente alguma forma de engenharia social é utilizada para levar o usuário a executar o programa. As técnicas incluem renomear o programa ou tentar renomear a extensão .exe para fazê-lo parecer uma imagem, por exemplo.
- **Infecção de macro:** A forma mais moderna de vírus começou a aparecer nos anos 90. Vírus de macro exploram serviços instalados no computador. Muitos devem se lembrar do vírus “I Love You”, um dos pioneiros na infecção por macro. Vírus de macro infectam aplicações como o Word ou Excel anexando-se às etapas de inicialização destas aplicações e então quando a aplicação é executada, as instruções do vírus executam antes que o controle seja passado para a aplicação. Depois o vírus se replica, infectando partes adicionais do computador.

(GREGG, 2008, pág. 262).

A medida que mais sistemas começaram a entrar para as redes, os autores de vírus encontraram uma forma mais fácil de infectar hospedeiros pelas redes. Isto levou o

advento dos worms (BELAPURKAR; CHAKRABARTI; PONNAPALLI; VARADARAJAN, 2009, p. 58).

### **Fatores de Risco**

Um vírus de computador pode roubar ou apagar informações, fazer o computador ficar mais lento ou simplesmente bagunçar o sistema operacional. Nos dias de hoje, os vírus mais comuns são aqueles que roubam informações relacionadas a Internet Banking, para que o atacante possa transferir o dinheiro da vítima para sua conta privada, pagar contas ou comprar algo pela internet (OWASP, 2012).

### **3.3 – Comunicações Inseguras: Sniffing**

Aplicações geralmente falham na hora de encriptar tráfego de rede quando é necessário proteger comunicações sensíveis. A encriptação (geralmente SSL) deve ser usada em todas as conexões autenticadas, especialmente páginas web com acesso via internet, mas também em conexões com o backend. Senão, o aplicativo irá expor uma autenticação ou o token de sessão. Adicionalmente, a autenticação deve ser usada sempre que dados sensíveis, assim como cartões de crédito ou informações de saúde são transmitidos. Aplicações cujo modo de encriptação possa ser subvertido são alvos de ataques. Todos os frameworks de aplicações web são vulneráveis às comunicações inseguras (OWASP, 2007, pág. 27).

Existem outros problemas relacionados com comunicações inseguras como, por exemplo, o ataque do homem do meio, entretanto todos estes problemas exploram em sua essência, a mesma falha, que é a comunicação insegura através de uma rede. Como o Sniffing (ou farejamento) de pacotes é um dos problemas mais



comuns e conhecidos e ainda possui uma grande variedade de ferramentas para a sua exploração, iremos utilizá-lo como foco desta sessão para tratar sobre comunicações inseguras.

### **Definição**

Belapurkar, Chakrabarti, Ponnappalli e Varadarajan (2009, pág. 134) definem: Sniffing ou espionagem é o ato de monitorar o tráfego trocado entre dois pontos em uma rede. Esta é uma das formas mais antigas de ataque disfarçado, realizado para capturar informações importantes sem o consentimento do remetente e do destinatário.

Em muitas redes trafegam uma grande quantidade de informações e boa parte delas não deve estar protegida com criptografia. Mesmo que estejam criptografadas, o algoritmo de cifragem utilizado deve ser fraco ou vulnerável (GREGG, 2008, pág. 174).

Com um simples farejador de pacotes, um atacante pode facilmente ler todo o texto plano que trafega na rede. Os atacantes podem ainda crackear pacotes cifrados com algoritmos fracos e decifrar informações que os desenvolvedores de serviços Web consideram seguras. O farejamento de pacotes requer que o atacante insira um farejador de pacotes no caminho entre o provedor de serviço e o consumidor (BELAPURKAR; CHAKRABARTI; PONNAPALLI; VARADARAJAN, 2009, p. 134).

### **Sniffing Passivo vs. Sniffing Ativo**

Normalmente quando algo é referido como passivo, é porque não afeta o ambiente no qual está. Inversamente, quando algo é definido como ativo, é porque realiza algo

ativamente para si ou para o seu ambiente (HARRIS, HARPER, EAGLE, NESS, LESTER, 2005, pág. 127).

- **Sniffing Passivo:** O sniffer ouve e captura quadros para então analisá-los. Um sniffer passivo simplesmente põe o host em modo promíscuo para capturar tudo o que ele ouvir na rede. Esse tipo de sniffer funciona bem para redes que não possuam a presença de switches, mas sim a presença de hubs.
- **Sniffing Ativo:** O sniffing ativo é um método para farejar redes que possuem a presença de switches, os quais controlam o fluxo de dados para hosts específicos. Como o tráfego entre dois outros hosts normalmente não é direcionado para a porta no qual o sniffer está ativo, um método de sniffing ativo tenta roubar o tráfego para a máquina farejadora antes que ele chegue em seu destino. Este método envolve o envio de requisições ARP falsificadas ou respostas para um host alvo, para que ele atualize sua cache ARP com um IP falso no mapeamento de endereço address-to-MAC. O envenenamento da cache ARP, como isto é chamado, força o computador da vítima a endereçar os quadros (na camada 2) que estão sendo enviados para outro endereço de IP legítimo (na camada 3), para o endereço MAC do sniffer.

(HARRIS, HARPER, EAGLE, NESS, LESTER, 2005, pág. 127).

### **Defesa contra Sniffing**

Segurança de porta é uma característica de muitos switches que limita quais endereços MAC podem usar uma designada porta do switch. Esta pode ser uma forma de fazer com que apenas 1 endereço MAC possa ser plugado nesta porta, forçando a porta a rejeitar quadros enviados por outros endereços MAC. A parte

ruim desta solução é a sobrecarga do administrador que terá que associar cada endereço MAC a uma porta do switch exaustivamente. Segurança de portas pode adicionar uma nova camada de segurança a rede, mas pode ser administrativamente cara (HARRIS, HARPER, EAGLE, NESS, LESTER, 2005, pág. 137).

Há uma forma melhor de defender-se contra o envenenamento de cache ARP sem grandes custos administrativos, que é a detecção. Envenenamento de ARP e combinações estranhas de endereços MAC com endereços IP podem ser detectadas por ferramentas como a ARPWatch. Ela roda em Unix/Linux e envia um e-mail de notificação quando alguma atividade suspeita é detectada. A ferramenta WinARPWatch é a correspondente para sistemas Windows (HARRIS, HARPER, EAGLE, NESS, LESTER, 2005, pág. 138).

Finalmente, a melhor forma de defesa contra o sniffing ativo ou qualquer sniffing é a criptografia. Aplicações como SSH que proporcionam autenticação e criptografia dos dados não estão suscetíveis a ataques de sniffing (HARRIS, HARPER, EAGLE, NESS, LESTER, 2005, pág. 138).

Use SSL para todas as conexões que são autenticadas ou que transmitam informações sensíveis ou de valor, assim como credenciais, cartões de crédito, e outros. Assegure que as comunicações entre os elementos da infra-estrutura, como servidores de web e sistemas de banco de dados, estão propriamente protegidas pelo uso de camadas de transporte de segurança ou de encriptação de nível de protocolo para credenciais e informações de valor intrínseco (OWASP, 2007, pág. 28).

## **Ferramentas Sniffers**

Ferramentas para Sniffing vêm em diferentes formas e tamanhos, desde dirigidas por uma simples linha de comando à aplicações gráficas com conectividade com bases de dados e capacidade de administração remota. Não importa o tipo de alarme emitido por cada um, todos os sniffers compartilham o atributo comum de serem capazes de capturar quadros de alguma rede. Apesar de sniffers serem comumente referenciados como capturadores de pacotes, é a camada 2 de quadros que é realmente capturada. A camada 3 de pacotes, que está dentro dos quadros, é capturada também, mas ela é apenas parte da informação gravada. É comum, entretanto, que as pessoas se refiram a qualquer protocolo de unidade de dados (PDU), independentemente da camada do modelo OSI que ele está, simplesmente como “um pacote” e muitos textos não distinguem a camada 3 de pacotes da camada 2 de quadros quando descrevem analisadores de protocolos (HARRIS, HARPER, EAGLE, NESS, LESTER, 2005, pág. 125).

Para decodificar os quadros e funcionar em modo promíscuo, os sniffers tiram vantagem da biblioteca capturadora de pacotes libpcap para Unix/Linux e WinPcap para Windows. Essas bibliotecas, que são utilizadas por muitas ferramentas que precisam dissecar pacotes, são normalmente baixadas separadamente e precisam ser instaladas antes da instalação da ferramenta sniffer (HARRIS, HARPER, EAGLE, NESS, LESTER, 2005, pág. 126).

### **3.4 – Negação de Serviço (DoS Attack)**

### **3.5 – Buffer Overflow**

### **3.6 - BackDoor**

## **4.1 – Scanner de Rede: NMAP**

### **Escaneamento de Portas**

O escaneamento de portas é o processo de se conectar a portas TCP e UDP com o propósito de encontrar quais serviços e aplicações estão disponíveis no dispositivo alvo. (GREGG, 2008, p. 111).

Este tipo de prática pode ser utilizada tanto para fins de auditoria de segurança da rede, ajudando os administradores da rede a encontrarem portas abertas que podem comprometer a segurança de um terminal isolado ou de uma rede ou parte dela, bem como por pessoas mal intencionadas a procura de portas abertas na rede, as quais poderão ser exploradas e servirem de portas de entrada para ataques à rede.

Segundo GREGG (2008, p. 122), o escaneamento de portas foi questionado quanto a sua legalidade nos Estados Unidos. Enquanto que para alguns o escaneamento de portas era considerado crime, para outros esta prática não poderia ser considerada crime uma vez que não traria prejuízo algum ao alvo.

A questão de ser ou não ser considerada crime a prática de escaneamento de portas é discutida até hoje, pois apesar de não trazer prejuízos diretos ao alvo, pode levar à uma exposição de falhas de segurança do elemento escaneado. Devido a este tipo de exposição, que pode facilitar e até levar a um ataque, escanear portas é uma prática que só deve ser feita em redes sob sua jurisdição ou com a concessão de autorização para tal ato. O escaneamento de redes não autorizadas pode ser considerado ilegal e o autor da prática poderá estar sujeito a multas e penalidades determinadas de acordo com a legislação local vigente.

## A Ferramenta

O NMAP é uma ferramenta de código aberto para exploração de rede e auditoria de segurança, o qual serve tanto para escanear redes amplas bem como hosts individuais. Utilizando pacotes IP em estado bruto, o NMAP consegue descobrir de maneira eficiente, quais hosts estão disponíveis na rede e quais serviços estes oferecem. É possível ainda descobrir mais informações do alvo escaneado tais como tipo e versão do sistema Operacional utilizado, quais tipos de filtros de pacotes (firewall) estão em uso, entre outras (NMAP, 2012).

O resultado de um scan realizado com o NMAP conterá entre outras informações uma tabela de portas do alvo, a qual é o objeto principal do nosso estudo com esta ferramenta, pois são as informações obtidas do estado de cada uma destas portas que irá compor um relatório sobre a segurança do alvo escaneado. Nesta tabela encontraremos:

- O número da porta descoberta
- O protocolo por ela utilizado
- O nome do serviço
- O estado da porta que pode ser:
  - **Aberto:** Uma aplicação esta escutando a porta e esperando o recebimento de pacotes nesta porta.
  - **Filtrado:** Existe um filtro, firewall ou algum outro obstáculo na rede bloqueando a porta de forma que o NMAP não consegue determinar se ela está aberta ou fechada.
  - **Fechado:** A porta recebe e responde aos pacotes enviados pelo NMAP, entretanto não existe nenhuma aplicação ouvindo a porta.

Portas fechadas também necessitam de atenção e observação pois em um determinado momento elas podem se abrir.

- **Não Filtrado:** Para este caso, a porta responde aos pacotes enviados pelo NMAP, entretanto o NMAP não consegue determinar se a porta está aberta ou fechada.

## **Especificação do Alvo**

Com o NMAP é possível escanear hosts individuais mas também redes inteiras, ou hosts adjacentes a um especificado.

Para escanear um host único, basta digitar o endereço do host tal como 192.168.1.1 por exemplo.

Para escanear hosts adjacentes basta utilizar o estilo de endereçamento CIDR, especificando um endereço base e o número de bits o qual o NMAP deverá manter do início do endereço fornecido nesta varredura. Por exemplo, 192.168.10.0/24 escanearia os 256 hosts entre 192.168.10.0 (binário: 11000000 10101000 00001010 00000000) e 192.168.10.255 (binário: 11000000 10101000 00001010 11111111). De um total de 32 bits que compõem o endereço, apenas os 24 primeiros serão mantidos, variando os 8 últimos. O menor valor de bits especificado permitido é 1 e o maior 32 (que corresponderia a escanear o próprio host).

Além disso o NMAP permite mais de 1 tipo de especificação de hosts na mesma linha de comando, não sendo necessário repetir o comando para outros hosts desejados caso haja mais de 1 ou várias faixas de hosts a serem pesquisadas.

## **Descoberta de Hosts**

Para os casos de redes muito grandes, às vezes não se sabe ao certo quais hosts escanear na rede. Mesmo que se saiba a faixa de endereços a qual se deve escanear, nem todos os hosts podem estar ativos no momento do scan e por isso seria dispendioso fazer um scan de toda a rede porta a porta. Por isso muitas vezes temos a necessidade de saber anteriormente quais são os hosts existentes na rede e quais hosts estão ativos.

O NMAP oferece suporte à descoberta de hosts o que nos permite otimizar as nossas buscas, economizar nossos recursos e o nosso tempo. Se nenhuma opção de descoberta de hosts for especificada, por padrão o NMAP envia pacotes TCP ACK destinados à porta 80 e espera um pacote ICMP (Internet Control Message Protocol) Echo Request como retorno para cada host enviado.

Dentre as opções para descoberta de hosts temos:

**-sL (Scan Listagem):** É uma forma simples de listar cada host da rede especificada, sem enviar nenhum pacote aos hosts alvo. O NMAP irá se utilizar de uma técnica de DNS reverso para descobrir os nomes dos hosts. Neste modo o NMAP irá reportar também ao final do scan a quantidade de IPS da rede.

**-sP (Scan usando Ping):** Esta opção faz com que o NMAP realize o scan na rede apenas usando o ping, de forma que somente os hosts ativos irão responder. A consequência de usar o ping para descobrir os hosts ativos da rede é o envio de pacotes à todos os hosts da rede, o que poderá levantar suspeitas de que a rede está sendo escaneada. Por outro lado, pode ser mais valioso saber quais hosts estão ativos na rede do que apenas a listagem dos hosts.



**-p0 (Sem Ping):** A opção sem ping, pula o descobrimento de hosts da rede. Nesta opção o NMAP fará um scan mais agressivo, como o escaneamento de portas e detecção de SO em hosts que foram verificados como ativos.

**-PS [listadeportas] (Ping usando TCP SYN):** Neste modo, o NMAP envia um pacote TCP vazio com a flag SYN marcada. Por padrão este pacote é enviado à porta 80, mas uma porta alternativa ou uma lista de portas pode ser especificada como parâmetro (ex: -PS22,23,25,80,113,1050,35000) sendo que para mais de uma porta, o scan nas outras portas são executados em paralelo.

A flag SYN indica aos sistemas remotos que você está tentando estabelecer uma comunicação. Normalmente a porta de destino estará fechada e um pacote RST (reset) será enviado de volta. Caso a porta esteja aberta, o alvo irá dar o segundo passo do cumprimento de-três-vias (3-way-handshake) do TCP respondendo com um pacote TCP SYN/ACK TCP.

A máquina executando o NMAP derruba então a conexão recém-nascida respondendo com um RST ao invés de enviar um pacote ACK que iria completar o cumprimento-de-três-vias e estabelecer uma conexão completa. O pacote RST é enviado pelo kernel da máquina que está executando o NMAP em resposta ao SYN/ACK inesperado, e não pelo próprio NMAP.

**-PA [listadeportas] (Ping usando TCP ACK):** O ping utilizando TCP ACK funciona da mesma forma que o ping utilizando SYN. A diferença é que nesta opção a flag TCP ACK é marcada ou invés da flag SYN. O pacote ACK finge reconhecer dados de uma conexão TCP estabelecida, quando na verdade nenhuma conexão existe de fato. Assim os hosts remotos ativos deveriam sempre responder com pacotes RST, revelando sua existência no processo.

A razão pela qual o NMAP oferece scans utilizando pacotes SYN e ACK é a maximização das chances de driblar firewalls, pois alguns firewalls estão configurados, por exemplo, para rejeitar pacotes do tipo SYN entrantes exceto aqueles destinados a serviços públicos como o site web da empresa ou servidor de correio eletrônico e nestas condições pacotes do tipo ACK teriam sucesso.

**-PU [listadeportas] (Ping usando UDP):** Nesta opção o NMAP envia um pacote UDP vazio ou de um tamanho especificado (--data-length) para as portas informadas. Caso nenhuma porta seja especificada no parâmetro [listadeportas] a porta padrão 31338 é escolhida. Uma porta alta incomum é utilizada propositalmente como padrão porque enviar para portas abertas normalmente é indesejado para este tipo particular de scan. Se o pacote chegar a uma porta fechada na máquina-alvo, a sondagem UDP deve causar um pacote ICMP de porta inalcançável como resposta e isso diz ao NMAP que a máquina está ativa e disponível.

Muitos outros tipos de erros ICMP, tais como host/rede inalcançável ou TTL excedido são indicativos de um host inativo ou inalcançável. A falta de resposta também é interpretada dessa forma. Se uma porta aberta é alcançada, a maioria dos serviços simplesmente ignoram o pacote vazio e falham ao retornar qualquer resposta, por isso uma porta pouco provável de estar em uso é escolhida. A principal vantagem de se usar o scan de Ping UDP é que ele passará por firewalls e filtros que bloqueiam apenas os pacotes TCP.

Existem ainda outras técnicas para a descoberta de hosts na rede utilizando o NMAP que não serão abordadas aqui, mas que podem ser acessadas diretamente no site [http://nmap.org/man/pt\\_BR/man-host-discovery.html](http://nmap.org/man/pt_BR/man-host-discovery.html)

## Técnicas de Escaneamento de Portas

Para cada caso de uso do NMAP podemos escolher dentre algumas técnicas de scan que melhor atende as necessidades da varredura. Em alguns casos por exemplo desejamos escanear uma rede inteira sem sermos notados, em outros casos o scan esta sendo feito pelo próprio administrador da rede para auditoria de segurança e esse detalhe não é importante, mas outras características no scan são desejadas. Para isso temos algumas técnicas de scan de portas:

**-sS (scan TCP SYN):** O scan SYN é a opção de scan padrão mais popularmente utilizada, pois pode ser executada rapidamente escaneando milhares de portas por segundo em uma rede rápida, não bloqueada por firewalls intrusivos. O scan SYN é relativamente não-obstrutivo e camuflado, uma vez que ele nunca completa uma conexão TCP. Esse tipo de scan permite uma diferenciação limpa e confiável entre os estados aberto (open), fechado (closed), e filtrado (filtered).

**-sT (scan TCP connect):** O scan TCP connect é o scan padrão do TCP quando o scan SYN não é uma opção. É uma opção para quando o usuário não tem privilégios para criar pacotes em estado bruto ou escanear redes IPv6. Ao invés de criar pacotes em estado bruto como a maioria dos outros tipos de scan fazem, o Nmap pede ao sistema operacional para estabelecer uma conexão com a máquina e porta alvos enviando uma chamada de sistema connect(). Essa é a mesma chamada de alto nível que os navegadores da web, clientes P2P, e a maioria das outras aplicações para rede utilizam para estabelecer uma conexão.

Quando um scan SYN está disponível é normalmente a melhor escolha. O Nmap tem menos controle sobre a chamada de alto nível connect() do que sobre os pacotes em estado bruto, tornando-o menos eficiente. A chamada de sistema

completa as conexões nas portas-alvo abertas ao invés de executar o reset de porta entreaberta que o scan SYN faz. Isso não só leva mais tempo e requer mais pacotes para obter a mesma informação, mas também torna mais provável que as máquinas-alvo registrem a conexão, permitindo que um sistema IDS detecte o scan.

**-sU (scan UDP):** Pelo fato do escaneamento UDP ser normalmente mais lento e mais difícil que o TCP, alguns auditores de segurança ignoram essas portas. Isso é um erro, pois serviços UDP passíveis de exploração são bastante comuns e invasores certamente não ignoram essa possibilidade de exploração.

O scan UDP é ativado com a opção `-sU` e pode ser combinado com um tipo de escaneamento TCP como o scan SYN (`-sS`) para averiguar ambos protocolos na mesma execução. Ele funciona enviando um cabeçalho UDP vazio (sem dados) para cada porta alvo. Se um erro ICMP de porta inalcançável (tipo 3, código 3) é retornado, a porta está fechada. Outros erros do tipo inalcançável (tipo 3, códigos 1, 2, 9, 10, ou 13) marcam a porta como filtrada. Ocasionalmente um serviço irá responder com um pacote UDP, provando que a porta está aberta. Se nenhuma resposta é recebida após as retransmissões, a porta é classificada como aberta|filtrada. Isso significa que a porta poderia estar aberta, ou talvez que filtros de pacotes estejam bloqueando a comunicação. Scans de versões (`-sV`) podem ser utilizados para ajudar a diferenciar as portas verdadeiramente abertas das que estão filtradas.

Um grande desafio com o escaneamento UDP é fazê-lo rapidamente. Portas abertas e filtradas raramente enviam alguma resposta, deixando o Nmap esgotar o tempo (time out) e então efetuar retransmissões para o caso de a sondagem ou a resposta ter sido perdida. Portas fechadas são, normalmente, um problema ainda maior. Elas

costumam enviar de volta um erro ICMP de porta inalcançável. Mas, ao contrário dos pacotes RST enviados pelas portas TCP fechadas em resposta a um scan SYN ou connect, muitos hosts limitam a taxa de mensagens ICMP de porta inalcançável por padrão. O Linux e o Solaris são particularmente rigorosos quanto a isso.

O Nmap detecta a limitação de taxa e diminui o ritmo de acordo para evitar inundar a rede com pacotes inúteis que a máquina-alvo irá descartar. Infelizmente, um limite como o do Linux de um pacote por segundo faz com que um scan de 65.536 portas leve mais de 18 horas.

**-sA (scan TCP ACK):** Esse tipo de scan nunca determina se uma porta está aberta (ou mesmo aberta|filtrada) e é utilizado para mapear conjuntos de regras do firewall, determinando se eles são orientados à conexão ou não e quais portas estão filtradas.

Existem ainda outras técnicas de escaneamento de portas utilizando o NMAP que não serão abordadas aqui, mas que podem ser acessadas diretamente no site [http://nmap.org/man/pt\\_BR/man-port-scanning-techniques.html](http://nmap.org/man/pt_BR/man-port-scanning-techniques.html)

## **Especificação de Portas e Ordem de Scan**

É possível dizer ao NMAP quais portas devem ser escaneadas e se a ordem é aleatória ou sequencial. Para isso é possível especificar algumas opções:

**-p <faixa de portas> (Escaneia apenas as portas especificadas):** As portas individuais a serem escaneadas devem ser passadas como parâmetro separadas por vírgula (ex: -p 137, 139, 8080) enquanto que faixas de portas a serem escaneadas devem ser separadas por hífen (ex: -p 53-137).

**-F (Scan Rápido (portas limitadas)):** Especifica que você deseja escanear apenas as portas listadas no arquivo nmap-services que vem com o NMAP (ou o arquivo de protocolos para o -sO). Tornando o scan mais rápido do que escanear todas as 65535 portas de um host.

**-r (Não usa as portas de forma aleatória):** Por padrão o NMAP escaneia as portas em uma ordem aleatória. Esta opção diz ao NMAP para escanear as portas em ordem seqüencial.

## **4.2 – Scanner de Vulnerabilidades: NESSUS**

Como vimos anteriormente, vulnerabilidades em um sistema computacional, são pontos fracos ou “brechas” que podem comprometer a segurança do sistema ou da rede. Apesar das vulnerabilidades poderem ser encontradas em diferentes áreas, vamos nos ater neste estudo especificamente a vulnerabilidades encontradas no software do sistema, descartando, portanto as demais.

O Nessus é uma ferramenta bem conhecida, utilizada para realizar varreduras em sistemas computacionais a procura de vulnerabilidades. A ferramenta permite ainda realizar auditorias remotas e determinar se a rede foi invadida ou usada de maneira indevida. Além de verificar a presença de vulnerabilidades, é possível também verificar especificações de conformidade, violações de políticas de conteúdo e outras anomalias em um computador local. O Nessus é muito utilizado também por administradores de rede e profissionais da área de segurança a fim de detectar possíveis vulnerabilidades e falhas em uma rede, antes que elas sejam exploradas por pessoas mal intencionadas.

Estas são algumas das características da ferramenta Nessus, de acordo com o guia de instalação da versão 5.0 fornecido pelo fabricante:

**Varredura inteligente:** O Nessus não gera alarmes falsos, ou seja, o programa não pressupõe que um determinado serviço está sendo executado em uma porta fixa. O programa verifica uma vulnerabilidade por meio de exploração sempre que possível. Nos casos em que isso não for confiável ou afetar negativamente o alvo, o Nessus conta com um banner de servidor para determinar a presença da vulnerabilidade. Nesse caso, o relatório gerado indicará se esse método foi utilizado.

**Arquitetura modular:** A arquitetura cliente/servidor permite instalar o scanner (servidor) e conectar-se à interface gráfica do usuário (cliente) por intermédio de um navegador.

**Compatível com CVE:** A maioria dos plugins se conecta ao CVE para que os administradores possam recuperar mais informações sobre vulnerabilidades publicadas. As referências ao Bugtraq (BID), OSVDB e alertas de segurança dos fornecedores também são incorporadas com frequência.

**Arquitetura de plugin:** Os testes de segurança são realizados por meio de plugins externos. Dessa forma, é possível adicionar facilmente novos testes de criação própria ou selecionar plugins específicos existentes. A lista completa dos plugins do Nessus está disponível em <http://www.nessus.org/plugins/index.php?view=all>.

**NASL:** O scanner Nessus utiliza NASL (Nessus Attack Scripting Language), uma linguagem criada especificamente para a criação de testes de segurança.

**Banco de dados de vulnerabilidades de segurança atualizado:** É possível consultar as verificações de segurança mais recentes no link <http://www.nessus.org/scripts.php>.

**Compatibilidade entre plugins:** Os testes de segurança realizados pelos plugins do Nessus impedem que sejam realizadas verificações desnecessárias. Se o servidor de FTP não permitir logins anônimos, não serão realizadas verificações de segurança relacionadas a logins anônimos.

**Relatórios completos:** Além de detectar as vulnerabilidades de segurança existentes na rede e o nível de risco de cada uma delas (baixo, médio, alto e grave), o Nessus oferece soluções de como atenuá-las.

**Suporte total a SSL:** O Nessus é capaz de testar os serviços oferecidos por SSL, como HTTPS, SMTPS, IMAPS entre outros.

**Testes não destrutivos (opcional):** O Nessus permite ativar uma opção chamada “safe checks” para que desta forma, verificações que possam causar danos ou interrupções na rede sejam executadas através do uso de banners ao invés de explorar a falha.

### **Políticas de Segurança no Nessus**

Juntamente com o Nessus virão predefinidas algumas políticas criadas pela Tenable Network Security, Inc. Essas políticas são fornecidas como modelos para ajudar na criação de políticas mais específicas ou para serem usadas sem modificações para varreduras básicas dos seus recursos.



- **Varredura de rede externa:** Esta política foi projetada para a verificação de hosts externos e que normalmente apresentam menos serviços à rede. Os plugins relacionados a vulnerabilidades conhecidas de aplicativos da Web (as famílias de plugins CGI Abuses e CGI Abuses: XSS) são ativados com a aplicação desta política. Além disso, todas as 65.535 portas são verificadas em cada alvo.
- **Varredura de rede interna:** Esta política foi projetada levando-se em conta a melhoria do desempenho, pois pode ser usada para verificar redes internas de grande porte com muitos hosts, vários serviços expostos e sistemas incorporados, como impressoras. Os plugins “CGI Abuse” não estão ativados e um conjunto de portas padrão é examinado, mas não todas as 65.535.
- **Testes de aplicativos da Web:** Esta política de varredura é utilizada para verificar os sistemas fazendo com que o Nessus detecte vulnerabilidades conhecidas e desconhecidas nos aplicativos da Web. Os recursos de “difusão” do Nessus são ativados com esta política, o que fará com que o Nessus detecte todos os sites descobertos e verifique as vulnerabilidades presentes em cada um dos parâmetros, incluindo XSS, SQL, injeção de comandos e vários outros.
- **Preparar para auditorias de PCI DSS:** Esta política ativa as verificações de conformidade com a norma PCI DSS integradas, compara os resultados das varreduras aos padrões PCI e gera um relatório sobre o comportamento da conformidade. É importante observar que uma varredura de compatibilidade bem-sucedida não garante a conformidade nem uma infra-estrutura segura. As organizações que estejam se preparando para uma avaliação da PCI DSS

podem usar essa política para preparar suas redes e seus sistemas para a conformidade com a PCI DSS.

### **Opções de Varredura**

O Nessus permite que a varredura (scan) seja feita de diferentes formas:

- **Save Knowledge Base:** O scanner Nessus salva as informações de varredura no banco de dados de conhecimento do servidor Nessus para uso posterior. Isso inclui portas abertas, plugins utilizados, serviços descobertos etc.
- **Safe Checks:** A opção Safe Checks (Verificações Seguras) desativa todos os plugins que podem afetar negativamente o host remoto.
- **Silent Dependencies:** Se esta opção for selecionada, a lista de dependências não será incluída no relatório.
- **Log Scan Details to Server:** Salva detalhes adicionais da varredura no log do servidor Nessus (nessusd.messages), incluindo a ativação ou encerramento do plugin ou se um plugin foi interrompido. O log resultante pode ser usado para confirmar se determinados plugins foram usados e se os hosts foram examinados.
- **Stop Host Scan on Disconnect:** Ao marcar esta opção, o Nessus cessará a varredura se detectar que o host parou de responder. Isso pode ocorrer se os usuários desligarem seus PCs durante uma varredura, se um host parar de responder ou se o mecanismo de segurança (por exemplo: IDS) bloqueou o tráfego para um servidor. Se as varreduras continuarem nesses computadores, o tráfego desnecessário será enviado e atrasará a verificação.

- **Avoid Sequential Scans:** Normalmente, o Nessus verifica uma lista de endereços IP em sequência. Se esta opção estiver marcada, o Nessus verificará a lista de hosts em ordem aleatória. Isto pode ser útil para ajudar a distribuir o tráfego de rede direcionado a uma sub-rede específica durante varreduras extensas.
- **Consider Unscanned Ports as Closed:** Se uma porta não for examinada com um scanner de porta selecionado (por exemplo: fora do intervalo especificado), será considerada fechada pelo Nessus.
- **Designate Hosts by their DNS Name:** Deve-se usar o nome do host em vez do endereço IP na impressão do relatório.

Há também opções de controle de como a varredura deve ser feita pelo Nessus de acordo com a rede a ser verificada:

- **Reduce Parallel Connections on Congestion:** Permite que o Nessus detecte o envio de um grande número de pacotes e quando o pipe da rede atingir a capacidade máxima, o Nessus reduzirá a velocidade da varredura ao nível adequado para diminuir o congestionamento. Ao diminuir o congestionamento, o Nessus tentará reutilizar o espaço disponível no pipe da rede automaticamente.
- **Use Kernel Congestion Detection (Linux Only):** Permite que o Nessus monitore a CPU e outros mecanismos internos em caso de congestionamento e diminua o ritmo de maneira proporcional. O Nessus tentará usar sempre o máximo de recursos disponíveis. Este recurso está disponível apenas para os scanners Nessus instalados em Linux.

Além destas opções disponíveis para varredura no Nessus existem ainda outras opções disponíveis para o método de varredura a ser realizado, intervalo de portas a serem varridas e opções de performance da varredura, que podem ser conferidas no guia do usuário do Nessus.

## **Plugins**

O Nessus realiza os testes de segurança nos alvos através do uso de plugins e oferece plugins agrupados em famílias para que o usuário possa escolher plugins individuais específicos ou até famílias inteiras para incluir na varredura. Desta forma é possível criar seleções diferentes de plugins a serem utilizados para cada varredura, criando novas políticas de varredura.

Novas vulnerabilidades são descobertas diariamente e com elas novos plugins surgem para atender às novas necessidades de varredura. A arquitetura de plugins permite ao usuário do Nessus de forma fácil e rápida incluir ou excluir novos testes de segurança em sua varredura. Caso um plugin específico não esteja disponível, é possível baixá-lo da internet e anexá-lo à varredura.

## **Relatórios**

Após realizar a varredura em uma rede, o Nessus organiza as informações coletadas durante a varredura e as apresenta ao usuário em um relatório detalhado e organizado.

A tela “Reports” (Relatórios) funciona como um ponto central para exibir, comparar, enviar e baixar resultados de varreduras.

No relatório é possível selecionar hosts em uma lista para ver mais informações sobre o host, como o número de portas abertas, o protocolo e serviços utilizados em

cada porta, informações do sistema operacional, além de um resumo das vulnerabilidades encontradas categorizadas pelo grau de risco.

O Nessus irá indicar ainda, para cada vulnerabilidade listada no relatório uma ação para corrigir o problema, além de listar uma sinopse, uma descrição técnica, o fator de risco, a pontuação CVSS, resultados relevantes que demonstram a conclusão, referências externas, data de publicação da vulnerabilidade, data de publicação/modificação do plugin e disponibilidade da exploração.

### **4.3 – Detecção de Intrusão (SNORT)**

### **5.1 – O que é virtualização**

Segundo Gregg (2008, pág. 47) “Virtualização é o processo de emular hardware dentro de uma máquina virtual.” O processo de emulação de hardware duplica a arquitetura física necessária na máquina para o funcionamento dos programas e processos.

A empresa VMWare (2012) explica o funcionamento da virtualização como: “A virtualização permite executar várias máquinas virtuais em uma única máquina física, com cada VM compartilhando os recursos desse computador físico em vários ambientes. Máquinas virtuais diferentes conseguem executar sistemas operacionais diferentes e vários aplicativos no mesmo computador físico.”

De acordo com Laureano e Maziero (2008, pág. 6), diferentes processadores possuem diferentes conjuntos de instruções, o que faz com que cada arquitetura

necessite de um software diferente capaz de interpretar suas instruções. É por este motivo que não é possível executar em um processador Intel/AMD uma aplicação compilada para um processador ARM. As instruções em linguagem de máquina escritas para o processador ARM não serão compreendidas pelo processador Intel/AMD. Similarmente não é possível executar em Linux uma aplicação escrita para Windows, pois as chamadas de sistema são diferentes para os dois sistemas operacionais.

Todavia, é possível contornar esses problemas de compatibilidade através de uma camada de virtualização construída em software. Usando os serviços oferecidos por uma determinada interface de sistema, é possível construir uma camada de software que ofereça aos demais componentes uma outra interface. Essa camada de software permite o acoplamento entre interfaces distintas, de forma que um programa desenvolvido para a plataforma A possa ser executado sobre uma plataforma distinta B. (LAUREANO; MAZIERO, 2008, pág. 6)

Usando os serviços oferecidos por uma determinada interface de sistema, a camada de virtualização constrói outra interface de mesmo nível, de acordo com as necessidades dos componentes de sistema que farão uso dela. A nova interface de sistema, vista através dessa camada de virtualização, é denominada máquina virtual. (LAUREANO; MAZIERO, 2008, pág. 6)

A virtualização foi amplamente adotada pela IBM nos anos 60 para permitir que um único mainframe pudesse comportar múltiplos sistemas operacionais com seus diferentes usuários. Em 1990 a virtualização volta a ganhar visibilidade e em 2000 passa a ser largamente utilizada na emulação de servidores, permitindo o funcionamento de vários servidores em poucas máquinas físicas. Desta forma

reduziram-se os custos com os equipamentos utilizados e aumentaram a escalabilidade e flexibilidade dos servidores. A virtualização ainda encontrou uso no desenvolvimento de software, com a finalidade de testar novos software em diferentes sistemas operacionais e plataformas de hardware. Além disso a tecnologia de virtualização encontrou uma crescente aplicação como componente chave dos modelos de cloud computing e no ensino a distância (NANCE; HAY; DODGE; SEAZZU; BURD, 2009, pág. 5).

Inúmeros programas podem ser usados na virtualização de sistemas, tais como VMWare, Virtual PC, Open VZ, entre outros. Dependendo da quantidade de recursos de hardware disponíveis na máquina do usuário, como espaço de armazenamento, memória RAM, capacidade de processamento, é possível que diversos destes programas emulem sistemas virtuais ao mesmo tempo de forma independente, sem que um interfira no funcionamento do outro. Sistemas virtuais de modo geral, emulam a maior parte dos recursos de hardware necessitados pelos sistemas operacionais para o seu funcionamento. O computador físico que está hospedando uma máquina virtual dedica parte dos seus recursos de hardware ao funcionamento do sistema virtual. O disco rígido de uma máquina virtual nada mais é do que um grande arquivo na máquina hospedeira. Outros recursos como processamento e memória RAM da máquina virtual também são compartilhados da máquina hospedeira.

## **5.2 – Vantagens do uso de virtualização no ensino de segurança**

De acordo com Nance, Hay, Dodge, Seazzu e Burd (2009, pág. 4): A virtualização nos permite a emulação de uma rede inteira de computadores e dos softwares neles

instalados em uma única máquina física. Os computadores virtuais (ou máquinas virtuais) funcionam e se comportam exatamente como se estivessem rodando em um hardware real em uma máquina física. Máquinas virtuais podem ser configuradas para se conectarem a outras redes virtuais isoladas, de modo a nos permitirem a execução de testes de segurança em redes emuladas sem comprometer e afetar o desempenho de uma rede real.

Segundo Gregg (2009, pág. 48) o uso de uma VMware seria uma boa escolha para laboratórios, pois permite testar com facilidade ferramentas de segurança, testar upgrades e estudos de exames de certificação.

O uso e aplicação da virtualização pode se estender a diversas áreas e não se limitar apenas a pesquisa e ensino de segurança computacional. Máquinas e redes virtuais podem ser utilizadas em outros campos da ciência da computação como por exemplo na implementação de algoritmos de clusters sem a necessidade de múltiplos sistemas físicos.

As configurações das máquinas virtuais ficam armazenadas em arquivos na máquina hospedeiras, permitindo que com facilidade estes arquivos sejam copiados a fim de clonar a máquina virtual. Essa prática pode ser muito útil para a criação e restauração de backups durante testes com a máquina virtual, oferecendo mais flexibilidade para testes que podem comprometer a máquina virtual durante o aprendizado do aluno.

Diferentes máquinas virtuais podem se comunicar em uma máquina hospedeira através de uma rede virtual e podem ainda se comunicar com a rede mundial de computadores utilizando a interface de rede da máquina hospedeira. Em ambientes de teste e educação, pode ser interessante desconectar as máquinas virtuais da



internet a fim de se criar uma rede isolada para experimentação. Essa configuração é particularmente importante quando se usa as máquinas virtuais e suas redes virtuais para testes perigosos, como a experimentação de infecções por vírus. Assim as máquinas virtuais apresentam a vantagem de isolamento, não colocando em risco a máquina hospedeira, sua rede, nem outros usuários da internet (NANCE; HAY; DODGE; SEAZZU; BURD, 2009, pág. 5).

Uma outra vantagem de se usar virtualização no ensino de segurança é que no decorrer da aula, pode ser interessante o uso de diferentes sistemas operacionais para a realização dos testes. Desta forma, a virtualização do ambiente a ser utilizado facilita ao aluno ter em uma única máquina hospedeira diferentes sistemas operacionais em funcionamento, sem a necessidade de ficar reiniciando o computador para realizar tais testes. Além disso, a realização dos testes em máquinas virtuais permite que a máquina hospedeira mantenha-se livre das influências ocasionadas pelos testes.

Para Belapurkar; Chakrabarti; Ponnappalli e Varadarajan (2009, pág. 297) a virtualização tenha talvez o seu maior papel na segurança de desktops. Enquanto a tecnologia de virtualização está avançando rapidamente através da inovação de hardware e software no mercado, boa parte dela está focada em servidores. A medida que a tecnologia de virtualização se torna amigável para desktops, há muito o que se ganhar em termos de segurança dos desktops. Potencialmente poderiam existir domínios separados para rodarem aplicações seguras e não seguras, evitando vários problemas de segurança e privacidade nos hosts. Entretanto, algumas políticas de segurança (firewalls, níveis de isolamento e acesso de dispositivos) podem ser centralizadas e reforçadas à nível da máquina física,

deixando a máquina virtual do host completamente no controle do usuário, sem nenhuma implicação de segurança.

Para Vacca (2009, pág. 699) um dos benefícios da virtualização é a segurança proporcionada pelo isolamento da máquina virtual. Por outro lado, alguns códigos maliciosos passaram a adquirir a habilidade de detectar se estão sendo executados dentro de um ambiente virtual ou não. Pesquisas mostraram que códigos maliciosos podem escapar do ambiente virtual e atacar a máquina hospedeira ou ao menos roubar informações dela. Em resposta a tais fatos, os desenvolvedores da área de segurança estão adaptando novas características e funcionalidades na virtualização de sistemas para corrigir as vulnerabilidades e detectar tais ataques.

Apesar da possibilidade de ataques à máquina hospedeira pela utilização de máquinas virtuais, ambientes virtualizados são considerados seguros e são recomendados por diversos autores como ferramenta para o ensino de segurança e outras atividades.

### **5.3 – A emulação do ataque**

De acordo com Harris, Harper, Eagle, Ness e Lester (2005, pág. 73) o ato de varrer, sondar e explorar redes a procura de vulnerabilidades que podem comprometer a rede ou os seus hosts é chamado de teste de penetração. O primeiro objetivo de um teste de penetração é se apropriar da rede, o segundo objetivo é se apropriar da rede de quantas formas diferentes você conseguir, para que então seja possível apontar cada falha encontrada ao consumidor ou usuários da rede. O teste de penetração em uma rede é uma excelente forma de testar a eficácia das medidas de segurança de uma organização e de expor as falhas de segurança da rede.

Uma vez que administradores de redes, engenheiros e profissionais de segurança entendam como os atacantes agem, então, eles podem emular suas atividades para simular um teste de penetração útil. A emulação de um ataque é a única forma confiável de testar um nível de segurança de um ambiente e como este ambiente irá reagir quando estiver sendo atacado por um ataque real (HARRIS, HARPER, EAGLE, NESS, LESTER, 2005, pág. 15).

#### **5.4 – A construção do ambiente virtual**

Para que os testes realizados em nossos ambientes virtuais tenham uma maior abrangência dos diferentes tipos de problemas encontrados e em diferentes tipos de sistemas operacionais, utilizaremos máquinas virtuais com o sistema operacional Linux (Ubuntu) e também máquinas virtuais com o sistema operacional Windows XP.

Para criar-se um ambiente controlado é importante começar a sua criação do zero, pois ambientes prontos podem conter configurações indesejadas e ou até problemas pré existentes não resolvidos que podem influenciar e atrapalhar os testes a serem realizados.

Gregg (2008, pág. 16) ressalta: “Problemas antigos serão herdados devido aos erros ou equívocos de usuários anteriores. E a menos que você instale e configure corretamente as coisas do estado danoso, você nunca poderá ter plena certeza de como exatamente tudo foi configurado.”

No nosso caso, a instalação dos sistemas operacionais nas máquinas virtuais foram realizadas do zero para garantir a não existência de nenhuma configuração realizada anteriormente.

A escolha do sistema operacional Windows XP para uma das máquinas virtuais justifica-se por ser um dos sistemas Windows mais utilizados durante um maior período de tempo por grande parte da população. Segundo dados coletados da Wikipedia por Gregg (2008, pág. 32) a Microsoft vendeu mais de 1 milhão de cópias do Windows XP em 2006. Já para a escolha da distribuição Ubuntu como sistema operacional Linux, destacamos a crescente utilização e popularidade desta distribuição além da velocidade com que a comunidade Ubuntu avança em seu desenvolvimento.

Para o sistema Windows XP nenhum mecanismo de defesa tais como antivírus ou firewalls adicionais serão instalados, pois podem interferir no resultado dos testes a serem realizados, identificando uma ameaça a qual estamos simulando propositalmente para nossos estudos. Apenas o firewall incluso no Windows XP será utilizado como parte dos testes.

Para o processo de virtualização, iremos utilizar o software VMware em sua versão 4.0.2 build-591240. E na construção das máquinas virtuais iremos utilizar as configurações padrões sugeridas pelo VMware, exceto pela configuração de memória RAM, a qual iremos configurá-la para 512MB em todas as máquinas, pois ela é um fator limitante para o número de máquinas virtuais que conseguimos executar simultaneamente.

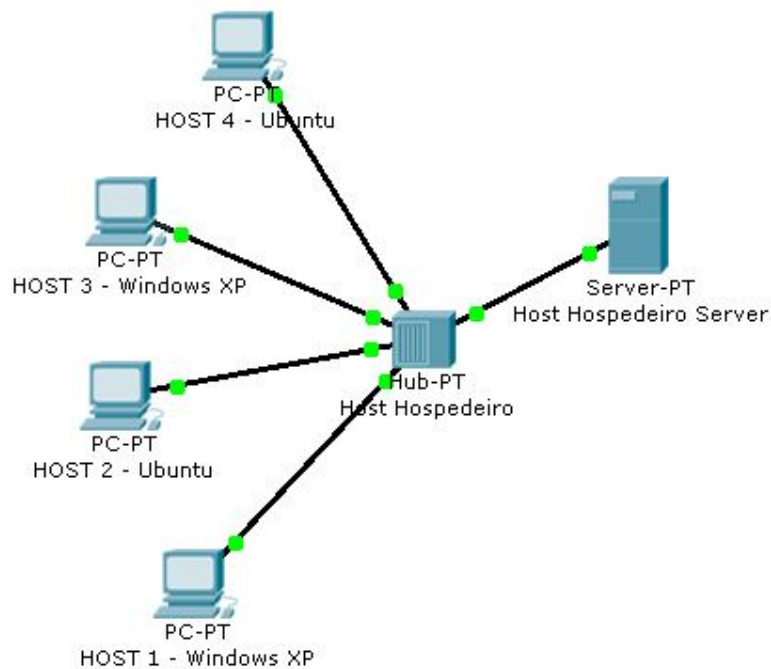


Para a simulação de uma rede de computadores nos quais efetuaremos os testes, iremos utilizar algumas instâncias das máquinas virtuais criadas. Assim, teremos como base 2 máquinas virtuais distintas, uma com o sistema Windows XP e outra com o sistema Linux Ubuntu e a partir destas faremos quantas cópias forem necessárias e puderem ser suportadas pelo host hospedeiro das máquinas virtuais.

As configurações de cada instância de máquina virtual copiada serão as mesmas das máquinas utilizadas como base, mas a medida que estas forem sofrendo alterações, cada cópia da máquina virtual irá manter suas novas configurações.

Basicamente iremos criar uma rede virtual formada por máquinas virtuais em nosso host hospedeiro. O VMware irá automaticamente alocar as máquinas virtuais

inicializadas em uma faixa de IPs, de modo que todas as máquinas virtuais possam se “enxergar” na rede. Supondo que iremos executar 4 cópias de máquinas virtuais, 2 de cada sistema operacional, a topologia da nossa rede virtual ficará assim:



## 6.1 – Scanner de Rede Utilizando o NMAP

### Objetivos

- Descobrir os hosts ativos da rede
- Varrer os hosts descobertos a procura de portas abertas
- Realizar as ações anteriores de diferentes formas em diferentes contextos

### Pré-Requisitos

- Leitura da sessão 4.1 Scanner de Rede: NMAP

- Execução simultânea de pelo menos 1 máquina virtual Windows e 1 máquina virtual Ubuntu para a realização dos testes.

Neste roteiro iremos focar nas principais funções do NMAP que é a verredura de portas. Apesar de o NMAP também servir para outras funcionalidades como a detecção de versões e tipos de sistemas operacionais, não iremos abordar estas práticas neste roteiro, pois existem outras ferramentas que também desempenham essas funções como o Nessus (scan de vulnerabilidades) no qual iremos tratar deste assunto no capítulo 6.2 intitulado Scanner de Vulnerabilidades: NESSUS.

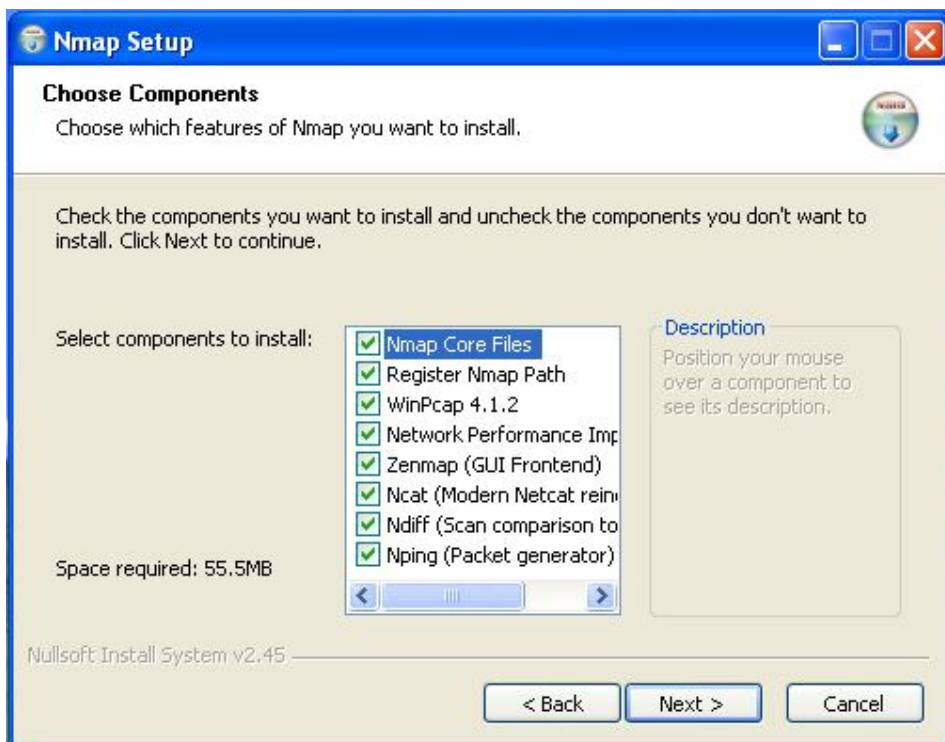
No ambiente virtual o qual iremos testar o NMAP, a ferramenta já encontra-se instalada em sua versão 5.21. Entretanto para instalá-lo no Ubuntu, basta utilizarmos o terminal e instalarmos o pacote do nmap:

**sudo apt-get install nmap**

Se preferir, você ainda pode instalar uma interface gráfica para utilizar o NMAP chamada ZenMap, através do comando:

**sudo apt-get install zenmap**

Para instalar o Nmap no Windows, basta baixar o executável no site <http://nmap.org/download.html> e seguir os passos de instalação padrões, deixando marcado para a instalação todos os componentes que acompanham o instalador:



Para este roteiro, iremos utilizar máquinas virtuais com ambos os sistemas operacionais na rede (Ubuntu e Windows XP) e para realizar as varreduras iremos utilizar a interface gráfica Zenmap, a qual é muito similar tanto no Linux quanto no Windows e para alguns casos o terminal de comandos do Ubuntu.

Ambos ambientes virtuais (Ubuntu e Windows XP) encontram-se propositalmente com algumas portas abertas, para que possamos varrê-las e detectá-las em nossos testes.

### Tarefa 1: A descoberta de hosts

Antes de iniciarmos a nossa varredura, é preciso definir o nosso alvo. Como vimos na sessão 4.1 do Nmap, é possível varrer um host individualmente ou uma rede inteira.

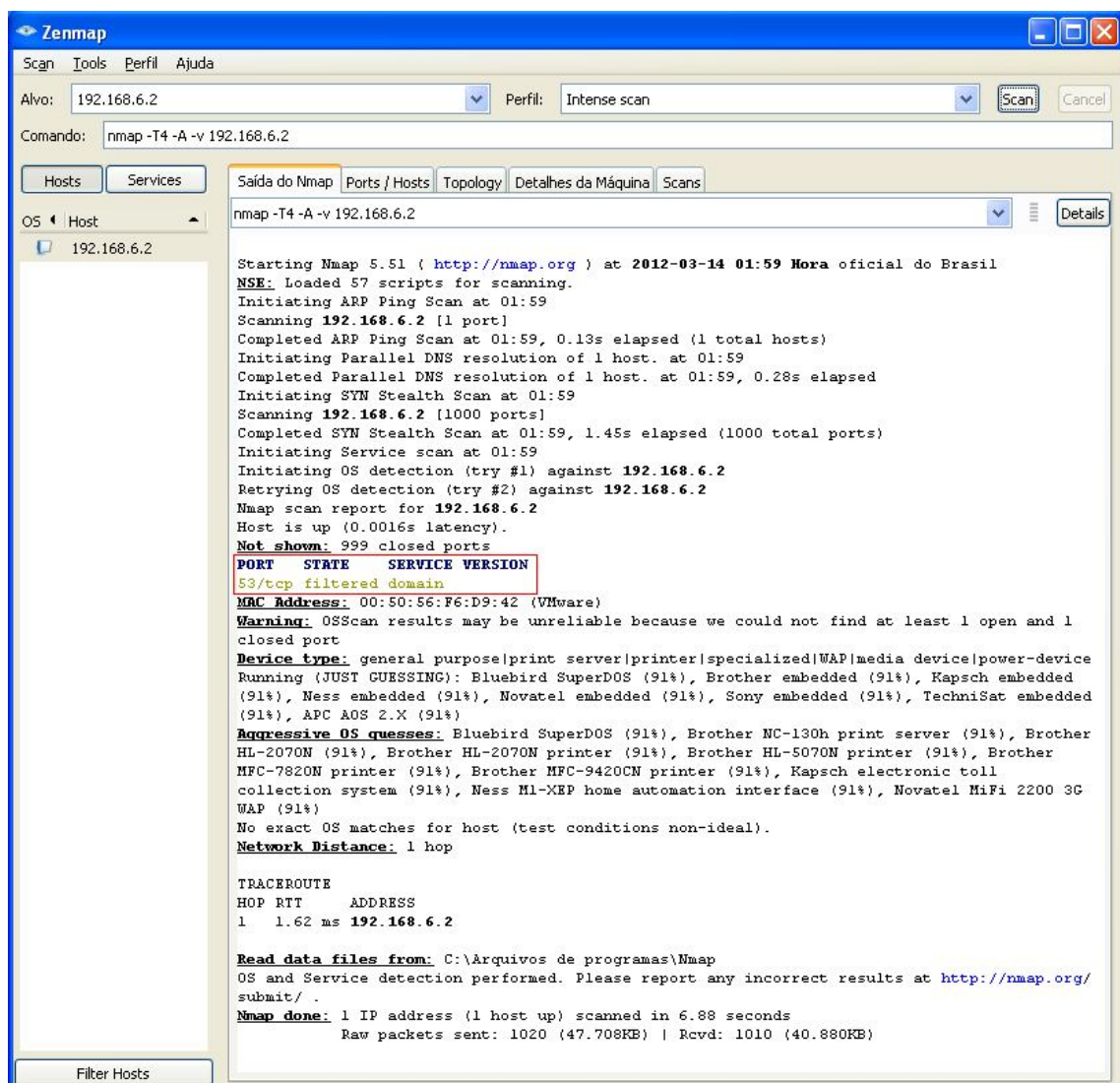


**Situação 1: Suponha que você conheça o endereço IP do alvo que deseja varrer em sua rede. Este endereço IP é o do gateway da rede, no nosso caso 192.168.6.2.**

Nesta situação vamos disparar uma varredura diretamente contra o alvo específico 192.168.6.2. Por enquanto não vamos nos preocupar com as opções de varredura.

Podemos realizar tal varredura simplesmente digitando o endereço alvo no campo “Alvo” da interface Zenmap.

A figura abaixo mostra o resultado da nossa varredura:



Como podemos observar na figura, a saída do Nmap mostra na tabela de portas que a porta de número 53 utiliza o protocolo TCP, encontra-se no estado “filtrada” (isso significa que existe um filtro, firewall ou algum outro obstáculo na rede bloqueando a porta de forma que o Nmap não consegue determinar se a porta está aberta ou fechada.) e o serviço que está ouvindo nesta porta é um servidor de domínios (“domain” como aparece o nome do serviço).

Escolhemos justamente o gateway da nossa rede virtual para varrer e assim sabemos que a porta está aberta, pois sendo um serviço de DNS que está ouvindo nesta porta, ele deve estar sempre atento para o caso de algum IP solicitar alguma tradução de endereços.

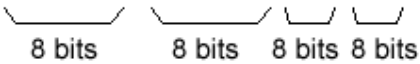
**Situação 2: Executando o número máximo de máquinas virtuais na rede que você puder, suponha que você desconheça o número de máquinas existentes na rede e os seus endereços IPs. Suponha também que a rede é de sua propriedade e que não há problema nenhum varrer a rede a procurar de hosts ativos.**

Nesta situação iremos disparar uma busca de hosts por toda a faixa de IPs em que as máquinas virtuais podem estar alocadas. Para isso iremos utilizar outro tipo de endereçamento, chamado de CIDR. Neste tipo de endereçamento, iremos passar ao Nmap um endereço base seguido do número de bits que o Nmap deve manter no início do endereço, variando os outros bits.

As nossas máquinas virtuais estarão sempre alocadas pelo VMware no intervalo de IP de 192.168.6.0 até 192.168.6.255. Desta forma iremos manter sempre o endereço base 192.168.6.X e variar o X para a nossa varredura. Como cada posição

no endereço IP ocupa 8 bits no endereçamento, então o nosso X final irá ocupar também 8 bits.

192.168.6.X



8 bits 8 bits 8 bits 8 bits

Desta forma o endereço o qual iremos varrer será: **192.168.6.1/24**

O último número digitado (1) será ignorado pelo Nmap uma vez que ele irá realizar uma varredura para cada combinação dos últimos 8 bits possíveis. O número 24 indica que iremos fixar os primeiros 24 bits e variar o restante de um total de 32 bits.

Uma forma alternativa de se especificar uma faixa de endereços é através do uso do traço (-) Ex: 192.168.6.1-255.

Agora que sabemos qual o intervalo de hosts devemos varrer, basta pedir para o Nmap varrer todas as portas de todos os hosts, certo?

Como nós pretendemos fazer isso de forma eficiente, evitando um grande desperdício de tempo a resposta é: Errado. Ainda não temos idéia de quantos hosts existem na rede e destes existentes quantos deles estarão ativos no momento em que realizaremos a varredura. Pode ser interessante saber o total de hosts na rede e podemos economizar recursos e tempo varrendo somente os hosts que estão ativos neste momento.

Como nesta situação estamos fazendo uma varredura em uma rede de nossa propriedade e não temos problemas quanto à identificação de nossa varredura, utilizaremos o tipo de varredura por ping para descoberta de hosts que pode ser facilmente detectado.

Para isso vamos digitar no campo “Comando”: **nmap -sP -v 192.168.6.1/24**

Este é o comando para varrer com o Nmap utilizando a opção **-sP** (Scan usando ping) na faixa de endereços CIDR de 192.168.6.0 até 192.168.6.255. A opção **-v** faz com que o Nmap mostre mais informações sobre o scan, então a partir de agora utilizaremos ela sempre.

Ao final do scan o Nmap deve listar o estado de todos os hosts da rede e é natural que muitos deles estejam inativos, pois o número de máquinas virtuais que podemos executar simultaneamente não é muito grande.

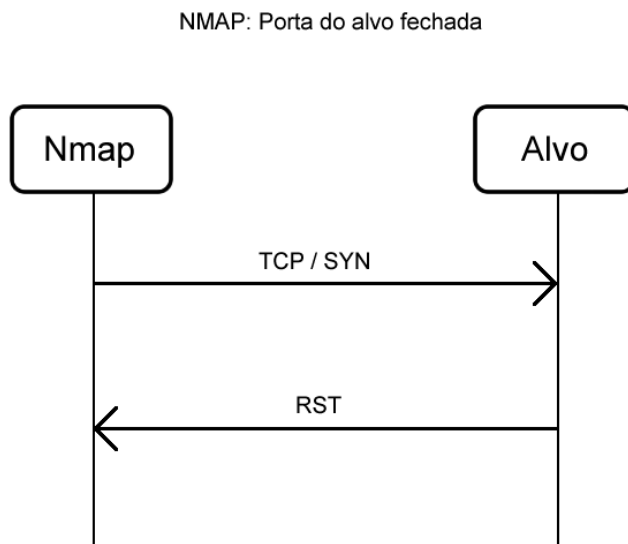
```
Nmap scan report for 192.168.6.132 [host down]
Nmap scan report for 192.168.6.133 [host down]
Nmap scan report for 192.168.6.134
Host is up (0.00087s latency).
MAC Address: 00:0C:29:CC:F2:FA (VMware)
Nmap scan report for 192.168.6.135 [host down]
Nmap scan report for 192.168.6.136 [host down]
Nmap scan report for 192.168.6.137 [host down]
Nmap scan report for 192.168.6.138 [host down]
```

**Situação 3: Tome por base a situação 2, mas desta vez, suponha que a rede não é de sua propriedade e que você deve evitar ao máximo levantar suspeitas de que a rede está sendo varrida.**

Neste caso, não podemos simplesmente disparar uma varredura pela rede utilizando a opção **-sP** como utilizamos na situação anterior, porque o fato de ela usar o “ping” convencional para a descoberta de hosts torna esta opção facilmente detectável por um firewall da rede por exemplo.

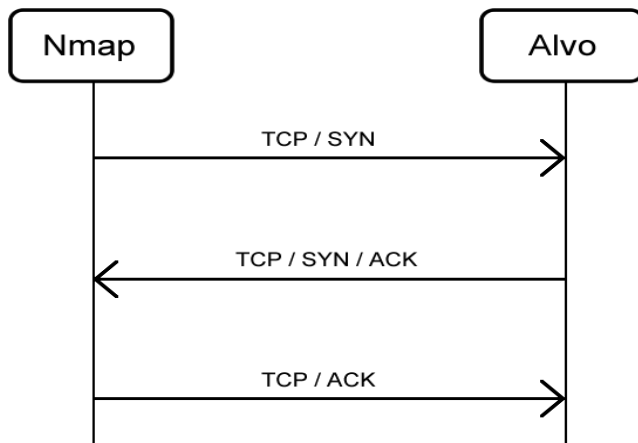
Para resolver este problema, o Nmap nos oferece algumas opções diferenciadas para o uso do Ping. Nestas opções o Nmap utiliza-se da estratégia de não completar a conexão com o host alvo para evitar a detecção.

Por exemplo na opção **-PS** pacotes TCP vazios são enviados com a flag SYN marcada para uma porta do host alvo. A flag SYN indica aos sistemas remotos que você está tentando estabelecer uma comunicação. Caso esta porta esteja fechada vamos receber um pacote RST de volta, indicando o fim do processo de “handshake” como mostra a figura abaixo:



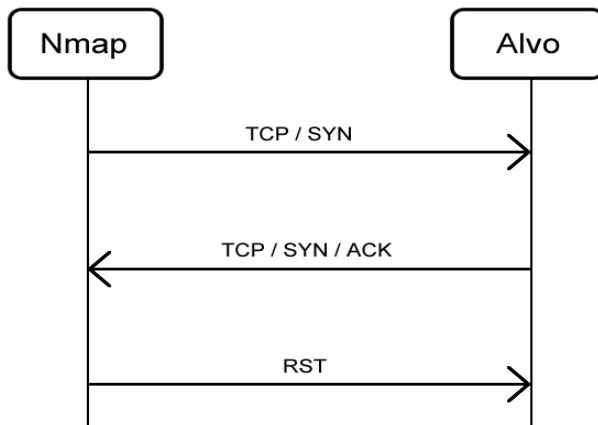
Caso a porta esteja aberta, vamos receber um pacote TCP SYN ACK e teoricamente deveríamos responder com um pacote TCP ACK para finalmente completar o “handshake” de 3 vias e estabelecermos uma conexão com o alvo:

NMAP: Conexão estabelecida com a porta do alvo



Entretanto se respondermos com um pacote TCP ACK e estabelecermos uma conexão com o host alvo, iremos expor a nossa conexão, podendo levar a detecção da nossa varredura. Para isso o Nmap irá responder com um pacote RST resetando a conexão e impedindo que ela seja completada. Neste ponto, o Nmap já capturou o pacote TCP SYN ACK e já sabe que a porta está aberta, assim o fato de derrubar a conexão imediatamente após receber o pacote, implica em não sermos descobertos mas não implica em perdas de informações para o Nmap.

NMAP: Conexão não estabelecida com o alvo



Faça o teste varrendo a rede com a opção **-PS** utilizando o comando:

**nmap -PS -v 192.168.6.1/24**

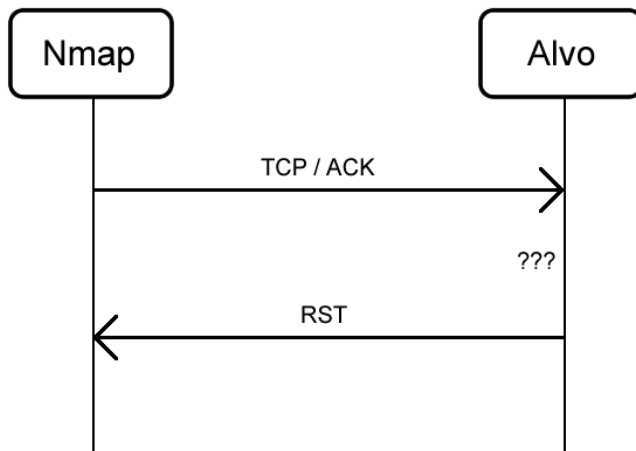
Veja na saída do Nmap que agora ele está utilizando a técnica de varredura por SYN:

```
Initiating Parallel DNS resolution of 1 host. at 16:15
Completed Parallel DNS resolution of 1 host. at 16:15, 0.16s elapsed
Initiating SYN Stealth Scan at 16:15
Scanning 2 hosts [1000 ports/host]
Completed SYN Stealth Scan against 192.168.6.2 in 1.39s (1 host left)
Discovered open port 902/tcp on 192.168.6.1
Discovered open port 912/tcp on 192.168.6.1
Completed SYN Stealth Scan at 16:15, 4.75s elapsed (2000 total ports)
Nmap scan report for 192.168.6.1
Host is up (0.0094s latency).
Not shown: 998 filtered ports
PORT      STATE SERVICE
902/tcp   open  iss-realsecure
912/tcp   open  apex-mesh
MAC Address: 00:50:56:C0:00:08 (VMware)
```

Similarmente a esta opção, o Nmap oferece também a opção **-PA** que ao invés de enviar pacotes do tipo TCP SYN, envia pacotes do tipo TCP ACK.

Neste contexto os pacotes do tipo TCP ACK indicam uma confirmação do recebimento de dados de uma conexão TCP estabelecida, quando na verdade não existe nenhuma conexão estabelecida com o alvo. Desta forma o alvo deve responder ao pacote TCP ACK com um pacote RST para encerrar a conexão, entretanto esta ação revela a existência do alvo caso ele exista e esteja ativo na rede.

NMAP: Alvo não reconhece conexão TCP



Faça o teste varrendo a rede com o comando:

**nmap -PA -v 192.168.6.1/24**

Os resultados devem ser similares aos da varredura pela opção -PS. Entretanto em alguns casos, a existência de um firewall bloqueando pacotes do tipo SYN, mas aceitando pacotes do tipo ACK pode fazer os resultados serem mais claros na opção -PA.

**Situação 4: Suponha que você queira descobrir os hosts da rede e você sabe que podem existir firewalls bloqueando o caminho. Caso os pacotes do Nmap encontrem um firewall, a resposta pode ser comprometida e não condizer com a realidade. Sabendo disso, você precisa maximizar as chances de obter bons resultados.**

Já vimos aqui que a opção -PA pode ser uma alternativa a opção -PS para driblar algum firewall que esteja bloqueando pacotes do tipo TCP SYN. Entretanto ainda pode ser provável que pacotes do tipo TCP ACK enviados por engano (o que no



nosso caso é proposital) sejam ignorados e despachados. Para aumentar ainda mais as chances de detectar os hosts da nossa rede, podemos utilizar ainda a opção de pacotes UDP. Alguns firewalls estão programados apenas para ignorarem e descartarem pacotes TCP mas não pacotes UDP. Desta forma a opção `-PU` pode ser uma alternativa para a nossa detecção.

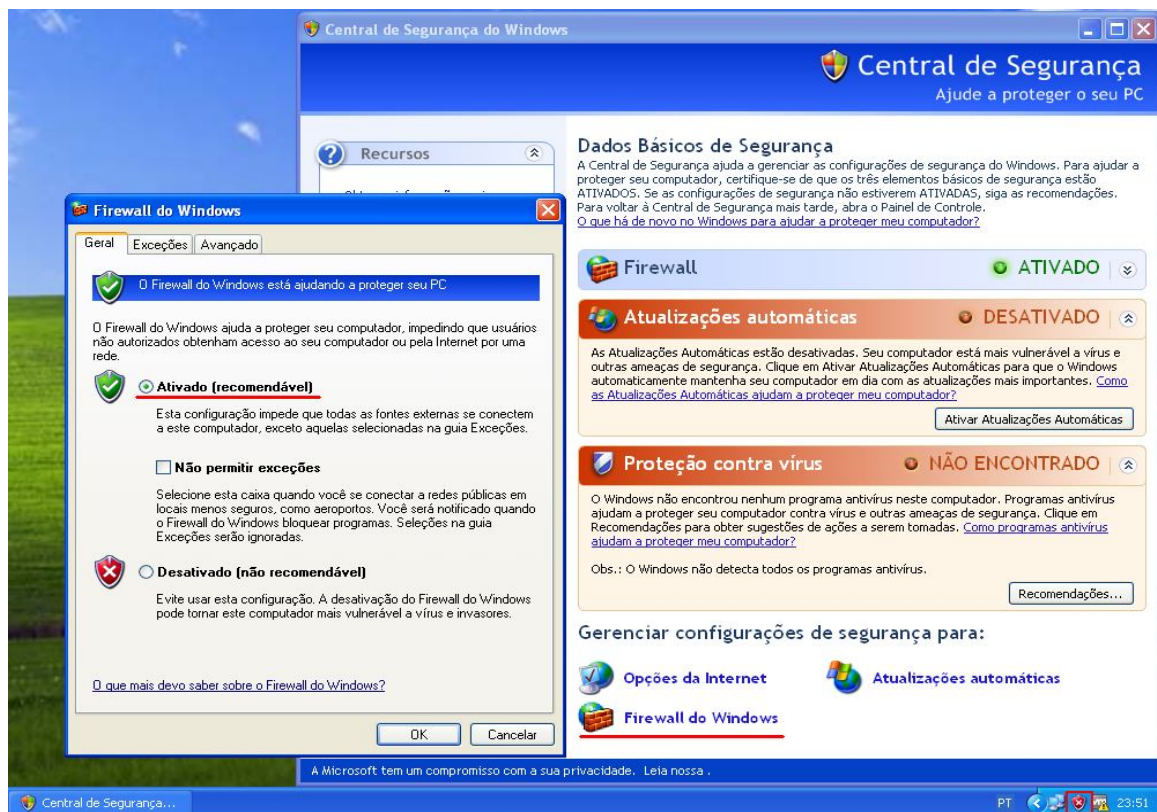
### **`nmap -PU -v 192.168.6.1/24`**

A sondagem UDP normalmente é realizada em uma porta alta que provavelmente estará fechada e deve causar um pacote ICMP de porta inalcançável como resposta. Isso diz ao NMAP que a máquina está ativa e disponível.

Entretanto isso não nos dá nenhuma garantia de que conseguimos detectar todos os hosts da rede com sucesso, pois há firewalls que estão configurados para bloquearem todos ou quase todos os tipos de pacotes de hosts desconhecidos. Desta forma, a máquina alvo pode não responder a nenhum pacote ICMP, impedindo a detecção.

Faça o teste novamente repetindo o comando **`nmap -PU -v 192.168.6.1/24`** mas desta vez ative o Firewall do Windows de uma das máquinas virtuais e veja o resultado.

Você pode ativar o Firewall clicando duplamente no ícone do escudo vermelho próximo ao relógio e em seguida vá até “Gerenciar configurações de segurança para:” e clique em “Firewall do Windows”. Na aba “Geral” selecione a opção “Ativado” e clique em OK.



Você irá notar que desta vez as portas antes detectadas pela varredura não mais foram detectadas na máquina em que o firewall foi ativado.

## Uma técnica falha

Existe ainda uma outra opção para listar hosts na rede com o Nmap que utiliza uma técnica de DNS reverso. Esta técnica promete listar os hosts da rede sem enviar nenhum pacote a eles. O comando para realizar tal teste é:

```
nmap -sL 192.168.6.1/24
```

Entretanto em nenhum dos testes realizados esta técnica funcionou. Mesmo existindo alguns hosts ativos na rede, o Nmap sempre irá reportar que todos os hosts estão inativos. Faça o teste para comprovar.

## Tarefa 2: Varredura de portas

Agora que já conhecemos os nossos possíveis alvos na rede, não precisamos mais disparar varreduras por toda a rede, evitando o desperdício dos nossos recursos e do nosso tempo. Para isso anote os endereços de IPs encontrados em suas varreduras anteriores, pois iremos utilizá-los novamente aqui.

Basicamente as técnicas que utilizamos para o descobrimento de hosts são as mesmas que iremos utilizar para a varredura de portas. Essas técnicas podem ser inclusive combinadas para que tenhamos uma varredura mais completa, como por exemplo, TCP e UDP. Mas simplesmente ativar todas as opções de busca ao mesmo tempo pode fazer com que o tempo de busca se torne muito alto, inviabilizando a nossa varredura.

**Situação 1: Você não tem privilégios de administrador (ou root) mas quer fazer uma varredura nos hosts descobertos a procura de portas.**

Neste caso você não poderá utilizar a opção de TCP SYN do Nmap, pois esta opção utiliza pacotes brutos e para utilizar estes pacotes é necessário ter privilégios de administrador (ou root). Entretanto existe uma opção de varredura de portas no Nmap para este caso e ela se chama “Scan TCP Connect” –sT.

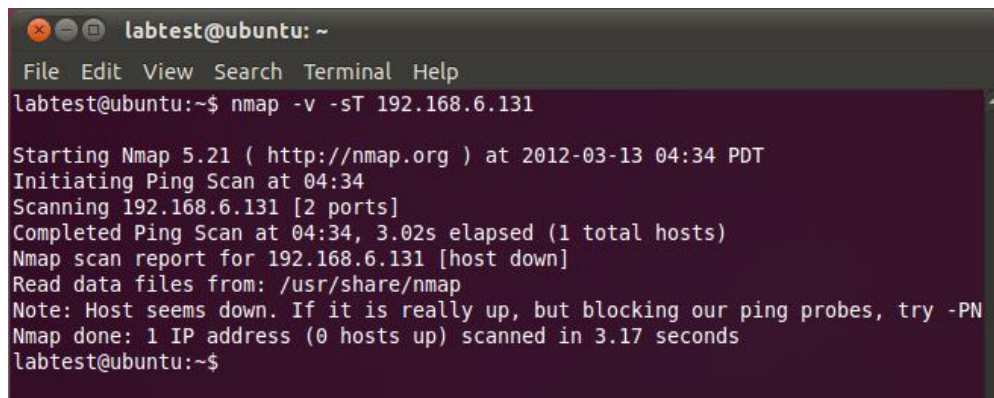
Para realizar este teste utilize de preferência um terminal de uma máquina virtual Ubuntu sem o privilégio de root. Certifique-se antes de que haja na rede pelo menos 1 máquina virtual Windows com o Firewall do Windows ativado. Digite o comando:

**nmap –sT -v 192.168.6.X**

Substitua o X pelo número final de um IP que você encontrou em sua rede durante a descoberta de hosts. Você ainda pode colocar uma faixa de IPs no lugar de apenas

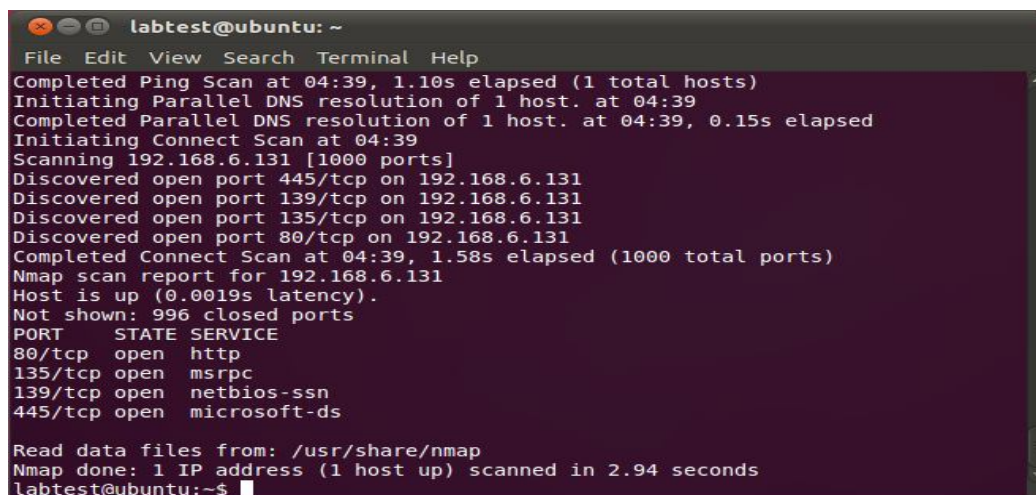
1, caso tenha encontrado vários IPs seqüenciais na rede, ou então pode digitar vários endereços de IPs na mesma pesquisa.

Você irá notar que o Nmap não conseguiu nem notar a presença da máquina alvo na rede, pois o Firewall do Windows está descartando o pedido de conexão enviado pelo Nmap:



```
labtest@ubuntu: ~  
File Edit View Search Terminal Help  
labtest@ubuntu:~$ nmap -v -sT 192.168.6.131  
  
Starting Nmap 5.21 ( http://nmap.org ) at 2012-03-13 04:34 PDT  
Initiating Ping Scan at 04:34  
Scanning 192.168.6.131 [2 ports]  
Completed Ping Scan at 04:34, 3.02s elapsed (1 total hosts)  
Nmap scan report for 192.168.6.131 [host down]  
Read data files from: /usr/share/nmap  
Note: Host seems down. If it is really up, but blocking our ping probes, try -PN  
Nmap done: 1 IP address (0 hosts up) scanned in 3.17 seconds  
labtest@ubuntu:~$
```

Agora vá até a máquina virtual Windows e desative o firewall (como explicado neste roteiro) e repita o teste.



```
labtest@ubuntu: ~  
File Edit View Search Terminal Help  
Completed Ping Scan at 04:39, 1.10s elapsed (1 total hosts)  
Initiating Parallel DNS resolution of 1 host. at 04:39  
Completed Parallel DNS resolution of 1 host. at 04:39, 0.15s elapsed  
Initiating Connect Scan at 04:39  
Scanning 192.168.6.131 [1000 ports]  
Discovered open port 445/tcp on 192.168.6.131  
Discovered open port 139/tcp on 192.168.6.131  
Discovered open port 135/tcp on 192.168.6.131  
Discovered open port 80/tcp on 192.168.6.131  
Completed Connect Scan at 04:39, 1.58s elapsed (1000 total ports)  
Nmap scan report for 192.168.6.131  
Host is up (0.0019s latency).  
Not shown: 996 closed ports  
PORT      STATE SERVICE  
80/tcp    open  http  
135/tcp   open  msrpc  
139/tcp   open  netbios-ssn  
445/tcp   open  microsoft-ds  
  
Read data files from: /usr/share/nmap  
Nmap done: 1 IP address (1 host up) scanned in 2.94 seconds  
labtest@ubuntu:~$
```

Você verá que agora o Nmap consegue detectar que o host está ativo na rede e consegue detectar também as suas portas abertas.

Mas esta técnica só é interessante de ser usada no caso de o usuário não ter privilégios de administrador, pois ela tentará se conectar utilizando o protocolo TCP com cada porta do host alvo e por isso pode ser facilmente detectada, visto que ela irá completar a conexão em caso de encontrar uma porta aberta. Além disso ela pode facilmente falhar em seus resultados caso encontre algum bloqueio pela frente, como um firewall.

**Situação 2: Você tem privilégios de administrador (ou root) e quer fazer uma varredura nos hosts descobertos a procura de portas.**

O fato de termos privilégios de administrador nos permite a utilização de pacotes brutos para a investigação de portas nos hosts alvos. Para isso a opção mais utilizada no Nmap para descoberta de portas é a “Scan TCP SYN” que da mesma forma como vimos na descoberta de hosts, não completa a conexão com as portas alvo e é mais difícil de ser detectada.

Para testá-la digite o comando **sudo nmap -v -sS 192.168.6.X** em um terminal Ubuntu e em seguida digite a senha do usuário (labtest).

Entretanto a presença de firewalls atrapalha os resultados de varreduras realizadas com esta opção. Faça o teste com o firewall do Windows ativado e desativado.

Para maximizar as chances de passar por firewalls, pode-se acrescentar à varredura a opção de “Scan UDP” -sU, mas isso tornará a varredura mais lenta e não garantirá os resultados caso o firewall também bloqueie pacotes UDP.

Assim podemos criar um novo comando que utiliza as duas opções ao mesmo tempo: **sudo nmap -v -sS -sU 192.168.6.X**

Faça o teste e note que agora o Nmap varre a procura de portas utilizando ambos protocolos TCP e UDP.

## **Resumo Roteiro Nmap**

### **Objetivos**

- Descobrir os hosts ativos da rede
- Varrer os hosts descobertos a procura de portas abertas
- Realizar as ações anteriores de diferentes formas em diferentes contextos

### **Pré-Requisitos**

- Leitura da sessão 4.1 Scanner de Rede: NMAP
- Execução simultânea de pelo menos 1 máquina virtual Windows e 1 máquina virtual Ubuntu para a realização dos testes.

### **Tarefa 1: A descoberta de hosts**

Situação 1: Suponha que você conheça o endereço IP do alvo que deseja varrer em sua rede. Este endereço IP é o do gateway da rede, no nosso caso 192.168.6.2.

- Faça uma varredura no alvo específico 192.168.6.2 digitando o endereço no campo “Alvo” da interface Zenmap.

Situação 2: Executando o número máximo de máquinas virtuais na rede que você puder, suponha que você desconheça o número de máquinas existentes na rede e os seus endereços IPs. Suponha também que a rede é de sua propriedade e que não há problema nenhum varrer a rede a procurar de hosts ativos.

- Execute a varredura por Ping: `nmap -sP -v 192.168.6.1/24`

Situação 3: Tome por base a situação 2, mas desta vez, suponha que a rede não é de sua propriedade e que você deve evitar ao máximo levantar suspeitas de que a rede está sendo varrida.

- Execute a varredura utilizando pacotes SYN: `nmap -PS -v 192.168.6.1/24`
- Execute a varredura utilizando pacotes ACK: `nmap -PA -v 192.168.6.1/24`

Situação 4: Suponha que você queira descobrir os hosts da rede e você sabe que podem existir firewalls bloqueando o caminho. Caso os pacotes do Nmap encontrem um firewall, a resposta pode ser comprometida e não condizer com a realidade. Sabendo disso, você precisa maximizar as chances de obter bons resultados.

- Execute a varredura utilizando pacotes UDP: `nmap -PU -v 192.168.6.1/24`
- Ative o firewall do Windows de uma das máquinas virtuais e repita o procedimento anterior.

Outra técnica:

- Execute a varredura utilizando a técnica de DNS reverso: `nmap -sL 192.168.6.1/24`

## **Tarefa 2: Varredura de portas**

Situação 1: Você não tem privilégios de administrador (ou root) mas quer fazer uma varredura nos hosts descobertos a procura de portas.

- Certifique-se de que haja na rede pelo menos 1 máquina virtual Windows com o firewall ativado. Em um terminal de comandos Ubuntu sem privilégios de

root execute: `nmap -sT -v 192.168.6.X` (troque o X pelo número do IP a ser varrido).

- Desative o firewall do Windows da máquina virtual e repita o procedimento anterior.

Situação 2: Você tem privilégios de administrador (ou root) e quer fazer uma varredura nos hosts descobertos a procura de portas.

- Certifique-se de que haja na rede pelo menos 1 máquina virtual Windows com o firewall ativado. Execute o comando em um terminal de comandos do Ubuntu: `sudo nmap -v -sS 192.168.6.X` (troque o X pelo número do IP a ser varrido e utilize a senha labtest).
- Desative o firewall do Windows da máquina virtual e repita o procedimento anterior.
- Faça uma varredura mais completa: `sudo nmap -v -sS -sU 192.168.6.X`

## **6.2 – Scanner de Vulnerabilidades Utilizando o NESSUS**

### **Objetivos**

- Entender e criar políticas de varredura para os nossos testes.
- Descobrir vulnerabilidades de segurança em diferentes hosts da rede interna e externa.
- Analisar o resultado de tais vulnerabilidades e as medidas que podem ser tomadas para mitigá-las ou eliminá-las.

### **Pré-Requisitos**



- Leitura da sessão 4.2 – Scanner de Vulnerabilidades: NESSUS
- Execução simultânea de pelo menos 1 máquina virtual Windows e 1 máquina virtual Ubuntu para a realização dos testes.

Neste roteiro vamos utilizar o Nessus em sua versão 5.0.0 instalado na máquina virtual Ubuntu para procurar por vulnerabilidades em hosts da rede (Windows e Linux) e posteriormente analisar estas vulnerabilidades para saber qual é o grau de risco de cada uma delas e o que podemos fazer para mitigá-las ou até eliminá-las.

O Nessus já encontra-se instalado em nossa máquina virtual Ubuntu que é a máquina que utilizaremos para executar o Nessus e realizar as varreduras por vulnerabilidades. Entretanto, para instalarmos e configurarmos o Nessus em sistemas Ubuntu devemos proceder da seguinte forma:

- Vá até o site <http://www.nessus.org/download/> e clique em “AGREE” para concordar com os termos de prestação de serviço.
- Faça o download da opção Ubuntu 11.10 (32 bits): Nessus-5.0.0-ubuntu1110\_i386.deb (24668 KB).
- Em um terminal vá até o diretório onde está o pacote deb e digite: `sudo dpkg -i Nessus-5.0.0-ubuntu1110_i386..deb` (senha root: “labtest”).

```
labtest@ubuntu:~/Desktop$ sudo dpkg -i Nessus-5.0.0-ubuntu1110_i386.deb
[sudo] password for labtest:
(Reading database ... 147278 files and directories currently installed.)
Preparing to replace nessus 5.0.0 (using Nessus-5.0.0-ubuntu1110_i386.deb) ...
$Shutting down Nessus : .
Unpacking replacement nessus ...
Setting up nessus (5.0.0) ...
nessusd (Nessus) 5.0.0 [build R23018] for Linux
(C) 1998 - 2012 Tenable Network Security, Inc.

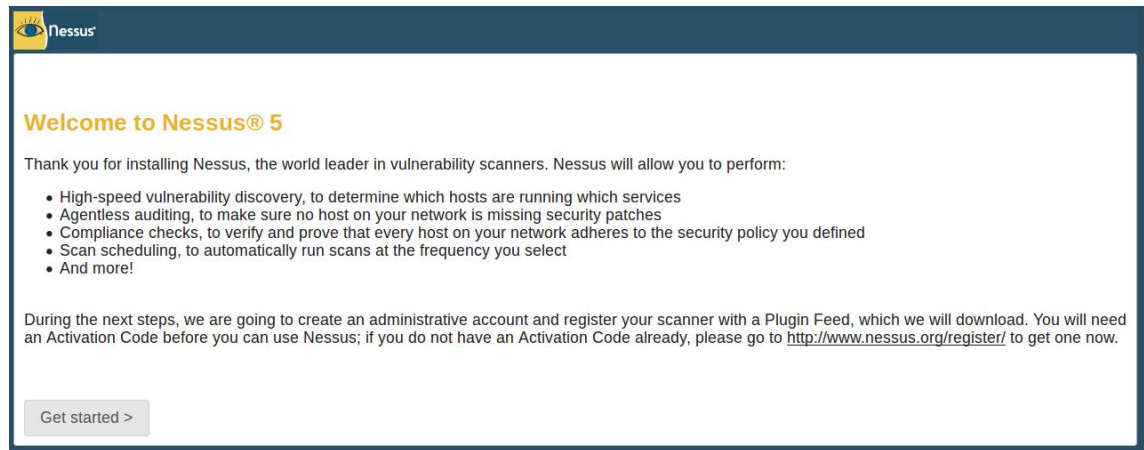
Processing the Nessus plugins...
[#####]

All plugins loaded

- You can start nessusd by typing /etc/init.d/nessusd start
- Then go to https://ubuntu:8834/ to configure your scanner

Processing triggers for ureadahead ...
```

- No terminal inicie o Nessus servidor digitando: `sudo /etc/init.d/nessusd start`
- Em um navegador web digite: <https://ubuntu:8834/> para acessar a interface Web Server do Nessus e em seguida clique em “Get Started”.



- Crie um login e uma senha para a conta de administrador, no nosso caso login: labtest senha: labtest.
- Em seguida será necessário digitar o código de ativação. Para isso visite <http://www.nessus.org/register> e selecione a opção “Home Feed” a qual é gratuita e para uso doméstico.



- Será necessário concordar com os termos e em seguida clicar em “AGREE”.
- Preencha seu nome e e-mail para receber o código de ativação.

- Insira o código de ativação e aguarde até que o Nessus faça o download dos plugins mais atuais.
- Entre com o login e senha do administrador para criar um outro usuário.
- Clique em Users e em seguida em Add.
- Escolha um nome de usuário e uma senha e em seguida clique em Submit (certifique-se de não ter marcado a caixa de seleção “Administrator”).
- Pronto! Agora basta utilizar a interface Web Server do Nessus com o novo usuário criado para realizar as varreduras por vulnerabilidades.

Se você ainda não iniciou o Nessus servidor, inicialize-o em um terminal de comandos do Ubuntu através do comando: **sudo /etc/init.d/nessusd start** em seguida acesse a interface Web Server do Nessus através do navegador web em:

<https://ubuntu:8834/>

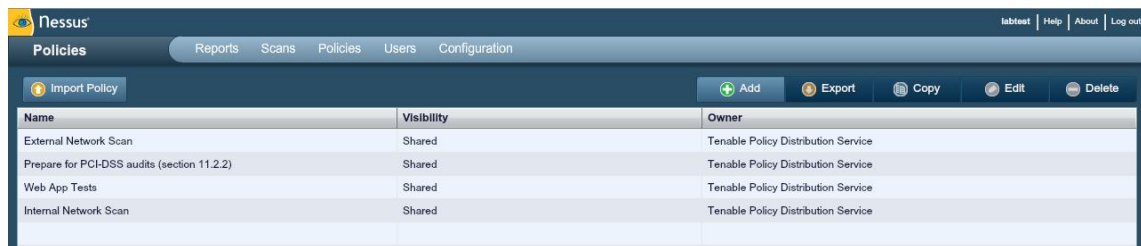
Para testar o Nessus, iremos realizar aqui 3 tarefas sequenciais que são: Definir as políticas que o Nessus irá utilizar para fazer a varredura, realizar a varredura propriamente dita de acordo com as políticas definidas e por fim analisar os resultados obtidos.

### **Tarefa 1: Definir as políticas**

Nesta tarefa vamos analisar as políticas existentes do Nessus e também criar nossas políticas para executar as nossas varreduras. Entretanto a criação de políticas no Nessus não é uma tarefa por definição estática, pois a constante atualização dos plugins utilizados pelo Nessus pode nos oferecer diferentes opções de configurações de acordo com novos plugins adicionados. O fato de as varreduras do Nessus darem cobertura a muitas vulnerabilidades faz com que a lista de opções

de configurações seja um pouco extensa, de modo que vamos desconsiderar muitas delas e deixar a configuração padrão, como indica o próprio manual da ferramenta.

Antes de criarmos novas políticas note que o Nessus já traz algumas políticas definidas. Confira clicando na aba “Policies” que aparece na parte superior da interface:



Você pode analisar detalhadamente as configurações de cada política definida pelo Nessus se fizer login como administrador e clicar em “Edit” após selecionar uma das políticas.

Agora vamos criar nossas próprias políticas para varrer as redes.

Para as configurações que não forem citadas, deixe-as como estão por padrão.

### Política 1: Rede virtual

- O primeiro passo é clicar em “Add” na aba “Policies”.
- Em seguida no submenu “General” vamos preencher na área “Basic” o nome da nossa política no campo “Name” com “Rede virtual”.
- Na área “Scan” vamos desmarcar a opção “Safe Checks”, pois esta opção fará com que o Nessus não utilize plugins que possam causar instabilidade no alvo remoto. Como nós vamos varrer a nossa própria rede virtual e esta é uma rede criada para testes, não há problemas caso isso aconteça.

- Na área “Network Congestion” iremos deixar ambas opções desmarcadas como estão, pois não é preciso nos preocuparmos com o congestionamento em nossa rede virtual.
- Na área “Port Scanners” também não iremos mexer. Aqui o Nessus irá utilizar o Nmap instalado em nossa máquina virtual para efetuar a varredura a procura de portas.
- Na área “Port Scan Options” iremos alterar o campo “Port Scan Range” para “all”. Esta política terá o objetivo de varrer a nossa rede virtual e como sabemos que a nossa rede não é muito grande devido às nossas limitações de executar muitas máquinas virtuais ao mesmo tempo, podemos fazer a varredura em todas as portas dos hosts.
- Na área “Performance” deixaremos as configurações padrões.
- Não iremos utilizar credenciais, portanto passemos para o submenu “Plugins” para escolhermos quais plugins o Nessus irá utilizar para detectar possíveis vulnerabilidades.
- Note que por padrão todos os plugins estão ativados. Como queremos fazer uma varredura completa iremos deixar a maior parte deles ativados, desativando apenas alguns que não são interessantes para a nossa rede virtual.
- Clique na bolinha verde das seguintes famílias de plugins para desativá-las por completo (ela ficará cinza), pois não serão úteis em nossa política personalizada para varrer a rede interna a qual conhecemos:
  - AIX Local Security Checks: Verificações para sistemas IBM AIX.
  - CentOS Local Security Checks: Verificações para sistemas CentOS Linux.

- CGI abuses: Verificações para aplicações web, incluindo testes de injeção SQL, inclusão de arquivo local (LFI), inclusão de arquivo remoto (RFI), entre outros.
  - CGI abuses : XSS: Verificações para aplicações web em cross-site scripting (XSS).
  - FreeBSD Local Security Checks: Verificações para sistemas FreeBSD.
  - HP-UX Local Security Checks: Verificações para sistemas HP-UX.
  - Junos Local Security Checks: Verificações para sistemas Juniper Junos.
  - MacOS X Local Security Checks: Verificações para sistemas Apple Mac OS X.
  - Solaris Local Security Checks: Verificações para sistemas Oracle Solaris.
- Dê uma olhada nas preferências do submenu “Preferences”. A lista de preferências muda de acordo com os plugins instalados no Nessus. Não será preciso alterar nada em “Preferences” pois os valores preenchidos por padrão pelo Nessus já está adequado para nossa varredura.
  - Por fim, clique em “Submit” para salvar a nova política.

## **Política 2: Rede externa**

- Siga os mesmos passos da Política 1, apenas mudando o preenchimento de alguns campos.
- No submenu “General” preencha na área “Basic” no campo “Name” com o nome “Rede externa”.
- Na área “Scan” iremos deixar as seguintes opções marcadas:

- Allow Post-Scan Report Editing: Nos permite a edição do relatório após a varredura.
  - “Safe Checks” automaticamente faz com que o Nessus desmarque plugins que podem causar algum tipo de dano à rede. Como iremos varrer uma rede externa a qual desconhecemos é importante marcar esta opção.
  - Silent Dependencies: Não inclui a lista de dependências entre plugins no relatório da varredura.
  - Avoid Sequential Scans: Faz com que a varredura não seja seqüencial caso haja vários IPs na rede, de modo a tentar evitar a detecção da nossa varredura nas redes externas.
- Na área “Network Congestion” iremos deixar ambas opções marcadas, para que o Nessus administre o congestionamento de tráfego e evite a sobrecarga na rede externa.
- Na área “Port Scanners” iremos deixar apenas a opção “SYN SCAN” marcada, de modo que o Nessus irá utilizar o Nmap instalado em nossa máquina virtual para efetuar uma varredura silenciosa a procura de portas.
- Na área “Port Scan Options” iremos deixar o campo “Port Scan Range” em “default” pois se a rede externa for muito grande, a varredura poderá levar muito tempo para terminar e gastará muito tempo a procura de portas pouco prováveis de serem utilizadas (o Nessus irá varrer aproximadamente 4.790 portas comuns).
- Na área “Performance” deixaremos as configurações padrões.

- Não iremos utilizar credenciais, portanto passemos para o submenu “Plugins” para escolhermos quais plugins o Nessus irá utilizar para detectar possíveis vulnerabilidades.
- Como vamos varrer diferentes redes externas, não sabemos exatamente que tipos de sistemas operacionais os hosts da rede poderão estar executando e portanto deixaremos habilitados todos os plugins. Como marcamos a opção “Safe Checks” o Nessus irá automaticamente desabilitar plugins que podem ser danosos para a rede, de modo que não precisaremos desmarcá-los aqui.
- Por fim, clique em “Next” e depois “Submit” para salvar a nova política.

## **Tarefa 2: Realizar a varredura**

Para acessar a tela de varreduras clique no menu superior “Scans”.

- Clique em “Add” e preencha os campos com os seguintes valores:
  - Name: Varredura de Rede Virtual 1
  - Type: Run Now
  - Policy: Rede Virtual
  - Scan Targets: 192.168.6.1/24
- Em seguida clique em “Launch Scan” para dar início a varredura na rede virtual interna.
- Realize o mesmo procedimento para realizar uma varredura em uma rede externa, preenchendo os campos da seguinte forma:
  - Name: Varredura de Rede Externa 1
  - Type: Run Now
  - Policy: Rede Externa
  - Scan Targets: [www.inf.ufsc.br](http://www.inf.ufsc.br)



### Tarefa 3: Analisar os relatórios

Agora que já realizamos as varreduras, vamos analisar os resultados.

Para acessar a tela de relatórios clique no menu superior “Reports”.

De um duplo clique no nome do relatório para analisá-lo.

Vamos começar analisando o relatório “Varredura de Rede Virtual 1”:

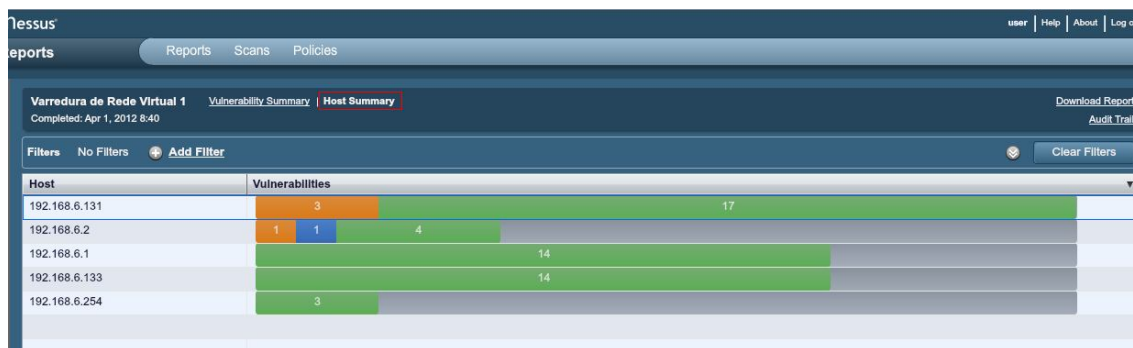


The screenshot shows the Nessus Reports interface. At the top, there's a navigation bar with 'Reports', 'Scans', and 'Policies'. Below it, the report title 'Varredura de Rede Virtual 1' is displayed, along with 'Vulnerability Summary' and 'Host Summary' tabs. The completion date is 'Apr 1, 2012 8:40'. On the right, there are links for 'Download Report', 'Remove Vulnerability', and 'Audit Trail'. Below the header, there's a filter section with 'Filters', 'No Filters', and an 'Add Filter' button. The main content is a table with columns: Plugin ID, Count, Severity, Name, and Family. The table lists various vulnerabilities and their counts, such as 'HTTP TRACE / TRACK Methods Allowed' (1, Medium) and 'Nessus SYN scanner' (5, Info).

Plugin ID	Count	Severity	Name	Family
11213	1	Medium	HTTP TRACE / TRACK Methods Allowed	Web Servers
12217	1	Medium	DNS Server Cache Snooping Remote Information Disclosure	DNS
57791	1	Medium	Apache 2.2 < 2.2.22 Multiple Vulnerabilities	Web Servers
57792	1	Medium	Apache HTTP Server httpOnly Cookie Information Disclosure	Web Servers
50686	1	Low	IP Forwarding Enabled	Firewalls
11219	5	Info	Nessus SYN scanner	Port scanners
19506	5	Info	Nessus Scan Information	Settings
20094	4	Info	VMware Virtual Machine Detection	General
35716	4	Info	Ethernet Card Manufacturer Detection	Misc.
11936	3	Info	OS Identification	General
22964	3	Info	Service Detection	Service detection
45590	3	Info	Common Platform Enumeration (CPE)	General
54615	3	Info	Device Type	General
10107	2	Info	HTTP Server Type and Version	Web Servers
10287	2	Info	Traceroute Information	General
20301	2	Info	VMware ESX/GSX Server detection	Service detection
24260	2	Info	HyperText Transfer Protocol (HTTP) Information	Web Servers
25220	2	Info	TCP/IP Timestamps Supported	General
43111	2	Info	HTTP Methods Allowed (per directory)	Web Servers
10114	1	Info	ICMP Timestamp Request Remote Date Disclosure	General
10884	1	Info	Network Time Protocol (NTP) Server Detection	Service detection
11002	1	Info	DNS Server Detection	DNS
12053	1	Info	Host Fully Qualified Domain Name (FQDN) Resolution	General
12634	1	Info	Authenticated Check: OS Name and Installed Package Enumeration	Settings
18261	1	Info	Apache Banner Linux Distribution Disclosure	Web Servers
22869	1	Info	Software Enumeration (SSH)	General
39521	1	Info	Backported Security Patch Detection (WWW)	General

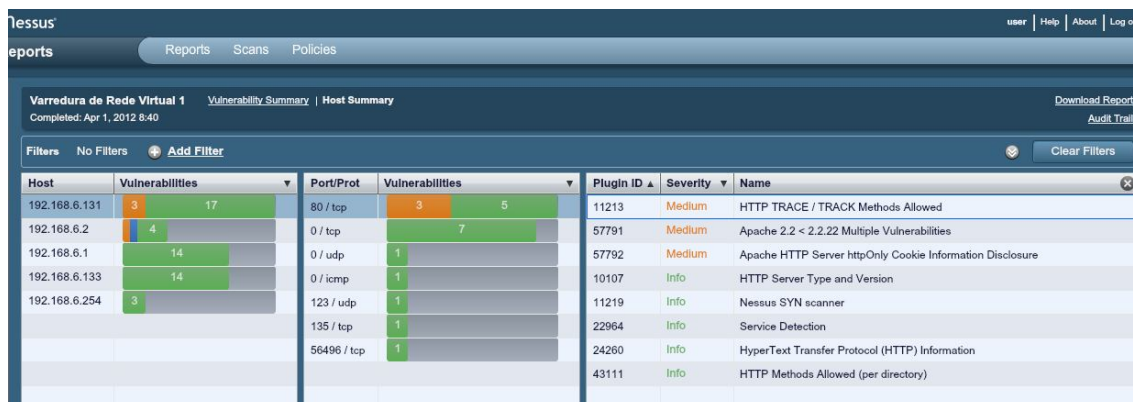
Nesta tela, temos a lista de vulnerabilidades encontradas em todos os hosts da rede alvo, classificados de acordo com a gravidade do risco.

Para visualizar a lista de vulnerabilidades encontradas em um host específico da rede, basta clicar no topo do relatório em “Host Summary” como mostra a figura abaixo:



Desta forma é possível visualizar em cada linha da tabela um host diferente da rede alvo e um gráfico colorido indicando o número de vulnerabilidades encontradas de acordo com a gravidade do risco.

Para se obter mais informações sobre estas vulnerabilidades, basta dar um duplo clique em cima do host desejado. Com duplos cliques em cima das novas informações mostradas é possível se obter mais informações:



Na imagem seguinte é mostrado o relatório da varredura “Varredura de Rede Externa 1”:

Nessus

Reports Scans Policies

Varredura de Rede Externa 1 Vulnerability Summary | Host Summary

Completed: Apr 1, 2012 8:58 (1 Error)

Download Report Remove Vulnerability Audit Trail

Filters No Filters Add Filter Clear Filters

Plugin ID	Count	Host	Port
17710	1	www.inf.ufsc.br	80 / tcp
22268	1		
33849	1		
35067	1		
41014	1		
50069	1		
57537	1		
11213	1		
17696	1		
31407	1		
35750	1		
39480	1		
43351	1		
44921	1		
57792	1		
11219	3		
10028	1		
10107	1		
11002	1		
11424	1		
11936	1		
12053	1		
19506	1		
22964	1		
24260	1		
31658	1		
35371	1		

Plugin ID: 17710 Port / Service: www (80/tcp) Severity: High

Plugin Name: PHP < 4.4.4 Multiple Vulnerabilities

**Synopsis:** The remote web server uses a version of PHP that is affected by multiple vulnerabilities.

**Description:** According to its banner, the version of PHP installed on the remote host is older than 4.4.4. As such, it is potentially affected by the following vulnerabilities:

- The c-client library 2000, 2001, or 2004 for PHP does not check the safe\_mode or open\_basedir functions. (CVE-2006-1017)
- A buffer overflow exists in the scanf function. (CVE-2006-4020)

**Solution:** Upgrade to PHP version 4.4.4 or later.

**See Also:** <https://bugs.php.net/bug.php?id=38322>  
[http://www.php.net/releases/4\\_4\\_4.php](http://www.php.net/releases/4_4_4.php)

**Risk Factor:** High

**CVSS Base Score:** 9.3 (CVSS2#AV:N/AC:M/Au:N/C:C/I:C/A:C)

**CVSS Temporal Score:** 7.3 (CVSS2#E:POC/RL:OF/RC:ND)

**Plugin Output:** Version source : Server: Apache/2.0.59 (FreeBSD) DAV/2 PHP/4.4.2 mod\_ssl/2.0.59 OpenSSL/0.9.8b  
Installed version : 4.4.2  
Fixed version : 4.4.4

**CPE:** cpe:/a:php:php

**CVE:** [CVE-2006-1017](#)  
[CVE-2006-4020](#)

**BID:** [16878](#)

Nesta imagem podemos ver as informações detalhadas de uma das vulnerabilidades encontradas no alvo “www.inf.ufsc.br”. Dentre estas informações, podemos ver a sinopse e a descrição da vulnerabilidade encontrada e em seguida a solução apresentada pelo Nessus para resolver o problema. Há ainda referências de links externos relacionados com a vulnerabilidade encontrada.

Se você preferir, é possível ainda gerar um documento HTML contendo o relatório, clicando no canto superior direito do relatório no Nessus em “Download Report”.

Agora que você já sabe analisar os resultados obtidos, analise cautelosamente cada vulnerabilidade encontrada nos relatórios de ambas as varreduras realizadas para aprender um pouco mais sobre elas.

### 6.3 – SQL Injection Utilizando o MANTRA

### 6.4 – Cross Site Scripting (XSS) Utilizando o MANTRA

#### Bibliografia

GREGG, Michael. **Build Your Own Security Lab: A Field Guide for Network Testing**. Publicado por Wiley Publishing, Inc., Indianapolis, Indiana, 2008. ISBN: 978-0-470-17986-4.

NANCE, Kara; HAY, Brian; DODGE, Ronald; SEAZZU, Alex; BURD, Steve. **Virtual Laboratory Environments: Methodologies for Educating Cybersecurity Researchers**. Department of Computer Science, University of Alaska Fairbanks; Department of Electrical Engineering and Computer Science, U.S. Military Academy; Anderson School of Management, University of New Mexico. Methodological Innovations Online, 2009.

VMWare. Disponível em: <<http://www.vmware.com/br/virtualization/virtualization-basics/what-is-virtualization.html>>. Acessado em 23 de Janeiro de 2012.

LAUREANO, Marcos Aurélio Pchek; MAZIERO, Carlos Alberto. **Virtualização: Conceitos e Aplicações em Segurança**. Pontifícia Universidade Católica do Paraná, Centro Universitário Franciscano. 2008.

DASWANI, Neil; KERN, Christoph; KESAVAN, Anita. **Foundations of Security – What every Programmer Needs to Know**. 2007. ISBN-13 (pbk): 978-1-59059-784-2. ISBN-10 (pbk): 1-59059-784-2.

CLARKE, Justin. **SQL Injection Attacks and Defense**. Publicado por Syngress Publishing, Inc. Elsevier, Inc. 30 Corporate Drive Burlington, MA 01803, 2009. ISBN 13: 978-1-59749-424-3.

HARRIS, Shon; HARPER, Allen; EAGLE, Chris; NESS, Jonathan; LESTER Michael. **Gray Hat Hacking: The Ethical Hacker's Handbook**. Publicado por McGraw-Hill/Osborne. 2100 Powell Street, 10<sup>th</sup> Floor Emeryville, California 94608, U.S.A. 2005. ISBN 0-07-225709-1.

OWASP. **OWASP Top 10 – 2007 The Ten Most Critical Web Application Security Risks**. Disponível em: < [http://www.owasp.org/index.php/Top\\_10](http://www.owasp.org/index.php/Top_10)>. Acessado em 30 de Junho de 2011.

CGI Security. **The Cross-Site Scripting (XSS) FAQ**. Disponível em: <<http://www.cgisecurity.com/xss-faq.html>>. Acessado em 16 de Janeiro de 2012.

OWASP. **OWASP Computer Viruses**. Disponível em: < [https://www.owasp.org/index.php/Computer\\_Viruses](https://www.owasp.org/index.php/Computer_Viruses)>. Acessado em 30 de Junho de 2011.

NMAP. **Guia de Referência do Nmap**. Disponível em: <[http://nmap.org/man/pt\\_BR/](http://nmap.org/man/pt_BR/)>. Acessado em 11 de Janeiro de 2012.