

Nicholas Ferreira

1ª edição

# O Guia do Hacker



O Guia para iniciantes no mundo hacker

# Prefácio

Depois de muitas perguntas que não foram respondidas, muitas dúvidas que não foram sanadas, muitos tópicos abandonados, eu decidi criar este livro, onde vou explicar o funcionamento do mundo hacker da forma mais detalhada, simples e descontraída possível, usando uma linguagem que todos possam entender claramente.

Se você veio aqui em busca de tutoriais *'explicativos'* que dão tudo na sua mão, aqui não é o seu lugar.

No futuro, escreverei outro livro, com os mesmos tópicos que esse, porém, com aplicações na prática. Publicarei este livro, se vocês gostarem e quiserem, eu escrevo o próximo, com os ensinamentos na prática.

A ideia da criação deste guia foi do membro **H.Cr7pt3r** do Fórum Guia do Hacker, então, créditos à ele pelo incentivo.

Parte do conteúdo deste livro foi espelhada no tópico FAQ, criado pelo membro **Void\_Witch**, em 05/11/2013. Outros tópicos e matérias também foram usados como consulta. Todas as referências bibliográficas estão anexadas ao final do livro.

Eu, **Nicholas Ferreira** (*Nickguitar.dll*), publicarei este guia oficialmente no Fórum Guia do Hacker, sendo qualquer outra fonte fruto de cópia. Dedico totalmente esse guia para os usuários do Fórum, especialmente para os membros iniciantes.

**Escrito de 24 de novembro de 2014 a 4 de dezembro**, Publicado em 5 de dezembro de 2014. Qualquer conteúdo retirado daqui e publicado depois desta data é fruto cópia!

# Sumário

1	Introdução
1.1	Hacker
1.2	Cracker
1.3	Gray Hat, o meio termo
2	Ética, a filosofia hacker
3	Engenharia social, um pouco de psicologia
3.1	As fraudes na WEB
4	Por onde começar, o que aprender?
5	Programação
5.1	Banco de dados
6	Pentest
6.1	Exploiting
6.2	Deface
6.3	Google Hacking
7	Redes: Uma visão geral
7.1	Negação de serviço (DoS)
8	Malwares: Tipos e características
9	Técnicas para esconder malwares
9.1	Crypters
9.2	Binders
9.3	Packers
10	Análise de malware: Uma visão geral
10.1	Análise estática
10.2	Análise dinâmica
11	Engenharia Reversa
11.1	Software Cracking
12	Criptografia
13	Anonimidade e segurança na web
14	Deep Web
15	FAQ
16	Referências bibliográficas

# 1. Introdução



Bom, imagino que você que baixou esse livro está iniciando agora no mundo hacker, ou então já iniciou há um tempo mas não tem conhecimento de outras áreas desse ramo. Então, os ensinamentos passados aqui serão feitos com uma didática e linguagem simples, para o fácil entendimento de todos.

Pretendo explicar com o máximo de detalhes possíveis, e de uma forma descomplicada, o funcionamento do mundo hacker, bem como as ferramentas e técnicas usadas por eles. Mas não confunda hacker com cracker, existe um abismo de diferença entre esses dois caras.

## 1.1 Hacker

Hacker, ou white hat (chapéu branco), de acordo com Eric Raymond, criador do símbolo hacker, o 'hacker' é o *indivíduo que tem um conhecimento extraordinário em informática, e usa esse conhecimento para elaborar e modificar softwares e hardwares de computadores, seja desenvolvendo funcionalidades novas ou adaptando as antigas, mas sempre usando o conhecimento de forma legal (legal no sentido de legalidade, dentro da lei)*. O hacker pode então desenvolver softwares antimalwares, para combater com as pragas virtuais, criadas pelos crackers.

## 1.2. Cracker

Não muito conhecido pelas pessoas, o cracker, ou black hat (chapéu preto), é praticamente o hacker do mal. Os crackers são indivíduos também com amplo conhecimento em informática, mas que usam esse conhecimento para quebrar sistemas de segurança a fim de obter vantagens ilícitas. Os crackers muitas vezes são generalizados e conhecidos apenas como os caras que crackeiam programas e jogos pagos, por exemplo photoshop, Battlefield 3, ou o próprio Windows (ou vai me dizer que você pagou por esse Windows que está usando agora?! Rs), mas na verdade os crackers fazem muito mais do que isso. Eles podem por exemplo criar softwares maliciosos (malwares) que roubam as suas senhas quando você digita.

## 1.3 Gray Hats

Os gray hats (chapéu cinza) são o meio-termo entre hacker e cracker. “Como assim meio-termo, Nick?”. Bom, nem todos são bonzinhos o tempo todo, né? Nem malvados toda hora. Gray hat é aquele cara que também tem bastante conhecimento em informática e hacking, sabe os limites, mas de vez em quando, ultrapassa esses limites, seja por diversão, ou qualquer outro motivo, mas sem causar danos significativos e sem roubar informações do sistema invadido. O gray hat pode invadir o PC de um colega, trocar o plano de fundo dele para um pênis e zuar com a cara dele pro resto da vida, mas sem causar qualquer dano significativo na máquina da vítima.

**Até o próximo capítulo, onde falaremos sobre ética!**

## 2. Ética Hacker



Antes de tudo, o que é ética? Ética, de acordo com o dicionário, é o ramo da filosofia que estuda o comportamento moral de um indivíduo em uma sociedade. Traduzindo do grego para português, ética significa bons costumes, caráter...

“Beleza Nick, mas o que isso tem a ver com hacker?”

Aí que tá... Como eu já expliquei acima, hacker é diferente de cracker, que é diferente do gray hat, já que o hacker tem noção do que faz, tem noção do que é certo e errado, e faz o correto.

Para conseguir bons contatos e fazer boas amizades no mundo hacker, é necessário agir com ética, ter caráter. Coisas como invadir um site puramente por diversão e apagar todo o conteúdo que lá tem é coisa de cracker, e totalmente antiético e imoral.

Um exemplo de atitude de um hacker: O hacker encontra uma falha de segurança no site da polícia federal, e com a exploração dessa falha ele teria acesso ao sistema de busca de indivíduos por CPF, uma espécie de consulta. Então ele entra em contato com o webmaster do site avisando da falha, sem causar nenhum dano e sem divulgar nada publicamente sobre a falha.

Um exemplo de atitude de um cracker: O cracker encontra uma falha em um dos servidores do Outlook, que permite que ele envie email para qualquer pessoa se passando por outra. Ao invés de alertar à Microsoft sobre a falha, ele a explora e consegue usá-la a seu favor, para enviar spam para várias pessoas.

Não quero que você pense que eu estou aqui para ditar regras, de forma alguma, estou apenas comentando sobre a ética que um hacker deve ter. Cabe a você, depois de ler o livro todo, a escolher que caminho quer seguir. Eu particularmente me considero um gray hat. Já ajudei muita gente que precisava, mas também já fiz algumas brincadeiras com certas pessoas, que estão marcadas até hoje, haha.

Até mais, e bons estudos!



### 3. Engenharia Social



O que é engenharia social? Não sei se já ouviu falar nisso, mas a engenharia social, também conhecida como no-tech hacking, é uma arma poderosa nas mãos de quem souber usá-la, não só no hacking, mas para outras coisas no geral também.

Engenharia social é a prática de conseguir informações sigilosas com outras pessoas, ou fazer com que elas façam o que você quer, sem perceber, usando a argumentação e persuasão a seu favor. Não entendeu nada disso? Vou dar um exemplo que aconteceu comigo, veja:

Eu jogava um jogo pirata online, e um cara tinha hackeado o site do jogo. Como já era de se esperar, fiquei com muita raiva e adicionei o cara no MSN. Perguntei por que ele tinha feito aquilo, e ele respondeu coisas que não tinha entendido... Comecei a xingá-lo, e ele não disse nada. Depois de um tempo xingando ele, ele começou a puxar assunto sobre super heróis comigo... Perguntou quem venceria em uma batalha entre Super-Homem e Homem Aranha, e eu fui conversando...

Não achei nada estranho o papo, e fui conversando e falando sobre o que eu gostava. Até que ele me disse “aguarde um momento, vou hackear um cara”. Daí eu pensei comigo mesmo, “Como assim?! Parece que é só chegar e pronto, hackeado...”. Alguns minutos depois eu fui desconectado do MSN, e quando loguei, a senha tava incorreta... Então eu percebi que o cara que ele disse que ia hackear era eu. Meu coração começou dar tiros, fiquei tenso, pensei em ligar pra polícia, muita coisa passou pela minha cabeça naquele momento...

Afinal, como ele fez isso? Naquela época, existia uma ferramenta de recuperação de senha, em que você selecionava uma pergunta que era disponibilizada pelo sistema, e escrevia a resposta para a pergunta, sendo essa resposta pessoal. Esse sistema é usado até hoje em vários sites.

Lembra-se disso? Pois é, foi por aí que ele me hackeou... A minha pergunta de segurança era o meu super herói de infância favorito, e a resposta era homem aranha. Se ele chegasse e perguntasse direto, qual o meu super herói favorito, eu iria desconfiar na hora e não iria responder... Mas ele foi esperto, e usou engenharia social para que eu falasse do que eu gostava, e então ele conseguiu a resposta, alterou minha senha e deixou uma mensagem no status dizendo para eu tomar cuidado.

Ele é um exemplo de gray hat (nesse momento), já que ele viu que eu estava vulnerável psicologicamente, conseguiu acesso à uma informação que poderia ser fatal nas mãos erradas e usou para invadir minha conta e deixar uma mensagem dizendo para eu tomar cuidado.

Esse foi o primeiro contato com hacking que eu tive, achei incrível, queria fazer com todo mundo, e foi por aí que tudo começou... O nick do cara que me hackeou? Hifterbuk. Gostaria de agradecer à você, Hifter, por ter me hackeado. Ahauhauhahau. Talvez, se não fosse por isso, você não estaria lendo isso agora.

## 3.1 Fraudes bancárias

### O golpe do telefonema

Um desconhecido liga para sua casa e diz ter sequestrado seu filho, e que quer R\$20.000,00 na conta dele hoje, se não vai matá-lo.

Você, no desespero, fala o nome do seu filho, perguntando se está tudo bem com ele. Sabendo disso, o golpista pode fazer ameaças usando o nome de seu filho, o que é bem mais aterrorizante...

Outros golpistas que utilizam mais a internet podem até selecionar suas vítimas. As redes sociais, principalmente o Facebook, são uma ótima forma para saber o que as pessoas estão fazendo. Estamos no tempo em que as pessoas publicam tudo que estão fazendo. *#PartiuAcademia*, *#PartiuEscola*, *#PartiuViajarPraDisney*, além de publicar fotos com a identificação de localização ativada, mostrando em que local a foto foi tirada. Isso se não for daquelas pessoas que publicam o próprio número do telefone pedindo para ser chamada no whatsapp para conversar.

Todas essas informações são valiosas na mão de criminosos. Eles podem descobrir uma certa rotina na sua vida, e planejar golpes seguindo-a. Por exemplo, eles sabem que você trabalha todos os dias das 9h até 18h, então sabem que nesse período, você não tem contato com sua família. Eles podem então ligar para seu(ua) filho(a), deixando o celular dele ocupado para novas ligações, e assim aplicar o golpe do telefonema em você, porque você não vai conseguir ligar para seu filho.

Uma dica que dou para você, é dizer o nome errado do seu filho quando atender esse tipo de chamada.

Ex:

- *Golpista: "Alô? Estou com seu filho aqui."*

- *Você: "Como assim?! Meu filho???"*

- *Golpista: "Sim, estou com ele e você vai ter que pagar pelo resgate, se não eu o mato."*

- *Você: "Está com meu filho (invente um nome qualquer)?"*

- *Golpista: "(finge perguntar o nome para 'seu' filho)... Sim, ele mesmo"*

- *Você: "Ah é? Mas não tenho filho com esse nome. Boa tarde e tchau."*

### O golpe da página fake

Outro golpe que está presente todos os dias é o da página fake. Bem provável que você já saiba o que é, ou pelo menos tenha ouvido falar, mas você sabe como ela funciona? Vou explicar.

A página fake, como o próprio nome diz, é uma página falsa de um determinado site. Exemplo: Banco do Brasil, Facebook, Gmail. Qualquer site com formulário de login pode ser alvo de página fake. O objetivo dela é pegar os dados de quem tentar logar nela. Como isso é feito? Simples. Quando você coloca seu login e senha, e clica no botão Logar, você envia uma requisição à uma página (*geralmente PHP ou ASP*), essa página verifica se seus dados estão certos e faz o login se estiver.

Nós podemos baixar a página, criando uma cópia idêntica, e podemos também alterar a página para qual o formulário fará a requisição quando alguém logar.

Nessa nova página, ao invés de verificar os dados inseridos, ela pega-os e salva em outra página, ou manda para o email do **cracker**. Veja abaixo um exemplo de spam contendo uma página fake em meu email:

RES: URGENTE - PROTOCOLO 03644549 26/11/2014 08:32:07



Nickguitar .dll 26/11/2014 ▶  
Para: [Redacted]

**Segurança**

Confira as **Facilidades** do **Bradesco Internet Banking**

**Prezado Cliente,**

Por motivos de segurança comunicamos a todos os clientes que, visando barrar o constante aumento de fraudes será obrigatorio realizar a **ATUALIZAÇÃO** do seu **CARTÃO CHAVES DE SEGURANÇA**.

Caso não efetue a **ATUALIZAÇÃO OBRIGATORIA** com urgencia, o acesso via **CAIXA-ELETRONICO E INTERNET- BANKING** será **SUSPENSO**.

Utilize o botão **SEGURANÇA** para efetuar à atualização.

**Atenção:**

A **ATUALIZAÇÃO OBRIGATORIA** é de responsabilidade do cliente. O Banco Bradesco S/A não se responsabilizará por danos sofrido caso às **CHAVES** não sejam atualizadas.

Bradesco

Imagem: Acervo Pessoal

Nessa imagem que recebi via email, havia um link para uma página fake, que capturava a agência e senha Bradesco da vítima. Uma pessoa desavisada poderia pensar que a mensagem é real, e fazer a suposta atualização, dando a agência e conta na mão de bandidos.

#### Dicas que eu posso te dar para não cair nesse golpe:

**1º** - Sempre verifique a URL do site antes de logar. A maioria das URLs de páginas fake tem o nome suspeito, diferente do original, ou com algum erro de digitação. Se for diferente do original, não coloque seus dados. Ex:

*Original: [bancodobrasil.com.br](http://bancodobrasil.com.br)*

*Falso: [bancodobrasi.com.br](http://bancodobrasi.com.br)*

**2º** - Verifique se o site tem <https://> no início. A maioria dos sites grandes usam esse serviço de segurança. Se o site que você for logar não tiver com esse serviço ativado, ou tiver com um cadeado quebrado no lugar, não logue, de forma alguma. Isso acontece porque o **crackertentou** forçar o [https](https://) na URL, sem esse serviço estar rodando, por isso é exibido um cadeado quebrado, alertando que o site pode ser falso. O [HTTPS](https://) é a junção do protocolo



HTTP + SSL (HTTP será visto no capítulo de redes, e SSL será visto no capítulo de criptografia).

**3º** - Utilize um filtro anti-phishing. Geralmente eles já vêm integrados à maioria dos navegadores e serve para alertar os usuários quando uma página suspeita de ser falsa é acessada. O usuário pode então decidir se quer acessá-la mesmo assim ou navegar para outra página. Há vários complementos para o Google Chrome e Firefox que bloqueiam as páginas suspeitas.

### DNS Poisoning

Infelizmente existem técnicas que burlam essas dicas, como o DNS Poisoning, onde o **cracker** infecta a máquina da vítima, alterando o arquivo HOSTS do Windows, que é responsável pelo redirecionamento de endereços IPs para as URLs que conhecemos. Por exemplo, o IP do Google é 173.194.42.137, se você digitar isso na barra de endereços do seu navegador, você será levado até o site do Google. Mas seria difícil gravar os IPs de todos os sites que conhecemos, e é para isso que serve o DNS, pra deixar os IPs mais *'bonitos'*.

O arquivo HOSTS do Windows fica localizado em `C:\Windows\System32\drivers\etc\hosts`, e nele contém o endereço IP do site, seguido do domínio referente à ele.

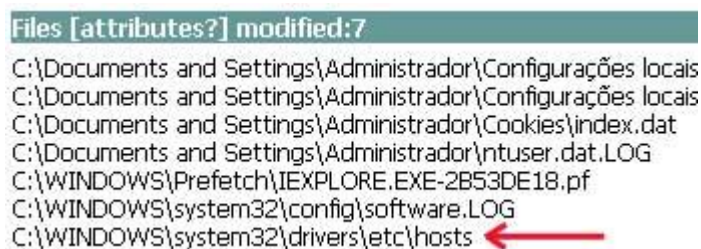
Se no final do arquivo você adicionar uma linha com o código:

`"127.0.0.1 google.com"`, você não conseguirá mais entrar no google pelo domínio google.com, porque ele será redirecionado para o servidor local (de IP 127.0.0.1).

Então, se for adicionado a linha:

`"xxx.xxx.xxx.xxxwww.bancodobrasil.com"`, Sendo `"xxx.xxx.xxx.xxx"` o IP da página fake, e o outro parâmetro o site do banco do brasil. Assim, quando você entrasse nessa URL, seria redirecionado para o site falso, e não perceberia, porque a URL não mudaria.

Abaixo um print de um malware que faz a alteração desse arquivo:



```
Files [attributes?] modified:7
C:\Documents and Settings\Administrador\Configurações locais
C:\Documents and Settings\Administrador\Configurações locais
C:\Documents and Settings\Administrador\Cookies\index.dat
C:\Documents and Settings\Administrador\ntuser.dat.LOG
C:\WINDOWS\Prefetch\IEXPLORE.EXE-2B53DE18.pf
C:\WINDOWS\system32\config\software.LOG
C:\WINDOWS\system32\drivers\etc\hosts ←
```

*Imagem: Acervo Pessoal*

A dica que dou para não cair nesses golpes é utilizar um bom anti malware (Recomendo o Malwarebytes), e sempre verificar o arquivo HOSTS antes de fazer login em alguma página importante. Basta executar o bloco de notas como administrador e arrastar o arquivo localizado em `"C:\Windows\System32\drivers\etc"` para lá, e verificar as últimas linhas. Se tiver o link de algum site lá, apague essas linhas imediatamente e faça uma varredura com o anti malware. Se isso continuar aparecendo, recomendo utilizar programas mais agressivos como o Combox. Se ainda assim não resolver, a solução é formatar.

Outras técnicas como a utilização de KL Banker, KL proxy e etc. são quase a mesma coisa, e as dicas que eu posso dar são as mesmas. Mantenha seu antivírus atualizado e tome muito cuidado com os sites onde entra. Sempre verifique o arquivo HOSTS.

### **Como os crackers clonam cartões?**

***O texto abaixo e dos próximos tópicos foi escrito por Hackuv e editado por mim. Créditos a ele =)***

Você está feliz, comprando algo com cartão de crédito, tudo certo. Ou quase isso. A fatura que vem no final do mês está com compras que você não fez. Compras em outras cidades, estados e até países. Você foi vítima, seu cartão foi clonado!

Os crackers conseguem os dados do seu cartão de crédito de duas maneiras.

A primeira é com um chupa cabra. O mesmo pode ser instalado em um terminal bancário ou até mesmo nessas maquininhas POS ou Pinpads que tem nas lojas. Os dados ficam salvos em um dispositivo Flash, e os estelionatários precisam ir "colher" os dados na loja, trocando os dispositivos e até em alguns casos, roubando os mesmos. Existem ocorrências que os chupas cabras tinham conexão Bluetooth e a distância se conseguia ter acesso a esses dados armazenados (falaremos mais sobre conexão via bluetooth no capítulo de redes). Lembrando que o chupa cabra funciona em três tipos de equipamentos: equipamentos POS, pinpads de TEF(*Transferência Eletrônica de Fundos*) e terminais bancários, e como é um sistema separado, não depende de nada. Pode ser instalado em qualquer leitora de cartão magnético.

Terminal POS:



*Imagem: MerchantDataSystems*

Chupa-cabra em terminal bancário:



*Imagem: Fraudes.org*

A segunda maneira é com malware TEF, que nada mais é que um Sniffer que atua na porta Serial(COM). Sniffer é um software que fica capturando todo o tráfego que passa em determinado local. Para ter acesso aos dados, ou os ladrões iam "colher", muitas vezes plugando um pendrive e o mesmo descarregava os dados para ele, ou o malware os enviava via email ou ftp ou qualquer protocolo de comunicação que esteja configurado, os dados fraudados. Lembrando que a técnica do vírus só se consegue fazer no TEF, pois necessita que os dados trafeguem por um computador antes de chegar na internet.

Terminal TEF:



*Imagem: PrismaCPI*

## Mas, como eles conseguem meus dados?

Muitos acreditam que é preciso inúmeros códigos e horas na frente do computador para burlar a segurança que existe em torno do site do Serasa, por exemplo. Um dos mais visados por conter muitas informações como CPF, RG, Filiação, Endereços, Telefones, Histórico Financeiro, etc. O Serasa, tem um critério muito rigoroso na hora de criar suas contas. Todas são pagas, e só para empresas.

Só que assim como outras, o Serasa também é uma empresa, e tem vendedores, representantes, distribuidores. Ai que as coisas ficam muito mais fáceis!

Mas o cracker não tem uma empresa. Então veja lá, uma rápida pesquisa no google: [CNPJ ENDEREÇO TELEFONE RAZÃO SOCIAL](#) e várias informações são retornadas:



Google

CNPJ ENDEREÇO TELEFONE RAZÃO SOCIAL

Web Imagens Mapas Shopping Vídeos Mais Ferramentas de pesquisa

Aproximadamente 5.450.000 resultados (0,25 segundos)

[PDF] [CNPJ RAZAO SOCIAL NOME FANTASIA ENDEREÇO BAIRRO ...](#)  
[www.tre-rs.gov.br/.../LISTA\\_TRE\\_nova\\_listagem\\_da\\_Green\\_Card\\_-\\_2801...](#)  
CNPJ. RAZAO SOCIAL. NOME FANTASIA. ENDEREÇO. BAIRRO. CIDADE. UF.  
TELEFONE. 95026480000176 IVO MARIO FERNANDEZ MACHADO ME.

[CBTU - Dados Cadastrais](#)  
[www.cbtu.gov.br/acbtu/cadastro/cadastro.htm](#)  
Dados Cadastrais. **Razão Social:** Companhia Brasileira de Trens Urbanos **CNPJ ...**  
Cep: 20221-901 **Telefone:** 21 3733-3399. Site: [www.cbtu.gov.br](#) ... Belo Horizonte.  
**CNPJ.:** 42.357.483/0005-50. **Endereço:** Rua Januária, 181. Bairro: Floresta

[PDF] [cnpj razao social nome fantasia endereco bairro cidade](#)  
[www.defesacivil.rj.gov.br/.../Rede\\_estabelecimentos\\_conveniados\\_Green...](#)  
CNPJ. RAZAO SOCIAL. NOME FANTASIA. ENDEREÇO. BAIRRO. CIDADE. UF.  
TELEFONE. 32500977001055 FLORESTA COMERCIO E INDUSTRIA S A.

[Quero encontrar pelo cnpj da empresa endereco, razão social e etc ...](#)  
[br.answers.yahoo.com](#) > ... > Negócios e Finanças > Empresas  
14/03/2008 - entre no site: [www.receita.fazenda.gov.br](#) acesso ao sitio da receita federal, depois clique em cadastros ai clique em cadastros **CNPJ** Pessoa ...

#Hackuv

*Imagem: Hackuv*

Viram? Acabamos de ter acesso a vários dados de empresas, usando apenas o Google. Mas o que vamos fazer com elas?

Como falei acima, o Serasa trabalha com distribuidores, então os crackers criam contas com nomes de outras pessoas e empresas, e tem acesso a toda a base de dados do Serasa. Mas nem tudo é um mar de rosas. Eles precisam adquirir créditos para suas consultas. Muitos pagam com cartões fraudados, outros com boleto, pois a senha não cai e não ocorrem muitos problemas judiciais depois...

Seus dados podem ser pégos em listas... Listas de todos os tipos. Faça um teste, digite seu email no google e veja se retorna alguma coisa. Pois é, certamente algo foi retornado, e é possível que ele esteja em alguma lista com vários outros emails. Esses vários emails são conseguidos com a invasão de sites, e seu email – juntamente com outros dados – está no banco de dados desse site, junto com os de várias outras pessoas que lá se cadastraram. Os crackers então salvam o banco de dados e extraem apenas as informações que são interessantes para eles, geralmente o email e a senha. Essas listas de email são vendidas à outros crackers especializados em enviar spam via email, e é por causa disso que todo dia você recebe mensagens e não sabe de quem é.

**Venho Lembrar que isso tudo é para conhecimento. O uso desta técnica é considerada falsidade ideológica pelo art. 229º Código Penal, e quem utiliza dessas técnicas para o mal está sujeito as penas da lei!**

Se me permitir, vou recomendar um filme sobre Kevin Mitnick, que na minha opinião, é o maior grey hat de todos os tempos. Ele usava mais da engenharia social para invadir, e já invadiu grandes empresas e órgãos governamentais sem ser pêgo.

O filme que eu recomendo é “Hackers - Operação Takedown”. Está disponível no youtube. Conta sobre algumas das invasões de Kevin, até sua captura pela polícia.

Hoje ele escreve livros e artigos sobre segurança de informações, dá palestras em diversos países e trabalha como consultor em segurança de sistemas.

Se você se interessar por engenharia social, ele tem dois livros, “A arte de enganar”, onde ele fala sobre como a engenharia social pode conseguir informações preciosas, e “A arte de invadir”, onde ele conta sobre sua experiência como cracker nos anos 90. Tem umas histórias bem interessantes nesse livro, e são até inspiradoras. Você pode procurar os dois em PDF por aí que vai encontrar.

Nos vemos no próximo capítulo, onde eu mostrarei por onde você pode começar a estudar no mundo hacker.

## 4. Por onde começar, o que aprender?

Uma pergunta que eu recebo bastante é essa. Usuários novatos que não sabem por onde começar, nem o que aprender primeiro.

Bom, essa pergunta é um pouco difícil de responder, não há uma resposta pronta para ela... Visto que o mundo hacker tem várias áreas, não tem como eu dizer para você começar por X ou Y. Se quiser focar na área de deface, terá que aprender a usar exploits, explorar falhas de programação, etc... Se quiser focar na parte de invasões à PCs, terá que aprender sobre malwares, e como deixá-los indetectáveis... Se quiser entrar na área do cracking, terá que aprender a programar em Assembly, entender sobre engenharia reversa, e sacar bem sobre a estrutura interna dos arquivos do Windows.

Acredito que esse livro te dará uma boa base para seus estudos, já que eu falo sobre as principais áreas do hacking. Assim, você poderá ver em qual área se encaixa melhor e focar seus estudos nessa área.

Na minha opinião, – *e agora estou falando por mim* – eu acho que de início os usuários novatos devem aprender a programar, porque isso vai dar a base de todo o estudo que eles precisam. Digamos que você queira invadir um site que está com uma vulnerabilidade de código no arquivo que faz a verificação de login.

Se você não sabe programar aplicações WEB, não vai saber explorar falhas no site, e não conseguirá efetuar a invasão.

Isso serve também para a área de invasão à computadores. Você cria um server de trojan e quer infectar uma vítima com ele, mas ele está sendo detectado pelo antivírus, e você precisa de um crypter para fazer com que ele torne-se indetectável. Se você não sabe programar, só vai conseguir o crypter comprando, e bem caro. Se você soubesse programar, teria construído um crypter indetectável, ou talvez até o próprio trojan indetectável.

Você deve estar se perguntando agora o que é programação, ou então já sabe, ou já ouviu falar, ou blablabla... Então nos veremos no próximo capítulo, onde eu falo um pouco sobre programação.



# 5. Programação



Nesse capítulo eu vou abordar a programação bem superficialmente, falando sobre ela, suas funções, algoritmo e lógica.

Bom, programação é basicamente a forma com que você vai conversar com o computador, e fazer com que ele interprete seus comandos, criando programas e códigos... “Como assim Nick?”

Veja, vamos supor que você queira encher o registro de um site com um IP qualquer. Você não vai ficar apertando F5 toda hora, né? Isso gastaria muito tempo, e tempo é dinheiro.

Então, para automatizar essa tarefa, você poderia criar um programa que ficasse clicando na tecla F5 automaticamente, a cada 0,5s. Então é só deixá-lo rodando e ele faz todo o trabalho para você.

Ok, esse exemplo ficou meio zoadado, mas acho que deu para entender.

Uma frase que gosto muito, e que até uso em minha assinatura no fórum é essa:

*“Quando aprendemos a ler, aprendemos a escrever.*

*Quando aprendemos a ouvir, aprendemos a falar.*

*Então, quando aprendemos a usar um computador, por que não aprender a programá-lo?”.*

Programação é uma coisa incrível, você faz literalmente o que quiser com ela, se dominar. Você pode tanto criar um programa simples que envia emails em massa para várias pessoas, como pode desenvolver um complexo script que consiga extrair informações bancárias de pessoas cadastradas em um site qualquer.

**“Ah Nick, legal... Mas como eu faço para começar a programar?”**

Bom, como eu disse, programação é a forma com que você vai conversar com o PC, então, para fazer isso, você precisa de uma linguagem né? Como se você estivesse conversando com um americano.

Você precisa saber inglês, para fazer com que ele te entenda, certo? Mesma coisa no computador. Você precisa saber a(s) linguagem(s) de programação para que ele te entenda.

A maioria das linguagens tem alguma similaridade na estrutura do código. Isso porque elas foram derivadas de outras linguagens antigas, chamadas linguagens-mãe.

Como quase todas são parecidas na sintaxe, é fácil de você entender superficialmente o código de qualquer linguagem, se dominar a lógica e algoritmo.

**“Ehh... Lógica?! Algoritmo??? O que p\*\*\*\* é isso?”**

Veja, para esclarecer um pouco antes de falar sobre algoritmo, vamos voltar ao exemplo das linguagens que usamos no dia-a-dia. Para você conversar com uma pessoa, você precisa organizar as palavras de forma a fazer sentido para quem vai ouvir, certo? Se eu chegar para você e falar “Ontem manteiga com comi pão” você não vai entender. Mas se eu organizar as palavras e falar na ordem, fazendo sentido e você vai me compreender, “Ontem comi pão com manteiga”.

## Algoritmo

Em programação, algoritmo é um esquema para resolver um determinado problema.

Vou dar um exemplo de algoritmo de um programa em PHP que verifica se um determinado valor é um número.

```
1. <?php
2.     $numero = 3;
3.     If(isint($numero)){
4.         echo "É um número!";
5.     }else{
6.         echo "Não é um número";
7.     ?>
```

Vamos entender esse código.

Na linha **1** e **7** nós estamos dizendo que o nosso código é na linguagem PHP, isso é padrão, em todas as linguagens terá algo que diga que o código inicia e termina.

Na linha **2** nós definimos uma *variável* chamada *numero*, contendo o valor 3. *Variável* é um espaço na memória usado para armazenar dados, qualquer tipo de dado, desde um numero até um texto.

Na linha **3** nós usamos uma estrutura condicional chamada *if*, que em inglês significa “se”, juntamente com a função “*isint()*”, que verifica se o valor digitado é um número inteiro. Nós falamos ao programa assim: “se o valor digitado for um número inteiro, então, faça isso:...”.

Na linha **4** nós definimos o que o programa vai fazer se a condição de cima for verdadeira. No caso, a condição de cima é verificar se o valor digitado é um numero inteiro. Então, se essa condição for verdadeira, ele vai exibir a mensagem “É um número!”.

Na linha **5** nós definimos o código que seria executado se essa condição fosse falsa. Nesse caso, esse código não seria executado, porque a condição é verdadeira (3 é um número inteiro).

Mas se o valor da variável \$numero não fosse um número, o código da linha 4 não seria executado, ele iria pular pra linha 5. O else significa “caso contrário”.

Então, traduzindo para o português, o código ficaria assim:

```
1. <?php
2.     Variável numero = 3;
3.     Se o valor da variável numero for inteiro, então:
4.         escreva “É um número!”
5.     caso contrário:
6.         escreva “Não é um número”
7.     ?>
```

Entenderam? Esse é apenas um dos conceitos das linguagens de programação. Se quiser se aprofundar em programação (e eu recomendo infinitamente que você faça isso), procure por cursos/apostilas de algoritmo e lógica de programação.

## Lógica

Supondo que você precise ir na casa do seu amigo, que fica na rua de trás. Para você chegar até lá, você precisa seguir uma série de sequências lógicas, como andar, virar à esquerda, seguir em frente, virar direita, virar na casa amarela e bater na porta, certo? Mas veja, há dois caminhos, o que eu disse acima, e o outro, em que você entra em um beco que tem na calçada e vai direto para casa dele, sem precisar virar em lugar nenhum.

Você segue uma linha reta, logo, percorre menos espaço. Qual caminho você prefere? Ir pela rua, dar voltas e andar mais, ou ir pelo beco em uma linha reta, andando menos?

Bom, se você é normal, escolheu a segunda opção.

É assim que você tem que pensar em programação, sempre faça a melhor e mais simples escolha possível, porque linhas e linhas de código dão dor de cabeça.

***“Será que essa é a forma mais simples de se fazer isso?”.***

*Pense nisso.*

## Tipos de conversão

Em programação, quando seu código está pronto para ser executado, você precisa salvá-lo e convertê-lo para um arquivo executável, certo? E essa conversão pode ser feita por compilação ou interpretação.

Se o método de conversão traduz todo o código do programa primeiro, para depois ser executado, dizemos que o código foi compilado, e o software que se responsabiliza por fazer isso é o compilador. Isso é útil pois o programa pode ser executado várias e várias vezes, sem precisar de uma nova compilação para cada execução, o que o torna mais versátil. Exemplos de linguagens compiladas: C, C++, Pascal, Visual Basic, etc.

Porém, se o método de conversão executa o código na medida em que ele é executado, dizemos que o código foi interpretado, e o software que se responsabiliza por fazer isso é o interpretador. Programas interpretados geralmente são mais lentos do que os compilados, mas são bem mais flexíveis, permitindo que eles rodem em várias plataformas. Por isso são chamadas de *script*. Exemplos de linguagens interpretadas: PHP, Perl, Python, Javascript, etc.

Linguagens como C e C++ são compiladas estaticamente, e seus códigos fontes são transformados diretamente em linguagem de máquina. Enquanto as linguagens mais modernas como Java, C# e Python têm seus códigos fontes transformados em uma linguagem intermediária (específica de cada linguagem), que será interpretada pela máquina virtual da linguagem quando o programa for executado.

## Níveis de programação

As linguagens de programação são divididas em 3 níveis, alto nível, médio nível e baixo nível.

As linguagens de **alto nível** são as mais simples, são as linguagens que todos começam aprendendo. Elas exigem menos do conhecimento de software do programador, porque trabalha com estruturas simples, e tem uma sintaxe bem amigável, fazendo com que todos possam entender. Vantagens: Podem ser executadas em várias plataformas sem grandes alterações no código. Desvantagens: Geram rotinas genéricas e complexas, portanto, ocupam mais espaço na memória. Exemplos de linguagens: PHP, C#, Visual Basic, Python, Perl, etc.

As linguagens de **médio nível** são um pouquinho mais complexas. Elas têm componentes de linguagens de alto nível e baixo nível, então são um meio-termo entre os dois. A sintaxe pode ser tanto simples quanto complicada, isso depende de como você vai programar. Vantagens: Ter mais poder sobre o computador, permitindo a criação de jogos e programas mais complexos, com qualidade profissional. Desvantagens: Alguns comandos são complicados para serem entendidos. Exemplos de linguagens: C, C++, etc.

As linguagens de **baixo nível** são bem mais complexas. São voltadas totalmente para a máquina, ou seja, são escritas usando instruções do processador do computador. Seus códigos são bem complexos e permitem que você execute instruções direto no processador. Vantagens: Programas são executados com maior velocidade de processamento, ocupando menos espaço na memória. Desvantagens: Os softwares não tem muita portabilidade. Um programa compilado em um determinado processador pode não rodar em um processador diferente. Exemplos de linguagens: Assembly, Cobol, etc.

Imagino que você entendeu sobre programação, mas ainda está em dúvida sobre qual linguagem escolher. Por isso vou mostrar abaixo algumas coisas que podem ser feitas, e as linguagens que você pode utilizar para fazê-las.

**Desenvolvimento WEB:** PHP, Javascript, ASP, JSP;

**Criação de Crypters (Capítulo 9):** VB6, VB.net, AutoIT, Pascal;

**Criação de exploits:** C/C++, Perl, Python, Ruby;

**Criação de Jogos:** Javascript, Python, C/C++/C#, HTML5;

**Manipulação de dados na memória e no processador:** Assembly, C++, Python;

**Desenvolvimento de aplicativos para smartphone:**

**Android:** Java, Javascript, PHP;

**iOS:** Objective C, Swift (Criada pela própria Apple);

**Programação desktop:** Todas citadas acima. C/C++, Java, VB e Pascal estão em alta no mercado;

## 5.1 Banco de dados

Todos nós já ouvimos falar sobre banco de dados. Seja na internet, no trabalho, ou com algum amigo. Mas você já parou para pensar para que serve e o quão importante é o banco de dados?

O que é um banco de dados? Bom, sabemos que um banco (agência bancária) é um lugar onde você pode guardar seu dinheiro de forma segura, e pode consultá-lo, sacar ou depositar quando quiser. Assim funciona o banco de dados na internet. Ele é um lugar na internet que armazena dados. Todo e qualquer tipo de dados. Desde números de fotos até cartões de crédito, ou nomes e senhas de usuários, ou os tópicos de um determinado site.

Você baixou esse livro no Fórum Guia do Hacker, mas isso só foi possível porque eu criei um tópico para o download, e o conteúdo desse tópico ficou armazenado no banco de dados do fórum.

Existem basicamente duas formas de se administrar um banco de dados:

- Centralizando todos os dados em apenas um banco de dados, ou seja, o banco fica apenas em um servidor, e contém todos os dados que devem ser acessados pelas aplicações e/ou pelos clientes, e como ele fica centralizado totalmente no servidor, requer mais capacidade de processamento do mesmo.

- Descentralizando os dados, dividindo-os em vários bancos e em servidores diferentes. São úteis porque cada banco exige apenas da capacidade de processamento do processador do servidor em que estão rodando, e como os dados estão divididos entre eles, cada um vai ficar com um pouco, não havendo sobrecarga.

Os bancos de dados devem ser gerenciados por algum lugar, e é por isso que existem os **Sistemas Gerenciadores de Banco de Dados (SGBD)**. Com esses softwares, é possível manipular os dados que são contidos no BD, podendo adicionar, remover, alterar, fazer consultas e etc. Um desses softwares, por exemplo, é o **MySQL**, que utiliza a linguagem **SQL (Structured Query Language)** como interface. É certamente o banco de dados mais utilizado atualmente, por conta da sua flexibilidade e facilidade ao trabalhar com seus comandos. NASA, Bradesco, HP, Nokia, Sony, Google, Cisco, todos esses gigantes utilizam o MySQL.

Uma falha bem comum e que foi explorada exaustivamente nos últimos anos é o SQL Injection. Essa falha permite que o invasor injete queries (*comandos*) SQL no servidor/programa devido a uma falha na validação de dados. Por exemplo, no formulário de login de um determinado site, o banco de dados pega o login e senha, e os coloca em um comando para verificar se os dados são reais. Mas, se os dados passados não forem validados antes, é possível passar comandos para o banco e fazer com que ele ache que ele deve fazer outra coisa. Você pode injetar um comando que retorne toda a lista de usuários e senhas, por exemplo. Veremos sobre essa falha e outras mais para frente no capítulo de vulnerabilidades.

Vamos falar sobre pentest agora, que tem muito a ver com isso. Até mais =)

# 6. Pentesting



Talvez seja a área mais cobiçada no mundo hacker. A área as invasões. Foi aqui que a fama do nome *hacker* se criou.

Traduzindo para o português, pentest significa “Teste de penetração”.

Mas não é essa penetração aí que você tá pensando não, safadinho... Penetração nesse caso se refere à intrusão em um sistema.

Esse teste serve para avaliar a segurança em determinado sistema, programa, computador ou até mesmo em uma rede. É possível conseguir quebrar a criptografia de uma senha do wi-fi do seu vizinho usando algumas ferramentas de pentest!

Mas lembre-se que isso é crime, e com a lei da Carolina Dieckmann, essa lei está mais frisada. Mas não vamos falar isso agora, vou dar-me a entender que todos testarão suas habilidades em ambientes controlados.

Existem profissionais que são pagos pra isso, os chamados Pentesters. São profissionais que são especializados em fazer auditoria no sistema em busca de falhas, vulnerabilidades, bugs, simulando sempre um ataque real para descobrir a ‘reação’ do sistema à determinada exploração.

Para que haja um bom planejamento antes do ataque, para que tudo seja feito corretamente, como planejado.

## **(Spoiler sobre GTA V abaixo)**

Você já jogou GTA V? Ou já viu algum vídeo da primeira missão?

Se não viu, eu explico. O personagem principal precisa de dinheiro para pagar uma dívida, e para conseguir esse dinheiro ele planeja assaltar uma joalheria. Dias antes do roubo ele entra nela com um óculos com câmera, para filmar as entradas de ar (por onde ele poderia entrar e sair despercebido), filmar os locais onde têm câmeras, e perguntar para a balconista o tipo de jóias que eles vendem (para saber se o roubo seria lucrativo). O resto eu não vou contar, essa é a parte interessante.

Os ladrões planejam dias antes do roubo, para que tudo dê certo. Imagine o tempo que os caras demoraram para planejar o assalto ao Banco Central.

Não quero passar uma imagem ruim do mundo hacker, até porque você já sabe bem a diferença entre um hacker e cracker. Mas se você quer saber como os crackers atacam, precisa saber de algumas coisas...

Bom, depois essa história toda foi só para explicar que no pentest, o procedimento é o mesmo. Não é só chegar e usar qualquer *exploit* (*explicarei nos próximos tópicos o que é exploit*) achando que vai funcionar. Você precisa estudar seu alvo, analisar o comportamento dele diante às várias situações.

Antes de falar sobre as etapas do pentest, vamos entender o conceito de exploit.



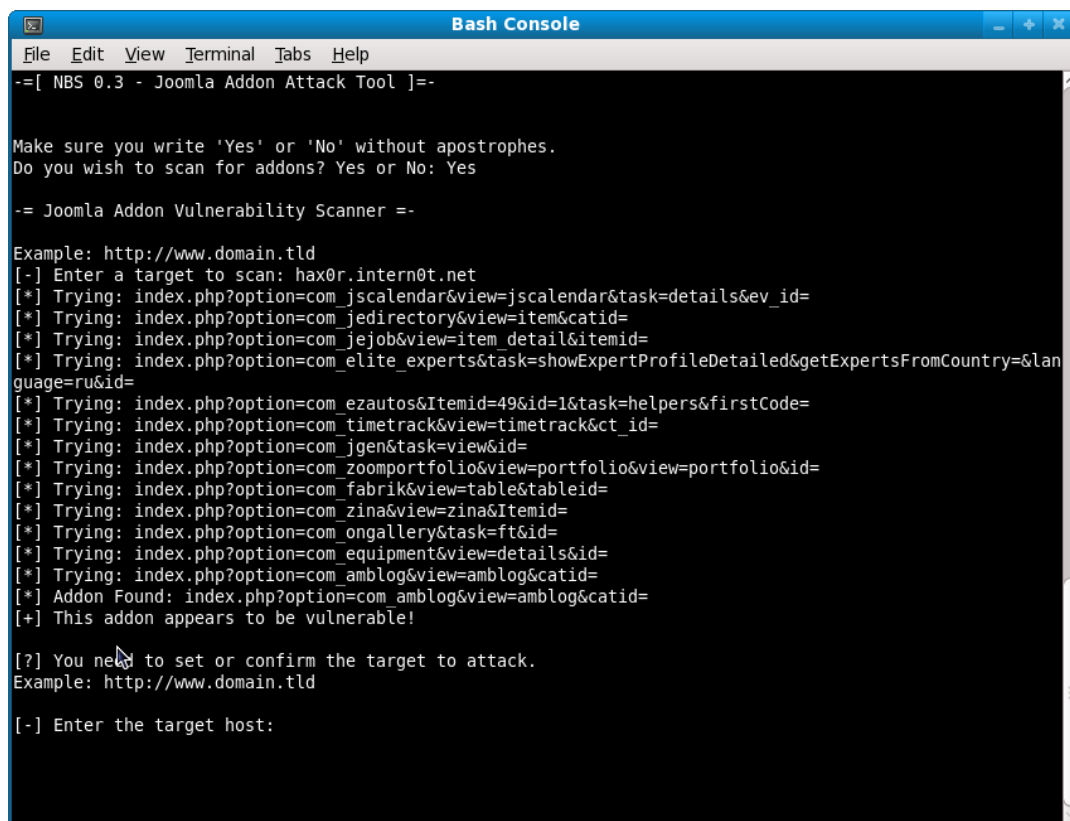
## 6.1 Exploiting

Por definição, exploit é um pedaço de código, programa ou uma sequência de comandos que se aproveita de vulnerabilidades no sistema para conseguir coisas como ganhar controle sobre um servidor, obter escalação de privilégio, ou atacar o servidor com *negação de serviço* (*Explicarei sobre negação de serviço (DoS) no capítulo de redes*).

Existem dois tipos de exploits, o remoto e o local. O local é o mais comum, ele é executado dentro do servidor da sua vítima, ou seja, é usado quando você já tem acesso ao servidor. Ele vai modificar o sistema em que ele mesmo está rodando.

Mas como conseguir acesso aos servidores? Aí que entra o exploit remoto.

Você pode se conectar ao servidor pelo exploit remoto e executar os comandos dentro dele, usando o seu PC. Esse tipo de exploit geralmente é usado para instalar *backdoors*, para que você mantenha acesso à ele. (*explicarei mais sobre backdoors na próxima secção*).



```
Bash Console
File Edit View Terminal Tabs Help
--[ NBS 0.3 - Joomla Addon Attack Tool ]--

Make sure you write 'Yes' or 'No' without apostrophes.
Do you wish to scan for addons? Yes or No: Yes

-- Joomla Addon Vulnerability Scanner --

Example: http://www.domain.tld
[-] Enter a target to scan: hax0r.intern0t.net
[*] Trying: index.php?option=com_jscalendar&view=jscalendar&task=details&ev_id=
[*] Trying: index.php?option=com_jedirectory&view=item&catid=
[*] Trying: index.php?option=com_jejob&view=item_detail&itemid=
[*] Trying: index.php?option=com_elite_experts&task=showExpertProfileDetailed&getExpertsFromCountry=&language=ru&id=
[*] Trying: index.php?option=com_ezautos&Itemid=49&id=1&task=helpers&firstCode=
[*] Trying: index.php?option=com_timetrack&view=timetrack&ct_id=
[*] Trying: index.php?option=com_jgen&task=view&id=
[*] Trying: index.php?option=com_zoomportfolio&view=portfolio&view=portfolio&id=
[*] Trying: index.php?option=com_fabrik&view=table&tableid=
[*] Trying: index.php?option=com_zina&view=zina&Itemid=
[*] Trying: index.php?option=com_ongallery&task=ft&id=
[*] Trying: index.php?option=com_equipment&view=details&id=
[*] Trying: index.php?option=com_amblog&view=amblog&catid=
[*] Addon Found: index.php?option=com_amblog&view=amblog&catid=
[+] This addon appears to be vulnerable!

[?] You need to set or confirm the target to attack.
Example: http://www.domain.tld

[-] Enter the target host:
```

Exemplo de exploit para Joomla. Imagem: Exploit-DB

Entendido o conceito de exploit, vou listar os 6 passos para que um bom pentest seja executado.

## 1. FootPrint e Fingerprint

Traduzindo de uma forma de fácil entendimento, footprint seria as 'pegadas', e fingerprint seriam as 'impressões digitais' do seu alvo. É conhecido também como levantamento de dados. E é com essas duas etapas que você vai estudar seu alvo, saber o que ele faz, tentar elaborar uma rotina para ele. Ex: Você sabe que o servidor faz backup toda sexta-feira às 19h, isso pode ser usado futuramente para algum ataque.

Esse levantamento de informações pode ser feito tanto usando softwares para isso, quando feito manualmente, por meio de engenharia social.

Em casos mais específicos, se o seu alvo for uma empresa, existe uma técnica chamada Trash Scouring, que foi usada muito por Kevin Mitnick. Falamos dele no capítulo de Engenharia Social. Trash Scouring é uma técnica que consiste em vasculhar o lixo de uma determinada empresa para obter informações que são jogadas fora, como documentos antigos, pagamentos de faturas, etc. Isso pode ser usado em uma engenharia social contra a empresa. ***“O lixo de uns é o luxo de outros”.***

## 2. Análise de rede e enumeração de serviços

Nessa etapa o pentester vai conseguir identificar quantas máquinas estão conectadas à rede, seus sistemas, portas abertas e os serviços que rodam nelas. *“Eh... Portas? Como assim?”*. Bom, esse assunto será mais detalhado no capítulo de redes. Porta basicamente é por onde a conexão será efetuada.

Imagine que você está na frente da casa de seu amigo, e precisa entrar nela para passar uma informação para ele. O que você faz? Você precisa da porta aberta para entrar na casa, se ela estiver fechada, você não entra, certo?

É mais ou menos assim na informática. Para você efetuar uma conexão com algum servidor, você e ele precisam estar com portas abertas.

E existem alguns serviços que rodam nessas portas. Sabe quando você vai entrar em algum site, e antes dele você coloca *“http://”*? Então, o HTTP (HyperText Transfer Protocol) é um serviço que roda na porta 80 dos sites. Todos os sites tem a porta 80 aberta. Se ela estiver fechada, o serviço HTTP fica indisponível, e você não consegue acessar o site. E vários outros serviços existem para que o servidor funcione, como o FTP (transferência de arquivos), SMTP (envio de email), etc. Isso será explicado detalhadamente no capítulo de redes.

## 3. À procura de vulnerabilidades

Depois de enumerar todos os serviços e portas abertas no servidor, é hora de procurar por vulnerabilidades nesses serviços.

A falha mais atual e fatal que eu conheço (até o momento em que estou escrevendo este livro) é o HeartBleed, que atinge o serviço de SSL (Security Sockets Layer), que é um serviço de segurança na transmissão de dados. Essa falha faz com que seja possível obter informações que o servidor envia/recebe.

Existem programas que fazem o scanneamento do servidor a procura de falhas, por exemplo o Acunetix, ou o Nikto, que são muito úteis para *defacers (explicarei sobre deface mais a frente)*.

Ou você pode tentar encontrar as falhas manualmente, inserindo parâmetros em alguns lugares do sistema para tentar encontrar uma reação inesperada por parte do servidor.

#### **4. Explorando a vulnerabilidade**

Após encontrar as vulnerabilidades, chegou a hora de explorá-las. A hora mais esperada do dia... Bom, para explorar uma vulnerabilidade, você pode usar dois métodos: O manual e o automatizado.

No método manual, você precisa ter mais conhecimento sobre o que está fazendo, precisa dominar o sistema da sua vítima para conseguir fazê-lo interpretar seus comandos.

E no método automatizado, você usa um exploit para fazer o serviço para você.

É aí que tá a utilidade da programação. Programando um exploit para uma determinada falha, você poupa o tempo de ter que fazer o mesmo processo cansativo toda vez que for invadir um sistema com aquela falha.

***“Mas eu não sei programar, como vou conseguir um exploit?”***

Vários sites disponibilizam bancos de exploits das falhas mais recentes que foram descobertas. Alguns são privados, os chamados Oday, e você só tem acesso à eles se comprá-los. Mas grande parte deles é gratuita. Sites como o Exploit-DB, 1337day, Rapid7, etc...

#### **5. Mantendo acesso**

Com o acesso ao servidor em mãos, você precisa fazer com que esse acesso se prolongue, se não o administrador pode perceber que foi invadido, corrigir a falha e bloquear as portas, e você não conseguirá se conectar novamente.

Para isso que foi criado o backdoor. Backdoor significa *“porta dos fundos”*, e é um *malware* criado para manter acesso ao computador da vítima. O backdoor abre portas escondido, sem que ninguém sabia, e fica aguardando conexão nessas portas. Assim, sempre que você quiser, você pode se conectar à ele, e o administrador não vai perceber, porque você estará usando portas que não eram usadas antes.

#### **6. Eliminação de evidências e evasão**

Todos aprendemos no primário que o prefixo “i” significa dentro, e o prefixo “e” significa fora. Então, se invasão é o fluxo para dentro, evasão é o fluxo para fora. É basicamente o abandono do local do *‘crime’* (*ainda estou levando em conta de que todos as invasões serão feitos em ambientes de testes*).

Mas não pode ser de qualquer jeito... Você precisa apagar seus vestígios antes, ou será pego pelo *webmaster*.

Existem arquivos chamados logs, e esses arquivos são responsáveis por registrar tudo que acontece dentro e fora do servidor. Se você acessa uma página de um site, o seu IP, junto com as informações da sua sessão, serão enviados para o arquivo de log, contendo a data do acesso à página.

É por causa desses arquivos que vários crackers são presos. Esquecem de limpá-los, e acabam deixando pegadas.

Mas não é só apagar o arquivo inteiro e pronto, se não, você vai deixar um buraco gigante, e qualquer um vai perceber que você esteve ali.

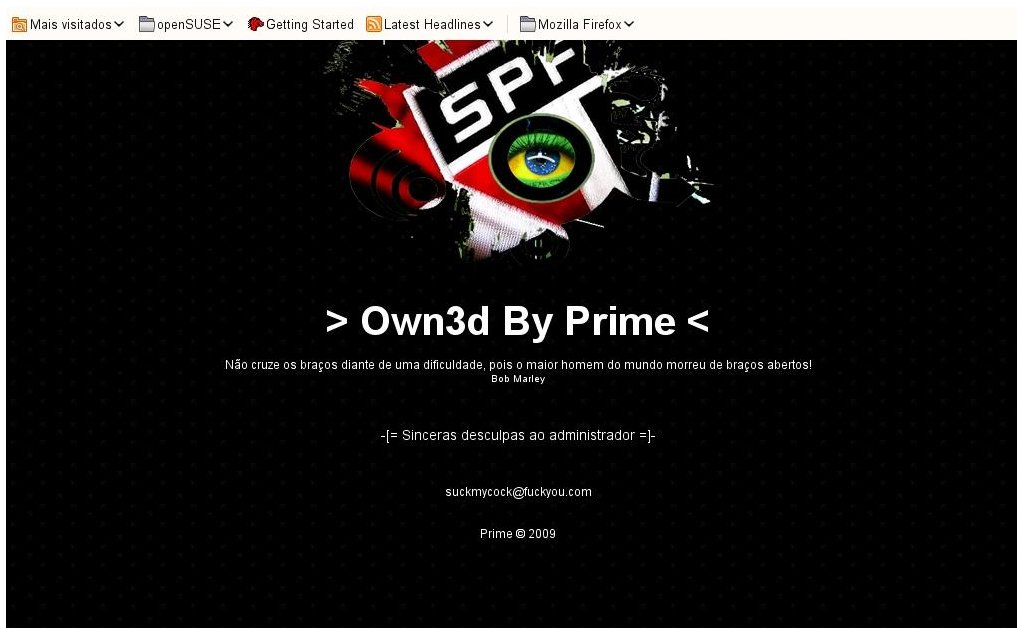
Você precisa apagar apenas as linhas dos logs referentes à seu IP. Para isso, procure pelo arquivo de logs, que geralmente fica na raiz do servidor, abra-o com algum editor de texto e procure pelo seu IP usando CTRL + F, apague todas as linhas que contenham seu IP e salve o arquivo. Depois disso será seguro sair.

**Mas cuidado, muitos servidores tem mais de um arquivo de logs, e fazem backups, então sempre procure pelas pastas...**

E esses são os 6 passos para um pentest bem sucedido. Claro que não ensinei como fazer o pentest passo-a-passo, eu expliquei apenas a composição desse ataque.

## 6.2 Deface

Deface é a arte de desconfigurar páginas de sites na web. Já entrou em algum site que estava com uma index diferente, com o fundo escuro, letras verdes, dizendo “hackeado”, com a foto da Anonymous? Pois é, esse é um exemplo de deface.



*Exemplo de um site invadido e desfigurado pelo Prime, antigo membro do GH*

O deface pode ser feito por três métodos principais, e estes se desenrolam em outros. São eles:

**Engenharia social** – Usando técnicas de engenharia social você pode conseguir acessos à informações que podem te levar a conseguir privilégios. Já dei o exemplo de quando eu fui hackeado no MSN via engenharia social. Daí é com você, vai da sua imaginação...

Você pode enviar um email se passando por um chefe para alguma pessoa que tenha um cargo menor pra conseguir informações, ou sei lá... Invente, mas não esqueça de planejar antes.

**Exploração à nível WEB** – A maioria dos defaces são feitos por esse método, talvez por ser o mais fácil, talvez por ser o mais comum. Nesse método você precisa conhecer bastante as linguagens que atuam na WEB, sendo alguns exemplos delas: PHP, ASP, Javascript, etc. E ter um bom conhecimento em banco de dados também.

Existem inúmeras falhas que atingem o nível WEB, por exemplo: SQL Injection, XSS, CSRF (Cross Site Request Forgery), LFI, RFI, LFD, etc... Essas são apenas algumas das mais comuns.

**Exploração à nível de servidor** – Por meio desse método você terá que atacar diretamente o servidor, como já foi explicado na seção principal de pentest.

Geralmente os serviços atacados são HTTP, FTP, SSH, MySQL, etc. Para isso, você precisará de exploits, que eu também já expliquei o funcionamento. Se você usar algum portscanner (programa que escaneia portas e serviços abertos), e ver que há um servidor FTP rodando no servidor, coloque o nome do servidor + versão + exploit no google. Ex: **“PureFTPD 3.2.2 exploit”**. No caso, PureFTPD é o nome do servidor FTP que está rodando. *(Todos esses nomes serão explicados no próximo capítulo: Redes)*. Na próxima seção nós falaremos sobre Google Hacking, uma técnica muito boa, onde se pode conseguir informações preciosas sobre os servidores, e que todos deveriam aprender.

Depois de obter acesso ao site, você deve hospedar uma shell nele. Shell é uma espécie de gerenciador de arquivos do hacker/cracker. É com ela que você pode alterar os arquivos, instalar backdoors, ler informações sobre o servidor, etc... As shells geralmente são desenvolvidas em PHP, e são hospedadas no site usando alguma função de upload presente no painel de administração.

Por exemplo, na página de criar uma nova notícia *(no painel de admin)* há a opção para fazer upload de uma imagem de capa para a notícia. Porém, ao invés de enviar uma imagem, o invasor envia a shell PHP, e então abre o caminho dela no servidor, e pronto.

### 6.3 Google Hacking

Com o Google Hacking você pode achar arquivos importantes dentro de site, páginas que deveriam ser secretas, lista de emails, lista de senhas, shells upadas, backups de banco de dados, e várias outras coisas que o Google indexa.

O Google tem vários recursos que facilita a vida do hacker. Por isso eu sempre digo quando me perguntam algo óbvio: **Aprenda a usar o Google, ele é seu amigo!**

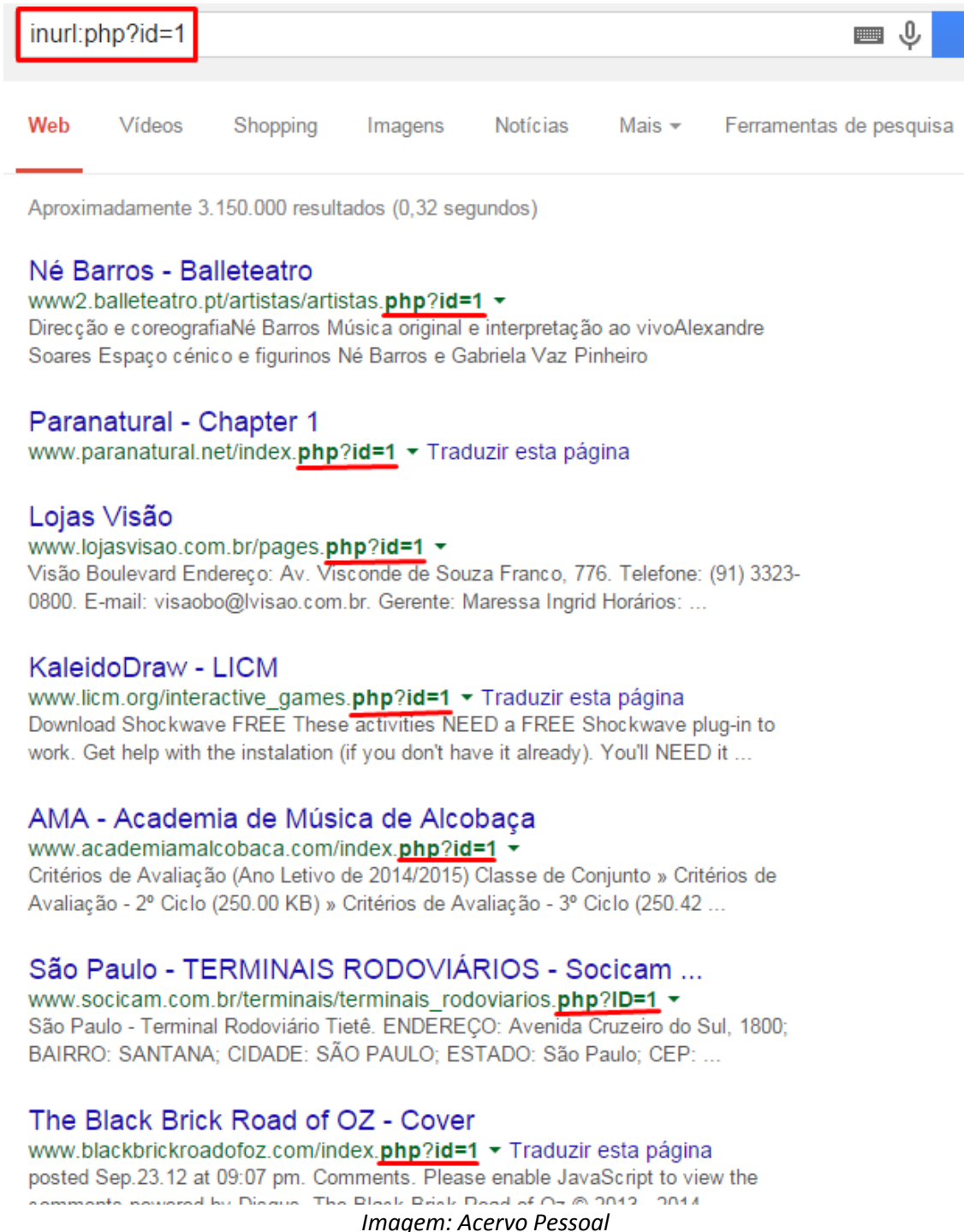
*(Quando digo “algo óbvio”, me refiro a alguma informação que pode ser facilmente adquirida com uma simples pesquisa no Google)*

Bom, um desses recursos que o Google oferece é a utilização de Dorks.

**“O que é essa coisa de dork, Nick?”**

Dorks são parâmetros que são passados na pesquisa para que o Google interprete melhor o que você quer e mostre os resultados de acordo com a pesquisa.

Quer um exemplo prático disso? Vamos digitar o seguinte comando no Google: **“inurl:php?id=1”**.



Notou algo diferente? Exato, todos os sites listados tem o parâmetro “*php?id=1*” na URL. Isso acontece porque você disse ao Google para listar apenas os sites que contenham esse parâmetro na URL.

### “E qual é a utilidade disso?”

Essa dork que eu passei pode ser modificada para encontrar falhas em sites. Uma falha muito comum é a SQL Injection, onde você injeta um comando no banco de dados do site para ele retornar a lista de usuários e senha. E esse comando é feito através da URL do site.

Com essa dork você pode achar sites que podem estar vulneráveis a SQLi, por que o id que está na URL é usado em uma consulta no banco de dados, então, se você alterar esse id para um comando qualquer, ele será executado pelo banco. Veja alguns exemplos de dorks:



**allintitle** [Faz uma busca de palavras que estão no título do site]

Exemplo: **allintitle:seja bem vindo**

*(Lista todas as páginas que contém o nome seja bem vindo no título do site)*

**allinurl** [Faz uma busca de arquivos/palavras na url]

Exemplo: **allinurl:index.swf**

*(Lista todas as páginas que contém o arquivo index.swf no site)*

**intext** [Faz uma busca específica de palavras que estão na página]

Exemplo: **intext:senhas**

*(Lista todas as páginas que contém a palavra “senha” no corpo)*

*(Tente pesquisar por **intext:seuemail@provedor.com**, e veja se há alguma lista com ele)*

**filetype** [Faz uma busca específica e retorna apenas o tipo de arquivo citado]

Exemplo: **filetype:txt**

*(Lista apenas arquivos txt, em qualquer site)*

Você pode combinar as dorks para fazer uma busca mais específica, por exemplo:

**intext:"PHPMysqlAdmin SQL Dump" filetype:sql**

*(Lista os arquivos .sql que contenham “PHPMysqlAdmin SQL Dump” no conteúdo. Nesse caso, dumps de bancos de dados inteiros serão exibidos, podendo ter algum conteúdo interessante nele)*

**@ +yahoo +hotmail +gmail filetype:txt**

*(Lista os arquivos .txt que tenham um arroba (@) seguido de “yahoo”, “hotmail” ou “gmail”. Ou seja, retorna listas de emails)*

E esses são apenas alguns exemplos de dorks, se tiver interesse, procure por [Google Dorks](#) no Google. *(Você pode usar dorks para isso, haha)*

Depois dessa explicação sobre deface e invasão, creio que surgiram algumas dúvidas relacionadas à rede. Vamos esclarecer algumas delas no próximo capítulo, onde eu vou explicar os principais protocolos usados. Até mais, te vejo lá!

# 7. Redes



Esse capítulo será um pouco extenso, porque eu vou tentar explicar sobre os protocolos mais importantes usados no dia-a-dia.

O conceito de rede de computadores é simples. Ela é formada por um conjunto de dois ou mais computadores conectados entre si. Podemos fazer uma analogia à uma rede de pesca. Suponha que cada nó da rede seja um computador, e este está se conectado com vários outros.



Vale lembrar que os computadores de uma determinada rede não precisam obrigatoriamente estar conectados à internet. Três computadores de uma pequena empresa podem estar configurados para trocar informações apenas entre si, e isso é uma rede.

## **Um pouco mais sobre portas**

Já falamos bem superficialmente sobre as portas, vamos aprofundar um pouquinho mais agora.

As portas variam de 1 até 65535, e são divididas em dois intervalos: as portas conhecidas e portas registradas.

As portas conhecidas estão entre 1 e 1023 e são usadas geralmente por serviços padrões do sistema, ou por programas executados por usuários com privilégios.

As portas registradas estão entre 1024 e 65535 e são usadas por processos comuns e por programas executados por um usuário comum.

Bom, vamos entender o conceito de protocolo de rede. O que é um protocolo? Falando de uma forma clara, protocolo é a linguagem na qual os computadores se comunicam entre si. Se eles não existissem, você não estaria lendo isso, porque não haveria internet para você baixar esse livro. Eles são responsáveis por te conectar à internet e fazer com que seu PC se comunique com outros.

***“Ahh, legal Nick, então isso é tipo uma linguagem de programação, mas para a internet, né?!”***

Sim, é como se fosse isso. A linguagem de programação é a linguagem que você usa pra se comunicar com o seu PC, e os protocolos de rede são as linguagens que os PCs usam para se comunicar entre si.

Existe uma pilha de protocolos chamada TCP/IP, e é a principal. Como o nome diz, é uma pilha, então são vários protocolos empilhados em uma sequência e em algumas camadas. Veja as 4 camadas da pilha de protocolos TCP/IP:

Camada	Protocolos	Função
Aplicação	HTTP, FTP, TELNET, POP3, SMTP, SSH, IRC, IMAP, DNS	Prover suporte aos programas e serviços para que você se conecte à internet
Transporte	TCP/UDP	Responsável pelo modo de envio dos pacotes
Rede	IP, ICMP, ARP, NAT	Controla a operação e endereçamento de pacotes
Física	Modem, RDIS, Bluetooth, USB	Converter pacotes em sinais eletrônicos e prover a conexão física com o PC

Eu espero muito que não esteja confuso, pois é fundamental você entender o funcionamento da rede.

Veja, vou comentar sobre alguns protocolos das 4 camadas.

### **Camada de Aplicação.**

**-HTTP (HyperText Transfer Protocol)** – Talvez o mais conhecido dos protocolos, é por causa dele que nos conectamos aos sites. Se você notar nas URLs dos sites, antes do link, tem escrito http://.

Se esse protocolo não existisse, todos os sites que você conhece hoje não existiriam, porque ele é o responsável por enviar os dados da requisição para o servidor, e retornar a página HTML para o seu navegador.

O método de conexão é bem simples. O computador cliente (*da pessoa que vê o site*) estabelece uma conexão com o computador servidor (*que armazena o site*), e envia uma requisição para ele contendo a URI (*caracteres usados para identificar protocolos*) e uma mensagem com as informações sobre o cliente, pedindo para que lhe seja retornado o conteúdo HTML contido no servidor.

O servidor responderá à essa requisição com uma *status line (linha de status)*, com a versão do protocolo usado e um código contendo o retorno da operação sendo esse código o de operação bem sucedida ou um código de erro. Depois dessa linha de status, a conexão entre o cliente e o servidor é interrompida, já que as informações já estão sendo exibidas no navegador do cliente. É por esse motivo que, se você abrir um site e desconectar da internet, ele continua aberto (*até você atualizar a página*).

**-FTP (File Transfer Protocol)** – Ahh, o bom e velho FTP... Ele é responsável pela transferência de arquivos entre o cliente e o servidor, e é bem mais rápida que a transferência feita via HTTP.

Quem aqui já procurou “como invadir um site” no youtube já deve ter se deparado com algum tutorial que usa esse protocolo como ‘ferramenta’. Na verdade, todos esses vídeos são falsos, você não invade o site de fato. *(Pelo menos não da forma que eles ensinam)*.

Talvez esse protocolo seja tão famoso quanto o HTTP, e é bem velho também.

Uma coisa que pouca gente sabe é que o FTP usa duas portas para se comunicar. A porta 21 é usada para sincronizar e a porta 20 é usada para a transferência.

Para que a conexão seja efetuada, o computador cliente *(que solicita a conexão)* precisa usar um programa de Cliente FTP, ou pode acessar o servidor diretamente usando um navegador WEB, onde será colocado as credenciais *(se necessárias)* para logar no servidor FTP. Com as credenciais aceitas, a conexão é efetuada e o cliente pode manipular arquivos no servidor.

Durante a transferência dos arquivos, duas representações para a transmissão de dados podem ser usadas *(na verdade são quatro, mas vou mostrar apenas as duas mais usadas)*:

**Modo ASCII**, onde o formato original dos dados é convertido em ASCII antes da transmissão, e convertidos novamente para a forma original depois.

Esse método é mais usado para a transferência de arquivos pequenos, e não é recomendado utilizá-lo para arquivos muito grandes.

Para quem não entendeu o que é ASCII, é um código para universalizar a transferência de dados pela internet, já que os teclados de alguns países têm acentos e caracteres especiais que não são usados em outros. Então tudo é convertido para ASCII para que não haja erros. É como se fosse o SI (Sistema Internacional), só que na informática :P

**Modo binário**, também conhecido como modo imagem, é o modo onde as máquinas transmitem os arquivos byte-a-byte, e o receptor vai armazenando em formato de pilha, até que todos os bytes sejam recebidos e o programa/arquivo possa rodar normalmente. Bem usado para o envio de imagens, executáveis e outros arquivos pesados.

**-DNS (Domain Name System)** – Se ele não existisse, talvez a internet não teria todo esse sucesso. Como eu expliquei na seção de página fake, o DNS é o protocolo que transforma os endereços IPs nas URLs dos sites que acessamos.

Vou usar o mesmo exemplo de antes. IP do Google é 173.194.42.137, então, se você digitar isso na barra de endereços do seu navegador, você será levado até o site do Google. Mas seria difícil gravar os IPs de todos os sites que conhecemos, talvez você conseguiria gravar uns 5 ou 6, mas são muitos números... E é para isso que serve o DNS, pra deixar os IPs mais ‘bonitos’.

Cada computador, ao obter um endereço IP e de Gateway, também obtém um endereço IP de um servidor DNS, sendo esse o responsável pela conversão, ou seja, toda vez que você digitar [www.google.com](http://www.google.com) no navegador, o servidor DNS vai converter para IP e assim você vai poder acessar o website. E cada vez que você entra num site, ele guarda as informações em seu cache, para quando você precisar novamente, não precisar fazer consulta à servidores DNS, economizando tempo e banda.

## Camada de Transporte

- **TCP (Transfer Control Protocol)** – Olha ele aí, o grande TCP. O mais usado, seguro, o top dos tops. O TCP é um ótimo protocolo porque é orientado à conexão, ou seja, quando uma requisição TCP é enviada, você tem certeza que ele vai chegar ao destinatário de forma correta, sem desvios, sem perdas. Todos os outros protocolos de aplicação usam ele na transmissão de dados. Ele se encarrega a quebrar os pacotes e depois remontá-los no destino, e, se um pacote estiver faltando ele se encarrega de pedir a retransmissão.

Os pacotes TCP têm campos chamados *flags*, que é onde ficam informações sobre o tipo de pacote. Por exemplo, uma requisição de sincronização tem a flag “SYN” ativada, enquanto que uma requisição de finalização de conexão tem a flag “FIN” ativada, e um pedido de reiniciamento de conexão tem a flag “RST” ativada. O computador que recebe os pacotes interpreta essas flags e faz o que é pedido.

Para que o TCP possa enviar um arquivo ou estabelecer uma conexão, ele usa um método chamado de Three-way Handshake ou seja um ‘aperto de mão’ em 3 vias. Preste atenção agora, porque isso será útil no próximo capítulo: DoS (*Negação de Serviços*).

Quando o cliente quer acessar o servidor Telnet na porta 23, ele envia um pacote SYN (*Synchronize(sincronização)*) para o servidor, pedindo para sincronizar com ele, e iniciar a conexão.

Cliente Servidor

**SYN + Porta 23**

----->

Então o servidor envia um pacote SYN/ACK (*Acknowledgement(reconhecimento)*) de volta para o cliente, dizendo que conseguiu sincronizar, aceitou o pedido de conexão e está aguardando a confirmação e o estabelecimento da conexão.

Cliente Servidor

**SYN/ACK + Porta (Acima de 1024)**

<-----

Apos o cliente receber o pacote com os flags SYN/ACK ativados ele envia um ACK (*Acknowledgement(reconhecimento)*) para confirmar o estabelecimento de conexão.

Cliente Servidor

**ACK + Porta 23**

----->

Traduzindo para o português, a ‘conversa’ entre os computadores seria algo assim:

- **Cliente:** “E aí servidor, vamos sincronizar nossos sistemas para iniciarmos uma conexão?”;
- **Servidor:** “Beleza, já reconheci que você quer conectar e já consegui me sincronizar com você, estou aguardando você iniciar a conexão”;
- **Cliente:** “Tranquilo, então eu confirmo a conexão. Já podemos trocar dados.”

Por isso o nome do método é Three-way Handshake, porque são feitos 3 verificações para ver se a conexão foi efetuada e se os arquivos foram passados. Por isso também que eu disse que ele é um pacote seguro, diferente do UDP, que veremos a seguir.

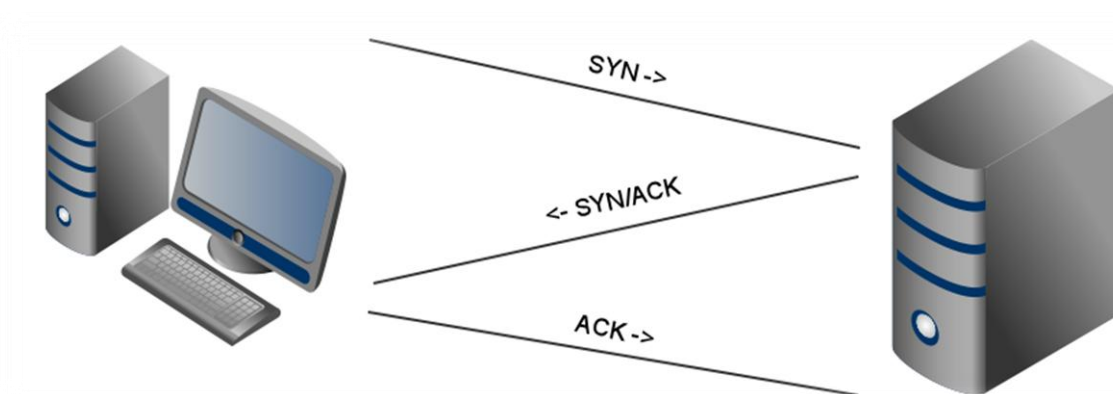


Imagem: Reddit

-**UDP (User Datagram Protocol)** – Esse é um protocolo bem mais simples que o TCP, e muito conhecido pela turma que gosta de fazer ataques de negação de serviço (*próximo tópico a ser abordado*), justamente por não ser orientado à conexão e não fazer a verificação se os dados foram enviados corretamente (*diferente do TCP*), então ele gasta menos tempo em cada pacote, e pode assim, enviar mais pacotes no mesmo período de tempo. Sendo assim, ele não precisa reenviar pacotes corrompidos e nem os confere. Alguns outros protocolos usam ele como o DNS e o DHCP.

Aí você diz: “**Mas Nick, se ele tem todas essas desvantagens, pra que usá-lo?**”

E eu respondo: **Velocidade.**

Imagine-se assistindo uma livestream no youtube, onde tem 70 mil pessoas conectadas, assistindo a mesma coisa. O servidor fica muito lento, porque a transmissão é em tempo real, e é muita gente acessando ao mesmo tempo. Se todos os pacotes enviados e recebidos fossem verificados, tal como os pacotes que se perderam no caminho, tudo ficaria lento e o serviço pararia de funcionar (Negação de serviço, novamente... rs). Aí que entra o UDP, ele é rápido, não faz verificação, então ele consegue transmitir todos os dados para todo mundo. Por isso acontece a perda de qualidade e travamento quando tem muita gente, já que ele não faz o reenvio de pacotes perdidos, o que se perdeu nunca vai chegar à você, então a imagem trava até receber outro pacote.



## Camada de Rede

- **IP (Internet Protocol)** – Também muito conhecido, o endereço IP serve para identificar o ser computador na internet. Muitos já disseram “*vou invadir seu PC pelo IP*”, mas não fazem ideia de como ele funciona.

O endereço IP é a nossa identificação na internet, como se fosse o nosso CPF, e este é um endereço único de 32 bytes, pois são 4 octetos ( $8 \times 8 \times 8 \times 8 = 32$ ) xxx.xxx.xxx.xxx, sendo sua faixa de atuação entre 0.0.0.0 e 255.255.255.255.

Lembre-se do exemplo da casa. Você precisa ir na casa de seu amigo, mas não sabe onde é. O que você faz? Exato, pergunta o endereço pra ele, só assim você consegue chegar até lá.

E é +/- assim que acontece na internet, os computadores (tanto clientes e servidores) precisam de um endereço IP para serem identificados e receberem as informações pela rede. O endereço de IP é único no momento, ou seja, o IP que você está usando agora é só seu, e de mais ninguém no mundo. Mas, se você desconectar, e outra pessoa conectar, ela pode receber o seu IP antigo.

Mas sempre em tempos diferentes, nunca no mesmo tempo, se não acontece um conflito, e os dois são desconectados para receberem novos IPs.

O IP se responsabiliza por fazer o endereçamento de pacotes, ou seja, ele é tipo ‘*a pessoa que coloca a carta no correio*’ da internet. Ele pega a sua requisição, com o seu IP, IP de destino e mensagem (*como numa carta, remetente e destinatário*), coloca tudo isso em um pacote e manda ele pro protocolo de envio (*TCP ou UDP*).

Atualmente o endereço IP está na versão 4, que é conhecida como IPv4 (*IP versão 4*), mas desde 2012, já está sendo usada a nova versão, o IPv6.

### ***“Mas Nick, pra que atualizar o IP se ele tá funcionando?”***

O IPv4 perdeu sua capacidade de expansão... São muitos computadores, servidores, notebooks, celulares, tablets, e outras coisas conectadas à internet 24h por dia. Como o endereço IP tem apenas 32 bits, todas as combinações resultariam em  $4 \times 10^9$ , que são **4.000.000.000** (quatro bilhões) de endereços. Mas veja, a Terra tem 8 bilhões de pessoas, e em uma casa há mais de um computador, tablet, celular... Fora os que não foram vendidos. Além das lan houses, e das salas com centenas de servidores.

Não é difícil de perceber que muita gente ficaria sem poder acessar a internet por não ter IP. É por isso que o IPv6 foi criado.

Ao contrário do IPv4 que tem 32 bits, o IPv6 tem 128 bits, e suas combinações resultariam em  $3,4 \times 10^{38}$  endereços. Pra quem quer ter uma noção do número de IPs, vou escrever: **340.000.000.000.000.000.000.000.000.000.000.000.000.000.000.000;**

Tentem achar um nome para esse número. Não é exagero, esse é realmente o número certo.

Além disso ele cobre todos os buracos de segurança que tinham no IPv4, onde os crackers faziam a festa. Isso combinado com o seu suporte obrigatório à IPSec (*Extensão do protocolo IP pra melhorar a privacidade, integridade dos dados e segurança*) certamente vai fazer os crackers terem dor de cabeça.

Bom, o IPv6 se destaca pois se difere bastante do IPv4. Veja a tabela com as diferenças:

IPv4	IPv6
Endereço de 32 bits	Endereço de 128 bits
Suporte opcional à IPsec	Suporte obrigatório a IPsec
Processo de fragmentação de pacotes feito pelo roteador	Processo de fragmentação de pacotes feito pelos hosts emissores
O cabeçalho do pacote inclui campos de opção	Todos os campos de opção foram mudados para dentro do campo <i>extension header</i>
O endereço tem que ser configurado manualmente	Funcionalidades de autoconfiguração

- **ICMP (Internet Control Message Protocol)** - Protocolo muito importante no diagnóstico da rede, porém é um protocolo que está sendo muito bloqueado devido ao seu mau uso por parte de usuários mal-intencionados. Hoje em dia, quase todos os firewalls ou roteadores já bloqueiam a entrada desse protocolo em suas redes, permitindo apenas em redes locais. O famoso Ping usa o protocolo ICMP, ou seja quando eu envio um ping à uma máquina, na verdade eu envio para ela o pacote ICMP Echo Request e espero o ICMP Echo Reply, assim posso saber que a outra máquina está ativa e funcionando. Antigamente existia um ataque chamado Ping Of Death, onde as pessoas tiravam serviços do ar enviando uma milhares de pacotes ping com tamanhos alterados (*Essa técnica é conhecida como Denial of Service, e será o tema do nosso próximo capítulo*), ou seja bem maior que o normal. Por isso, hoje em dia poucos são os hosts que respondem a esse protocolo.

A funcionalidade dele é apenas essa, mostrar se uma determinada máquina está ativa na rede ou não. Você envia um pacote ping (ICMP Echo Request) para uma máquina qualquer. Se ela estiver conectada à rede, e se o firewall dela não bloquear, ela responderá o seu pedido com o pacote ICMP Echo Reply, dizendo que está ativa.

### Camada física

- **USB (Universal Serial Bus)** – Um protocolo muito conhecido, o USB faz parte da camada física. E esse protocolo é visível, e está presente em nosso dia a dia. Você pode tocá-lo, fisicamente. Ele foi criado porque antigamente, quando você precisava instalar periféricos no computador, era necessário reiniciar a máquina, e isso não era uma boa ideia, por conta do número de conexões internas sendo realizadas.



Imagem: Wikipédia

Daí surgiu a tecnologia PnP (*Plug and Play*), onde o usuário não é obrigado a reiniciar a máquina, basta plugar o cabo e usá-lo.

O objetivo dele é fazer com que qualquer usuário sem experiência consiga instalar periféricos em sua máquina, sem dificuldades.

O USB padrão 1.1 foi criado em 1996, a 19 anos atrás, e tinha uma taxa de transferência de dados de 1,5Mbps, sendo alimentado por uma fonte de 5v, com 500mA.

As versões foram sendo lançadas ao longo dos anos, e em 2000 foi lançado o USB 2.0, que é o mais comum até hoje. O USB 2.0 tem uma taxa de transferência de dado de 480 Mbps (cerca de 60MB/s (Não confunda Mbps com MB/s), tendo como fonte a mesma do 1.0.

Mais recentemente, em 2009, foi lançada a versão 3.0 do USB, com incríveis 5GBps de transferência, e tem como fonte uma alimentação de 5V – 900mA, e pode enviar e receber dados ao mesmo tempo.

- **Bluetooth** – Também muito utilizado em aparelhos celulares e em dispositivos sem fio, o Bluetooth é um protocolo que usa a frequência de rádio de curto alcance para se comunicar com dispositivos próximos, conseguindo transportar informações de um dispositivo à outro sem a necessidade de cabos.



*Imagem: Wikipédia*

Ele está presente em celulares, mouses, teclados, fones de ouvido, notebooks, impressoras, câmeras digitais, consoles de videogames, entre outros dispositivos. O mouse que estou usando agora usa essa tecnologia!

Os dois dispositivos não precisam estar no mesmo campo de visão, só precisam estar no mesmo ambiente, e, dependendo da frequência de envio, o raio de alcance limite pode aumentar. 1m, 10m, 100m, isso depende da potência máxima usada.

O Bluetooth se encontra agora na versão 3.0, suportando transferências de até 24Mb/s.

Como a faixa do Bluetooth é aberta, ou seja, pode ser utilizada por qualquer dispositivo (*que tenha capacidade*), é necessário que o sinal do Bluetooth não sofra interferência, se não, não seria útil. Já aconteceu de você pegar o controle do quarto, ou de qualquer outro lugar, e usá-lo para desligar a televisão da sala? Pois é, isso acontece por causa da interferência. O dispositivo que está recebendo os comandos pensa que o controle que você usou é o original dele, e obedece suas ordens. O esquema de comunicação FH-CDMA (*Frequency Hopping - Code-Division Multiple Access*), que é usado pelo Bluetooth, faz com que a frequência seja dividida em vários canais.

O dispositivo que estabelece a conexão muda de um canal para outro de maneira bastante rápida. Este procedimento é chamado "*salto de frequência*" (*frequency hopping*) e permite que a largura de banda da frequência seja muito pequena, diminuindo sensivelmente as chances de interferência. No Bluetooth, pode-se utilizar até 79 frequências (ou 23, dependendo do país), cada uma "espaçada" da outra por intervalos de 1 MHz.

Bom, finalmente terminamos de falar sobre redes. Sei que achou o capítulo um pouco entediante por ser só teoria, mas é necessário entender como a rede funciona para entender o próximo tópico, sobre ataques de negação de serviço, os ataques que deixam os sites fora do ar. Vamos lá!

## 7.1 Negação de serviço (DoS)

Esse capítulo será muito interessante, porque é um tema bem polêmico e interessante, e certamente você vai querer saber como esse ataque funciona.

Bom, vamos entender primeiro o que é um ataque de negação de serviço.

Imagine que você acorda 7h da manhã para trabalhar. Pega o ônibus e vai para a estação esperar o metrô chegar. Tem muita gente com você, mas um grupo bem grande fura a 'fila', e passa na frente dos outros e o metrô já passa cheio. Quando ele para, todo mundo quer entrar ao mesmo tempo, até que o metrô fique sem lugar para você, e você não consiga entrar. Então ele fecha a porta e você não consegue sair dali, pois o metrô **negou prestar serviço** à você, já que estava congestionado.

Na informática é mais ou menos isso que acontece, quando você vê no jornal dizendo que "*hackers*" derrubaram o site da receita da fazenda, eles fizeram esse ataque. Enviaram diversas solicitações falsas, como se fossem reais, para que o site prestasse serviço a elas, até que o servidor fique sobrecarregado e não consiga dar conta de tudo. Então ele para de atender novos pedidos para priorizar os que ele já recebeu. Quando ele esvazia a memória e atende a todos os pedidos, ele volta a prestar serviços, e o site volta ao ar.

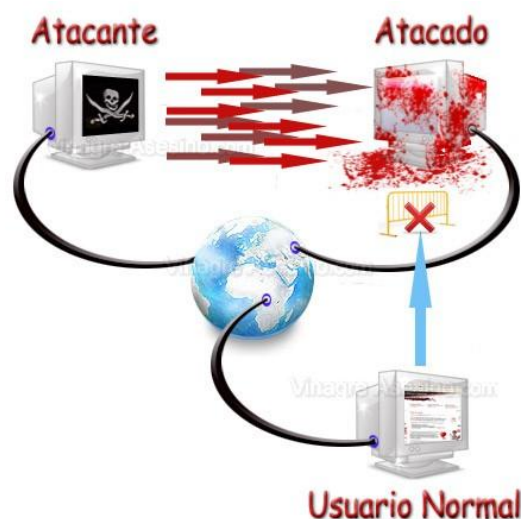


Imagem: DuckOpenSource

Os ataques de negação de serviço pode ser classificado em dois grupos principais:

**DoS (*Denial of Service*)** – É o ataque simples, onde apenas uma pessoa ataca o alvo. Não é tão comum hoje em dia por conta da sua baixa potência e eficiência.

**DDoS (*Distributed Denial of Service*)**– É o mais usado, onde muitos usuários (*cientes ou não*) atacam um alvo apenas.

**Como assim ‘cientes ou não’, Nick?**

Bom, existe um malware chamado Botnet, onde o cracker infecta várias vítimas, e usa o computador dessas vítimas pra atacar o alvo que ele quer. E isso é feito sem a vítima saber. Falaremos mais especificamente sobre esse malware no próximo capítulo.

### Tipos de ataques [D]DoS:

**Syn Flood** – O syn flood é um método de ataque de negação de serviço, onde o pacote usado para atacar é o TCP. Como nós falamos, o TCP tem um sistema chamado Three-way Handshake, ou seja, ele tem três etapas até concluir a sua conexão.

Revisando o conceito desse sistema, o cliente envia uma requisição com a flag **SYN** para o servidor, pedindo para sincronizar os dados. O servidor envia um pacote com a flag **SYN/ACK**, dizendo que já sincronizou e está aguardando a conexão, e então o cliente envia um outro pacote com a flag **ACK**, estabelecendo a conexão.

Então, se no meio do Three-way Handshake algum dos 3 pacotes não for enviado/recebido, ou se estiver corrompido, o servidor vai ficar esperando pelo envio de um novo pacote (já que o TCP faz isso automaticamente). Como o TCP é “*bonzinho*”, ele vai ficar esperando um novo pacote, mas, como nunca será enviado, vai ficar esperando por um tempo até fechar a conexão por time-out.

Agora imagine milhares de requisições incompletas ou não respondidas. São várias threads que estão se dedicando às requisições.

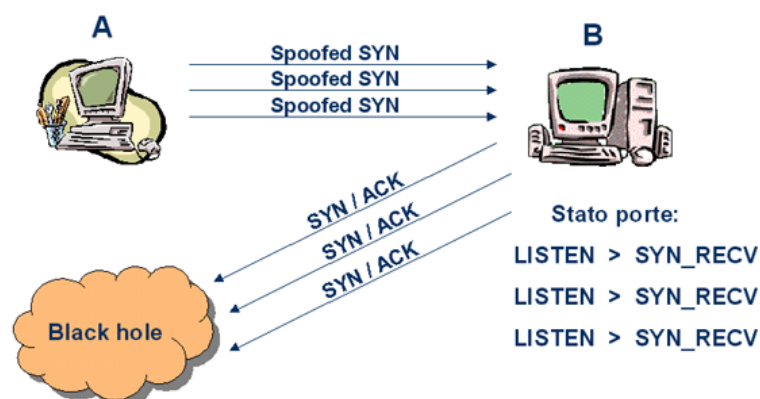


Imagem: Universidade Salerno

E é assim que o Syn Flood funciona, o cliente malicioso envia o pacote com a flag **SYN** pedindo sincronização, o servidor responde com **SYN/ACK**, mas o cliente não responde de volta, e então abre outra conexão diferente, fazendo a mesma coisa.

Então o servidor para de responder as novas requisições e continua processando as antigas, até que todas sejam processadas e o a memória esvazie. Só aí que ele volta a receber novas requisições.

**UDP Flood**– O UDP Flood é diferente do SYN Flood, já que o protocolo UDP não faz o procedimento de verificação dos pacotes (Three-way Handshake). Esse ataque é baseado na técnica de spoof, consiste em enviar requisições com o campo do IP alterado. Você vai enviar um pacote para o servidor se passando por outro computador. Nesse ataque, o cracker faz um pedido de conexão para a vítima, usando um IP falso.

Então, já que o protocolo UDP não oferece nenhum meio de garantia de entrega do pacote, o cracker simplesmente envia pacotes de UDP aleatoriamente para todas portas do servidor da vítima. Quando o servidor da vítima recebe os pacotes de UDP enviado pelo attacker, ele tenta determinar qual aplicação esta aguardando pelo pacote naquela determinada porta em que foi recebido o pacote. Porém, quando o servidor verifica que não existe nenhuma aplicação aguardando tal pacote recebido, ele emite um pacote ICMP Echo Request para o destinatário (que no caso forjou o endereço de IP da fonte) dizendo que o pacote não encontrou seu destino. Se uma grande quantidade de pacotes de UDP for enviado para as portas do servidor da vítima, o sistema poderá ser comprometido fazendo com que aconteça a negação de serviço.

**Smurf Attack** –O Smurf Attack é um ataque simples, porém eficiente, que se baseia na técnica de spoof.

No Smurf Attack, o cracker envia um ICMP Echo Request (Ping, como foi explicado na secção de redes) para todos os computadores de uma determinada rede, porém, esse pacote ICMP está spoofado, ou seja, o IP está forjado. E, como já vimos antes, quando você envia um ICMP Echo Request para uma máquina, se ela estiver online, responderá com um ICMP Echo Reply. Mas, como o IP do remetente foi alterado, o Echo Reply vai para o IP que está registrado no pacote, que é o IP da vítima.

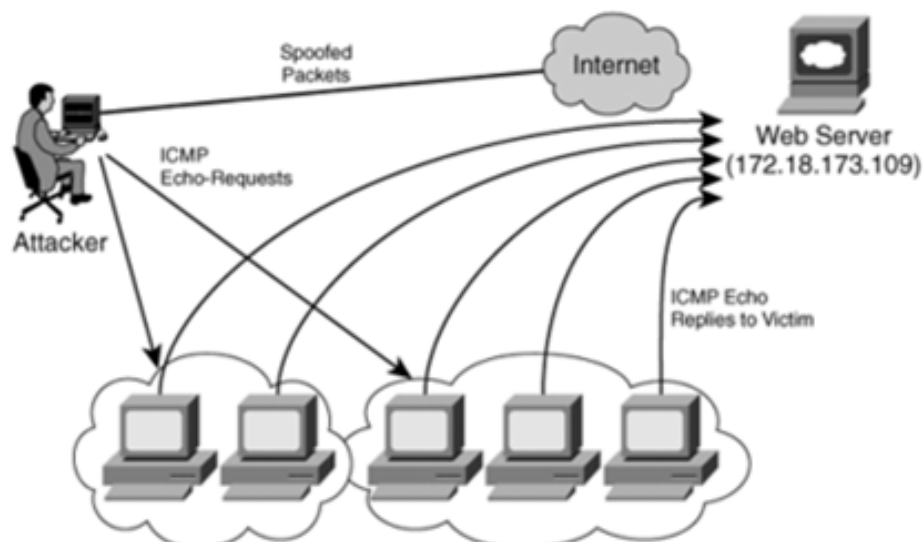


Imagem: InfoSecInstitute

Se você não entendeu, vou fazer uma analogia bem simples e garanto que vai entender:

*Veja, você quer deixar o seu colega X ocupado, então você envia uma carta se passando para todas as pessoas de uma família que tem 6 pessoas, pedindo para que elas respondam a carta. Como você enviou a carta se passando pelo seu colega X, todas as cartas, quando forem respondidas, serão encaminhadas a ele, e não a você. E ele vai ficar lotado de cartas, vai ter que ler todas, e vai ficar muito ocupado.*

Mas não pense que os ataques de negação de serviço acontecem apenas pela internet. Um computador pode dar crash também, se for submetido a um exploit que faça isso. Por exemplo, no Windows XP existe uma falha chamada Buffer Overflow, que permite que você consiga inserir trechos de código no próprio Windows.

Se você inserir um código de loop infinito, que faça com que ele retorne sempre para a mesma operação, o computador vai ficar sem memória e vai acabar exibindo a BSOD (Blue Screen Of Death, a famosa tela azul da morte).

Ataques [D]DoS também podem ser feitos no mundo real. Por exemplo, você – por algum motivo pessoal – pretende fazer com que uma determinada empresa não consiga atender seus clientes pelo serviço de suporte.

Então, você e várias outras pessoas fazem ligações para a central de atendimento, uma ligação após a outra, fazendo com que sempre que alguém de fora tente ligar, a ligação retorne ocupado. Isso aconteceu na Rússia, onde uma quadrilha de assaltantes ocupou o serviço de atendimento da polícia local enquanto assaltavam uma joalheria. Ninguém conseguiu chamar a polícia, e quando conseguiram, já era tarde demais.

***Ps: Não façam isso!***

Mas, para atacar um determinado site com vários computadores, você precisaria ter vários amigos que colaborassem com você no ataque, certo? ERRADO! Lembra quando eu disse que os atacantes poderiam ser cientes ou não do ataque? Pois é, aí que entra a botnet. Basicamente é um malware que se infiltra no computador das vítimas e realiza ataques de negação de serviço. Falaremos sobre ele e sobre os outros tipos de malware no próximo capítulo. Te vejo lá!

## 8. Malwares: Tipos e características



Acredito que todos aqui já ouviram falar em vírus, ou cavalo de tróia. Eles são malwares, bem diferentes, com funções específicas cada um. De maneira geral, todos falamos que o computador foi infectado por vírus. No entanto, existe uma série de outros tipos de ameaças, que são diferentes dos vírus e causam males diversos.

Nesse capítulo nós vamos aprender sobre os tipos de malware, tal como suas características, métodos de infecção e seus objetivos no sistema infectado.

Primeiro vamos para a definição de malware.

A palavra *malware* é originada do inglês, que significa “*Malicious Software*”, que significa software malicioso. E como o nome já diz, é um software que se instala no computador da vítima sem ela saber, e que tem fins maliciosos, como danificação dos arquivos, roubo de dados, roubo de senhas, propagação em massa, exploração do sistema, entre várias outras funções. Agora vamos para os tipos de malwares e suas características.

**Vírus:** Os mais conhecidos. Não pela sua função, mas sim por seu nome. A maioria dos malwares é considerada vírus pelos leigos, justamente por não saberem que eles são divididos em classificações. Nunca confunda vírus com malware. Todo vírus é um malware, mas nem todo malware é um vírus.

Bom, a função dos vírus é infectar arquivos do sistema e se multiplicar, como se fosse um vírus biológico. Ele ataca os arquivos e pode conter um código malicioso dentro dele. O mais conhecido é o vírus “I Love You”, com o nome original de “*Love-letter-for-you.txt*”, que após sua execução, enviava uma cópia de si mesmo para todos da lista de email da vítima. O vírus foi enviado para mais de 84 milhões de pessoas, e casou um prejuízo de 9 bilhões de dólares. Esse vírus teve tanto sucesso por usar **engenharia social** em seu nome. Com o nome de “eu te amo”, qualquer um ficaria curioso para saber quem era o remetente da mensagem, e certamente abriria o email, sendo infectado. Abaixo um print do email em que o vírus era contido.



Imagem: CRN



**Worms:** Os worms são uma espécie de subconjunto dos vírus, porém eles contêm algumas diferenças fundamentais em relação aos vírus. Basicamente, um worm é um programa que consegue fazer cópias de si mesmo sem infectar outros arquivos. A ideia dele é de instalar-se uma vez apenas no PC e, a então, procurar uma maneira de conseguir se espalhar para outros computadores.

Outra diferença é que, ao invés de querer permanecer não detectado, o worm cria uma instância única do seu código e permanece sozinho, já que ele procura brechas no sistema operacional infectado e garante que só ele vai rodar na máquina, evitando a infecção por outra ameaça. Isso quer dizer que o worm é um arquivo separado, que não se adere a arquivos existentes (*procedimento realizado pelo vírus*).

Esta estratégia utilizada pelo worm facilita o seu spreading (*propagação*) através de dispositivos USB e até mesmo em redes de computadores. Outra técnica utilizada pelos worms, e que é muito eficiente, é a distribuição de si mesmo através de e-mails, nos quais são criados anexos infectados. Estes e-mails são enviados para toda a lista de contatos da pessoa que teve o seu computador infectado, como no vírus I Love You, e a vítima nem sabe que isso está acontecendo.

**Keyloggers:** São usados para a captura de teclas no computador da vítima.

Os primeiros tipos de keyloggers eram peças físicas que eram conectadas entre o computador e a saída do teclado, como se fosse um adaptador, e o cracker tinha que inserí-lo no computador da vítima, ou seja, precisava de acesso físico ao PC para conseguir instalá-lo. Além de uma boa engenharia social, o cracker precisaria ter acesso ao PC novamente, já que ele teria que remover o equipamento depois que a vítima tivesse digitado o que ele queria. Abaixo uma foto de um keylogger físico:



*Imagem: Keelog*

O outro tipo de keylogger – e mais comum – é o software. Um programa que o cracker configura e tem a mesma finalidade do keylogger físico. A diferença é que este pode ser controlado remotamente, e tem variantes.

Quando infectada, a vítima tem seu teclado monitorado, e tudo que for digitado será capturado pelo keylogger, que enviará os dados para o cracker, geralmente via e-mail.

Veja abaixo o print de um keylogger feito em VB6:

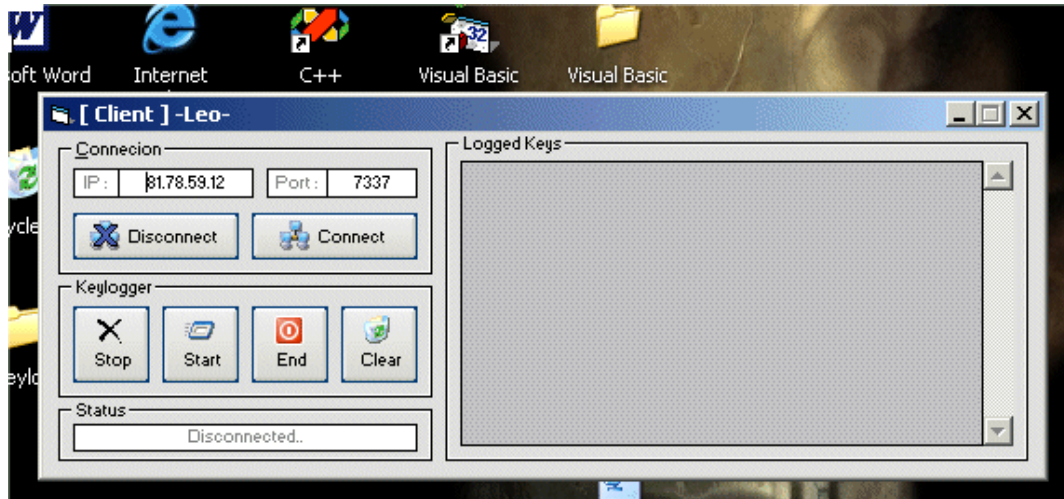


Imagem: Planet Source Code

Mas aí surge uma dúvida e você pergunta: ***“Mas Nick, as pessoas escrevem muitas coisas por dia, o cracker não levaria tempo para conseguir encontrar o que quer, diante de tanto texto?”***

Sim, e é por isso que foi inventado o Smart Keylogger, o keylogger que só pega o que é interessante para o invasor, como senhas, emails, etc...

Os Smart Keyloggers são mais conhecidos no mundo banker(*mundo dos fraudadores de dados bancários*), pois ficam escondidos no PC da vítima em estado de *listening*(*escuta*), aguardando que um site de banco seja aberto. Assim que a vítima digita a URL do site no navegador, o keylogger entra em atividade e fica aguardando pelo pressionamento das teclas. A vítima digita a senha, o keylogger captura a senha, e quando o site do banco é fechado, ele para a sua atividade e volta ao estado de escuta. Veja abaixo um print de um keylogger banker aberto na IDE Delphi de programação:

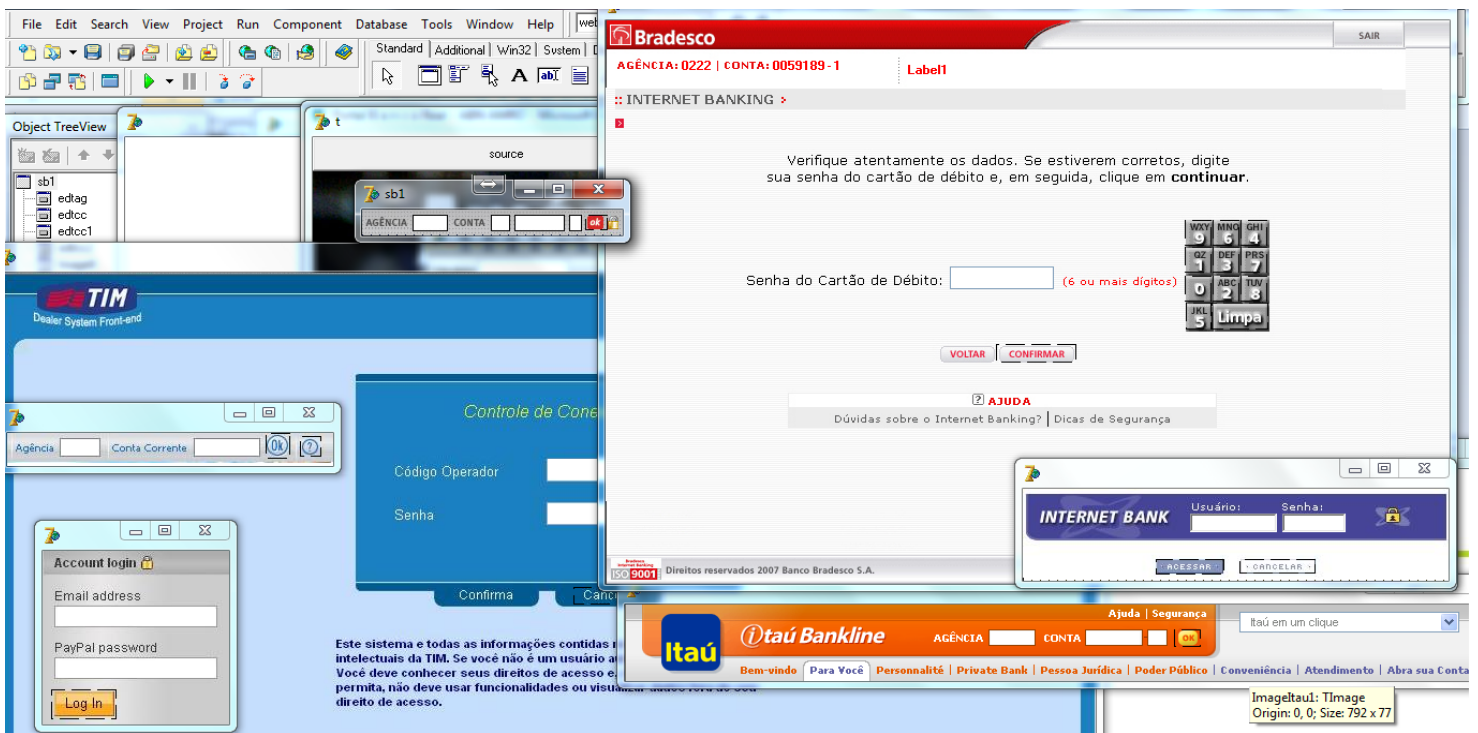
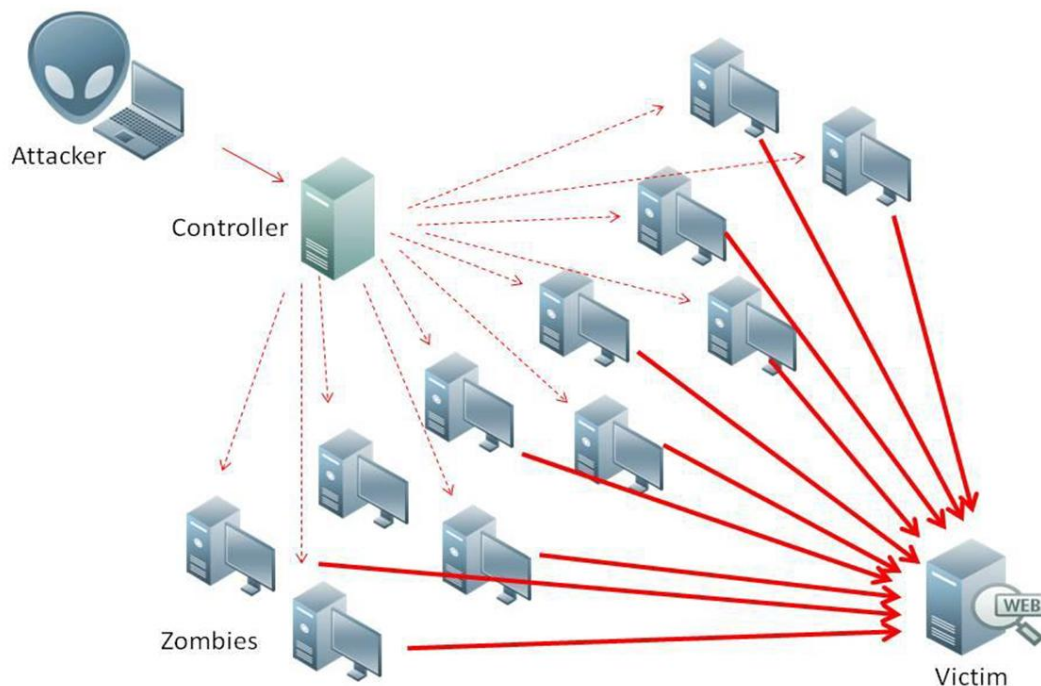


Imagem: Hackuv

**Botnets:** Já citadas anteriormente no capítulo de ataques DoS, a utilização de botnets é a principal forma de ataque de DDoS. O cracker envia o malware para várias pessoas - podendo ele ter a capacidade de se propagar, como um vírus - até formar uma rede, chamada de rede zumbi. Ele então tem controle sobre todos os *zumbis* infectados, e pode ordenar que eles iniciem ataques contra uma ou várias vítimas. Na ilustração abaixo você pode ver como funciona esse ataque. No caso, o **“Controller”** é o cliente da botnet que fica no PC do cracker.



*Imagem: NsFocusBlog*

Basicamente o cracker envia um comando para todos os infectados, e todos interpretam o mesmo comando, ao mesmo tempo. Se ele ordenar que todos ataquem o site X, todos farão. E isso é útil para ele, porque assim ele não precisa se preocupar em usar a própria banda de rede para fazer o ataque, já que está usando computadores escravos para fazer isso por ele. Então, a conexão dele com internet permanece estável, enquanto que a conexão dos zumbis fica oscilando.

**Backdoors:** São ferramentas simples que se acoplam aos malwares mais complexos. Sua função é abrir uma brecha no sistema da vítima, para garantir que o cracker consiga conectar lá sempre, ou seja, manter acesso. Os backdoors são usados em invasões à servidores WEB, como nós vimos no capítulo de **Pentest**, onde o cracker hospeda o malware no site da vítima, e sempre que quiser, pode conectar à ele, sem que a vítima saiba. Ele atua geralmente abrindo portas no firewall, ou utilizando de alguma brecha em algum programa ou serviço que se conecta a internet.



Imagem: TeeSupport

Mas o contrário também acontecer, você pode ser hackeado – indiretamente – por um site. O cracker pode hospedar arquivos maliciosos em alguma linguagem WEB, e esses arquivos exploram falhas existentes no navegador que você estiver usando. Há uma falha de buffer overflow no Internet Explorer 7, onde um script mal intencionado faz com que ele dê crash. **(Isso é um ataque de negação de serviço (DoS)!**)

**Rootkits:** Eles são bem complexos, geralmente são implementados em outros malwares para que fiquem indetectáveis pelo usuário. Sua função basicamente é usar técnicas que fazem com que o sistema não consiga perceber sua presença na máquina.

Por exemplo, se você estiver infectado com um determinado rootkit, e abrir o gerenciador de tarefas para ver se ele está executando, antes de o gerenciador ser aberto, o rootkit intercepta esse pedido e filtra os programas que aparecem na lista, excluindo ele mesmo dela. Assim, quando você olhar a lista, ele não estará lá, porque o programa gerenciador de tarefas foi modificado por ele para que ele não seja exibido na lista. Essa técnica é conhecida como API Hooking, onde você usa uma função do Windows para modificar ela mesma, em tempo real.

No fluxograma abaixo é possível ver como um rootkit pode alterar o gerenciador de tarefas, mostrando apenas os processos que não tem relação com ele:

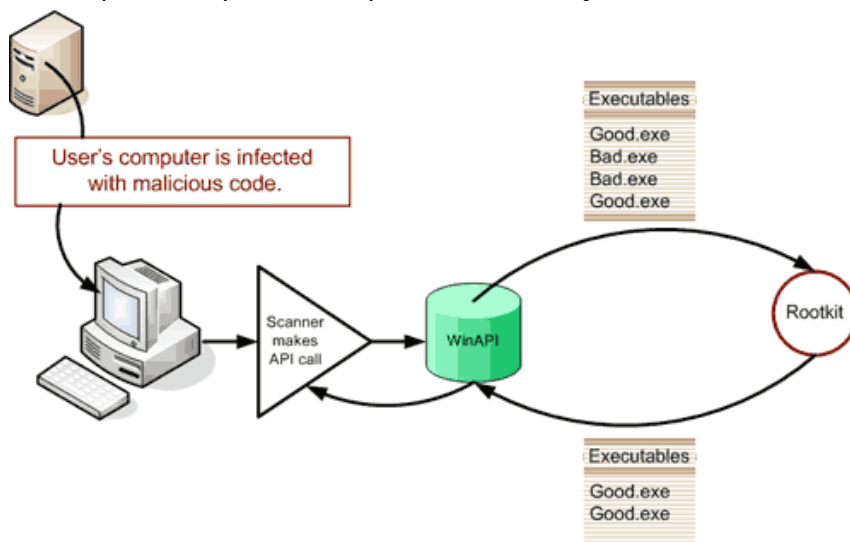


Imagem: Infolab-GR

## “Mas Nick, se eles se escondem no sistema, o que eles fazem de danoso?”

Como eu disse, eles são geralmente implementados em outros malwares para fazer com que eles sejam indetectáveis pelo sistema. Um rootkit pode se juntar à uma botnet, assim, o cracker terá sempre acesso à vítima.

**Ransomwares:** Não são muito conhecidos no Brasil, são mais utilizados na Europa, mas com certeza são os piores que existem. São terríveis!

Os ransomwares são malwares que se instalam no PC da vítima, e usam um algoritmo criptográfico que faz com que o PC se torne utilizável, e pede uma certa quantia em dinheiro para o resgate da senha até um certo tempo. Ou seja, uma vez infectada, a vítima não poderá mais utilizar o computador, pois todos os arquivos lá presentes estarão criptografados, e só o cracker tem a senha para descriptografar. Essa senha só será entregue caso a vítima faça o pagamento que o cracker deseja, caso contrário, a senha será destruída, e o PC não poderá mais ser usado.

O último RansomWare que eu tenho notícia é o CryptoLocker, que atacou algumas vítimas no Brasil no ano passado (2013). Ele se propagava por meio de uma engenharia social via email. O email falso se passava por uma companhia que estava oferecendo uma proposta de trabalho, e o executável estava anexado à mensagem. Nele, os crackers pedem a quantia de 100 dólares / 100 euros para que a senha seja enviada e o PC seja destravado. Caso contrário, em um determinado período de tempo, a chave de desbloqueio seria apagada, forçando a vítima a formatar o computador.

Print de um PC infectado com o CryptoLocker:



Imagem: Youtube



**Trojans:** Aí estão eles, os grandes, os poderosos, os mais conhecidos. Pensou que eu tinha esquecido deles? Deixei-os para o final. São os famosos cavalos de tróia.

Mas você sabe por que ele recebe esse nome? Bom, imagino que você saiba a história do cavalo de tróia, mas se não sabe, procure no Google.

Os trojans funcionam como o Cavalo de Tróia, são enviados para que se passem por outro programa, ou imagem, ou vídeo, e quando são executados, ocorre um processo de extração, onde o ele se instala no sistema, sem a vítima saber, e lá ele fica.

O trojan é dividido em dois programas, o cliente e o servidor. O cliente é o painel das vítimas, que fica sob controle do cracker. É nele que todas as ações são executadas. E o servidor é o malware em si, é ele que é executado pelas vítimas. São normalmente chamados de Client e Server, respectivamente.

Quais as funções do trojan? Os trojans atualmente contam com muitas funções, há alguns que contém todas as funções dos malwares citados acima, mas normalmente eles são usados para espionagem, captura de informações digitadas, manipulação de arquivos.

Você faz qualquer coisa com um trojan, tudo o que quiser. Capturar senhas, ver webcam, enviar pop-ups, abrir sites, baixar/enviar/apagar/criar arquivos, desligar o pc, inverter os botões do mouse... Até formatar o PC da vítima você consegue.

Existem dois tipos de trojans, os de conexão reversa e os de conexão direta.

**Conexão Reversa:** São os mais comuns, é o trojan em que o server se conecta ao client. Todas as vítimas ficam tentando se conectar ao cracker, e quando conseguem, ficam no aguardo dos comandos, para que possam executá-los. Dessa forma, o cracker consegue se comunicar com todos os servidores ao mesmo tempo, e executar tarefas simultaneamente.

Spynet, trojan de conexão reversa mais utilizado:

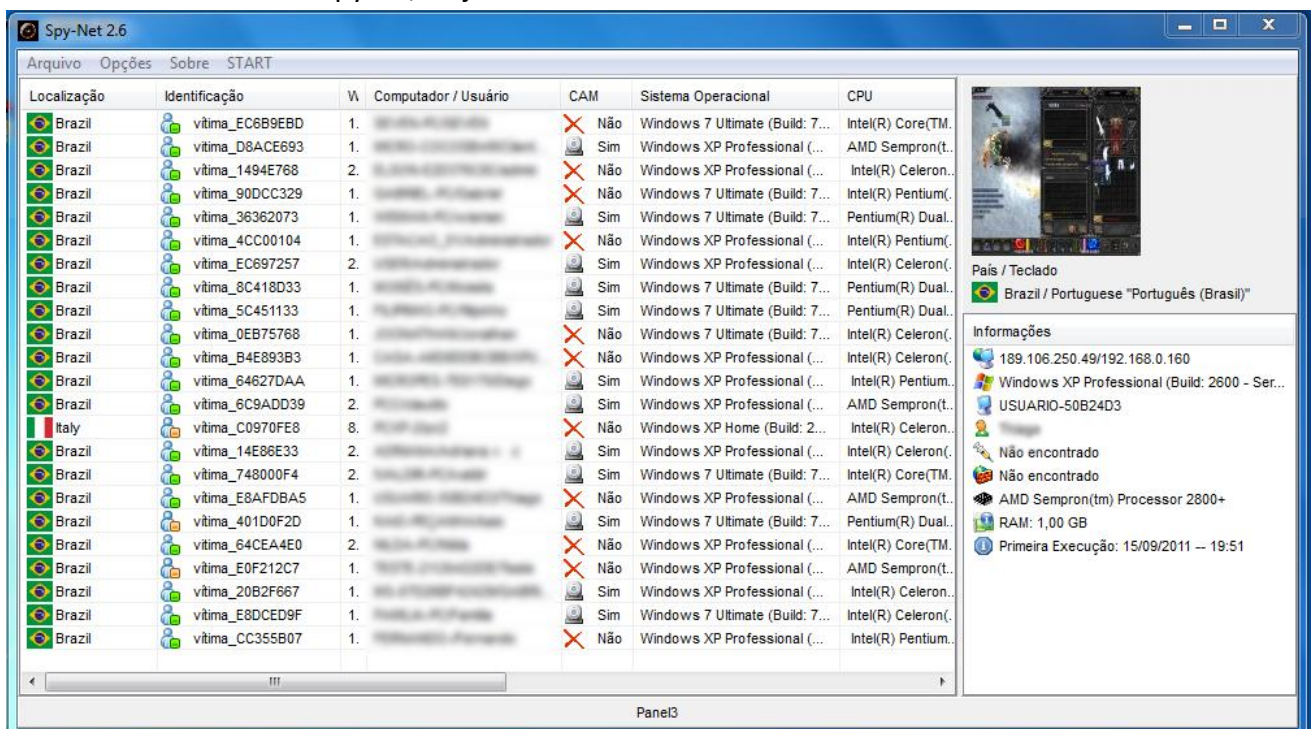


Imagem: Acervo pessoal

**Conexão Direta:** São menos comuns, justamente por terem perdido lugar para os de conexão reversa. Nesses, ao contrário dos de conexão reversa, é o client que se conecta com os servers. Então, o servidor instalado no computador das vítimas fica aguardando a conexão do client, que é feita manualmente pelo cracker. Quando a conexão é feita, a transferência de informações acontece como no de conexão remota, porém, com apenas uma vítima de cada vez. Essa é a desvantagem do trojan de conexão direta, ele consegue se conectar apenas a uma vítima de cada vez, então, não é possível enviar comandos em massa para todas as vítimas infectadas.

Trojan ProRAT, de conexão direta:

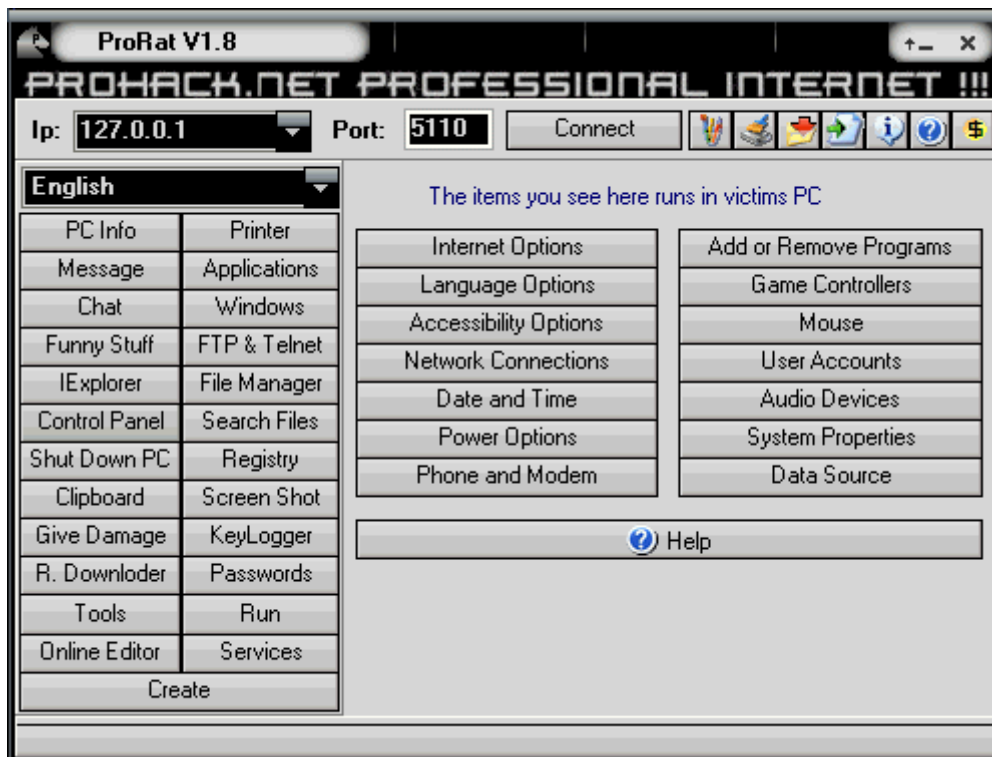


Imagem: MegaSecurity

## Trojan e Redes

Já que estamos falando sobre conexões, vale ressaltar que para os clientes se conectarem aos servidores (*ou o contrário*), é necessário que alguma porta esteja aberta, para que os dados transitem normalmente entre um PC e outro. Essa porta deve ser aberta no roteador do cracker, e liberada no firewall, para que não haja interferência na comunicação.

### ***“Mas Nick, minha internet é via rádio/3g”***

Bom, então você só conseguirá usar trojans quando abrir as portas de sua conexão. Como se faz isso? Geralmente você tem que ligar para a central e pedir suporte para que façam isso para você. Eles vão perguntar o motivo, e então você usa um pouco de engenharia social e inventa uma história. Sei lá, diz que é para criar um servidor de um jogo, ou pra usar o uTorrent, aí é com você.

Se falarem que não vão abrir, diz que vai cancelar o contrato e que vai contratar outra empresa (*de preferência, alguma concorrente direta, rival*), e vai ver que logo logo eles vão dar a solução para seu problema.

Muita gente não consegue usar trojans via rádio por causa disso. Vêem vídeos do youtube que são feitos para internet via cabo e não conseguem abrir as portas por não terem a senha do painel de configuração.

***“Entendi. Mas qual o tipo de trojan que eu vou conseguir usar em internet à rádio?”***

Essa pergunta eu não sei responder ao certo, porque nunca usei internet via rádio, mas há um método em que você consegue usar trojan de conexão direta.

Você precisa do Hamachi instalado no seu computador e no da vítima. Hamachi pra quem não sabe é um programa que cria uma rede privada entre dois ou mais computadores que estão em redes diferentes, como se fosse uma rede LAN virtual.

Ele é usado muito para criar servidores de jogos. Quando duas pessoas estão conectadas no mesmo servidor do Hamachi, ele gera automaticamente um IP virtual para os dois computadores, e cria uma VPN (Virtual Private Network) para eles.

Dessa forma você pode pegar o IP virtual da vítima (*que foi fornecido pelo Hamachi*), e usar o trojan de conexão direta para se conectar com esse IP. Depois disso é só enviar o server e esperar que ela execute. Quando você tiver certeza que ela executou, conecte-se ao IP e pronto.

A desvantagem desse método é que a vítima precisa estar com o hamachi instalado, aberto e no mesmo servidor que você, além de que ela precisa executar o server do trojan, o que requer um pouco de engenharia social para que isso seja feito.

### **Malwares em celulares**

Vale lembrar também que essas pragas não atacam apenas computadores, mas dispositivos móveis também, como celulares e tablets. Se você usa um smartphone Android, certamente já se deparou com aquelas páginas que dizem que seu celular está infectado, ou que está muito lento e precisa de um programa especial para atualizá-lo. Na verdade isso é um tipo de engenharia social, onde o atacante deixa a vítima com medo, passando uma informação falsa sobre o estado de seu aparelho, para fazer com que ela baixe o suposto aplicativo que promete acelerar o celular, mas na verdade só vai fazer com que ele fique mais lento.

E para os que dizem que o sistema da Maçã é melhor, saiba que estão enganados. Recentemente foi criado um protótipo de carregador para iPhone que contém um malware instalado nele, apenas aguardando algum iPhone se conectar a ele. Com apenas um minuto de conexão com o aparelho, ele já consegue infectar o sistema da maçã.

O malware foi capaz de comprometer completamente o sistema, independente das ações do usuário. A equipe que desenvolveu esse carregador modificado já contactou a Apple, e eles estão vendo o que pode ser feito para evitar que crackers usem isso.

Foto do carregador alterado:





Imagem: Canal iPhone

Para android também há muitos, **muitos, MUITOS** malwares. Alguns são disponíveis até gratuitamente, como é o caso do trojan AndroRAT, para android, que tem várias ferramentas como escuta telefônica, visualização das câmeras, gravação do microfone e etc.

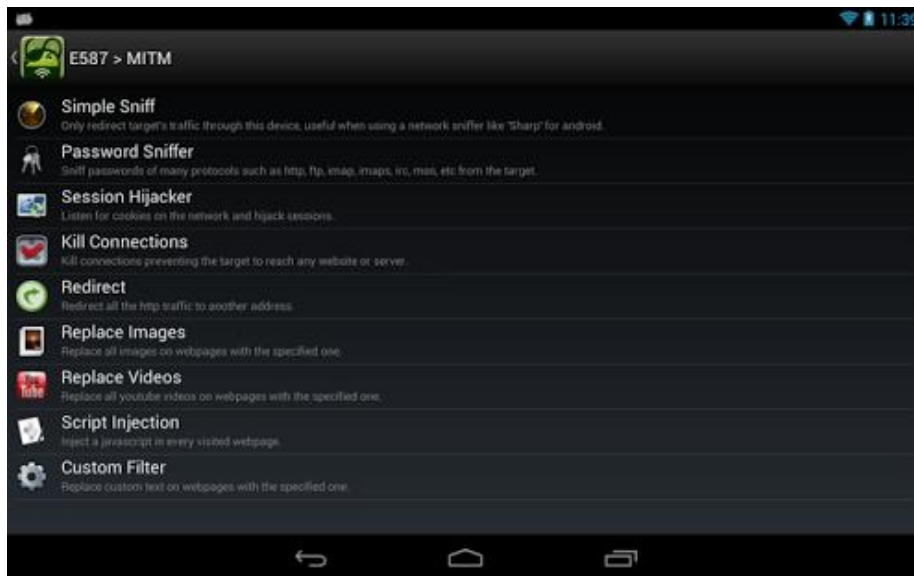
Trojan AndroRAT para invasão de Android, disponível gratuitamente:



Imagem: MackHacker

Também há ferramentas de pentest para Android, como o Dsploit, disponível gratuitamente na Play Store, com várias utilidades, como sniffer de sessões, DNS Poisoning, etc...

Print do Dsploit rodando em um tablet:



*Imagem: Google PlayStore*

No próximo capítulo nós vamos ver como os malwares passam despercebidos pelas ferramentas antivírus. Até lá!

# 9. Técnicas para esconder malwares



Hoje nós estudaremos sobre as técnicas usadas para fazer com que malwares consigam se instalar no PC das vítimas, sem que sejam detectados por programas antivírus.

Existem várias formas de deixar o malware indetectável, mas sem dúvida, a mais conhecida e mais usada é a de criptografar o código dele.

A criptografia basicamente serve para fazer com que um determinado código fique ilegível ao computador. E é isso que o crypter faz.

**9.1 Crypter** – Esse software carrega o malware, joga o código dele na memória do sistema, embaralha o código, criptografa e então salva o arquivo, em um novo executável. Quando esse novo executável é aberto, o programa automaticamente organiza o código e remove a criptografia, fazendo o processo inverso. Só aí que o malware de fato é executado.

Isso acontece porque os antivírus têm uma base de dados com várias strings suspeitas. Quando um programa com alguma dessas strings é executado, o antivírus reconhece aquilo como uma ameaça e não deixa ele ser executado.

É como se fosse uma revista policial na porta de uma boate, onde o policial revista todas as pessoas que tentam entrar, e se alguém tiver algum item ameaçador, essa pessoa é barrada pela polícia, que não deixa ela entrar na boate.

Screenshot de um crypter que usa a chave AES ou RC4 como criptografia:

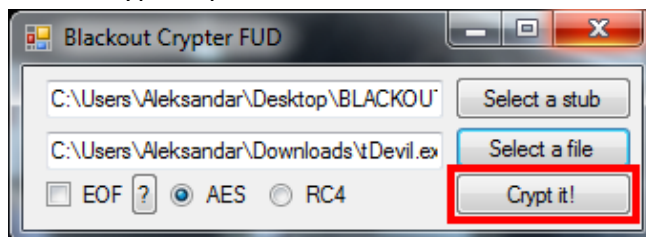


Imagem: HackPCOnline

O crypter é dividido em duas partes: O client e a Stub.

O client é a parte gráfica, o programa em si, onde você carrega o malware para ser criptografado. Mas um arquivo criptografado nunca poderia ser executado, pois seu código é ilegível. É aí que entra a stub.

A stub se responsabiliza por fazer o processo inverso do crypter, ou seja, ela gera um código que é anexado ao malware, e faz com que esse código seja executado antes do malware. Esse código é que faz toda a descriptografia do código, tornando o malware um executável novamente, e dessa vez, sem passar pelos antivírus.

**9.2 Binders** – Os binders (ou joiners) são softwares usados para juntar dois ou mais arquivos. Eles são usados geralmente em conjunto com os crypters, e alguns crypters têm até essa função implementada nele mesmo.

Quando os arquivos são submetidos ao binder, ele compacta os códigos dos executáveis e os coloca em pilha, logo abaixo do código de descompressão. Os binders têm uma espécie de stub acoplada neles mesmos, que se encarrega de fazer esse serviço de descompressão. Quando o executável “bindeado” é executado, a stub roda o código de

descompressão, que joga os códigos dos executáveis na memória como se tivessem sido executados separadamente.

Imagem do Cactus Joiner, o binder mais conhecido:

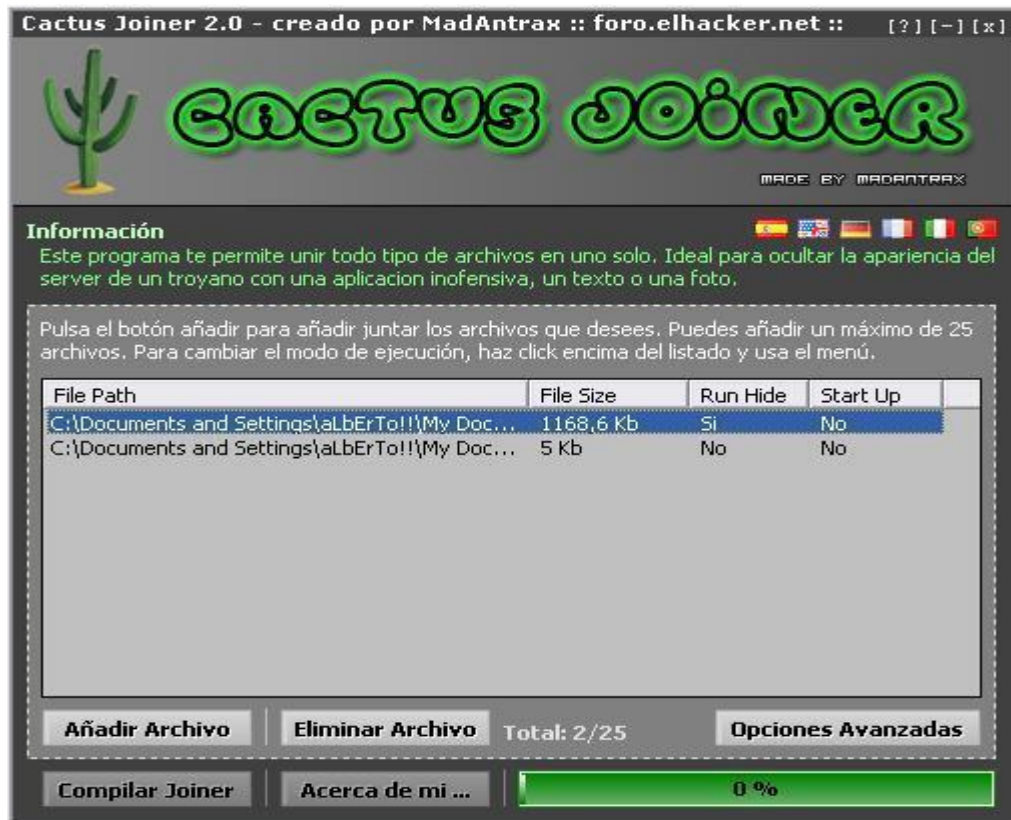


Imagem: PerlBal

Isso é útil quando você quer infectar uma vítima mas não tem uma boa engenharia social para fazer com que ela execute um programa que não funciona (*pra ela*), então você junta o server do trojan com alguma foto, vídeo ou com outro programa, e manda para ela. Quando executado, os dois serão executados, mas o server será executado em modo *stealth*, ou seja, em silêncio, sem a vítima saber.

**9.3 Packers** – Os packers são um pouco diferentes dos crypters. Eles funcionam como o Winrar, servem para compactar o código e deixá-lo bem menor que o original.

A diferença dos packers para o winrar é que o winrar, quando extraído, retorna o executável original, com seu tamanho original, enquanto o packer compacta o executável por inteiro, ocorrendo a descompactação apenas em run-time.

**“Como assim, Nick?”**

Supondo que você use um packer em um servidor de trojan que tem 455kb. Você comprime esse server com um packer, e na compressão esse tamanho vai para 122kb. Quando a vítima executar esse servidor comprimido, ele vai se autodescompactar, sem criar novos arquivos, tudo feito na memória.

Upx rodando no Windows XP:

```
C:\WINDOWS\System32\cmd.exe
Ultimate Packer for eXecutables
Copyright (C) 1996, 1997, 1998, 1999, 2000, 2001, 2002
UPX 1.24w Markus F.X.J. Oberhumer & Laszlo Molnar Nov 7th 20
Usage: upx [-123456789dlthUL] [-qvfkl] [-o file] file..

Commands:
  -1 compress faster          -9 compress better
  -d decompress              -l list compressed file
  -t test compressed file   -U display version number
  -h give more help         -L display software license

Options:
  -q be quiet                -v be verbose
  -oFILE write output to 'FILE'
  -f force compression of suspicious files
  -k keep backup files
  file.. executables to (de)compress

This version supports: dos/exe, dos/con, dos/sys, djgpp2/coff, watcom/le,
win32/pe, rtm32/pe, tnt/adam, atari/tos, linux/386

UPX comes with ABSOLUTELY NO WARRANTY; for details type `upx -L`.

C:\EJECUTABLES\upx>_
```

Imagem: Softonic

O arquivo compactado contém o código original e um código de descompressão. O código de descompressão vai na primeira sessão do executável, para ser o primeiro trecho de código a ser executado.

Quando o executável é executado, o código de descompressão joga o código original na memória e descomprime, fazendo como se ele tivesse sido executado normalmente. O funcionamento dele é parecido com o crypter, e com o binder o fluxograma é o mesmo, veja:

**Crypter: Entrada → Criptografia → Saída → Execução → Descriptografia**

**Binder: Entrada → Anexamento → Saída → Execução → Desacoplamento**

**Packer: Entrada → Compressão → Saída → Execução → Descompressão**

O problema dos packers é que eles geram sempre o mesmo código, então, os mais conhecidos já estão detectáveis aos antivírus. Por exemplo, o packer UPX, que é o mais conhecido, adiciona em dois lugares do executável a string “UPX0” e “UPX1”, para dizer que ali está o código de descompressão. Dessa forma, os antivírus conseguem detectar que o executável está compactado com o UPX, e ele já o reconhece como uma ameaça, mesmo que não seja.

Outra desvantagem do packer é que seu código de descompressão fica na primeira sessão do executável, então é mais fácil para um hacker encontrar o código e fazer a operação reversa. O nome disso é análise de malware, e isso é feito por meio da engenharia reversa, que são os nossos próximos capítulos. Vamos lá.

# 10. Análise de Malware



Com o surgimento exponencial de malwares, é necessário que pessoas se especializem em acabar – ou ajudar a acabar – com eles. Pra isso que serve a Análise de Malware.

Eu gosto muito dessa área do hacking, porque ela meio que faz uma dissecação no malware, e você consegue ver tudo que tem dentro dele, e o que ele faz. Eu tenho a mania de pegar componentes eletrônicos quebrados e abrir para ver como funciona, isso é desde criança, eu fazia isso com meus brinquedos.

Você já assistiu CSI? Aquele seriado americano que mostra a investigação da cena do crime? A análise de malware é basicamente isso.

O analista consegue o(s) arquivo(s) do(s) malware(s), e usando técnicas, consegue identificar como foi feito, por quem foi feito, quando foi feito, sua utilidade e o que ele fez no computador da vítima.

O ato de coletar dados em um sistema “vivo” (ou seja, que não foi totalmente comprometido pelo malware) causa algumas mudanças que o analista terá que saber.

Por exemplo, executar ferramentas como o Helix (*distribuição linux feita para análise forense*) a partir de uma mídia removível irá alterar dados voláteis quando for carregado na memória principal, e normalmente irá criar ou modificar arquivos e entradas no registro do sistema a ser analisado. Da mesma forma, usar ferramentas forenses que utilizam da conexão remota necessariamente – e obviamente – estabelece uma conexão de rede, executa instruções na memória, e causa outras alterações no sistema.

Por isso existem ferramentas que batem um “snapshot” do computador infectado, salvando informações da memória ram, conexões de rede ativas, processos executando, serviços rodando, tempo de execução, arquivos temporários criados e etc. Tudo isso é salvo em um arquivo. Esse arquivo é emulado em um computador virtual.

O que vem a ser um computador virtual? Computador virtual, ou máquina virtual (MV) é um software onde você emula o live disc de um determinado sistema.

Ou seja, você pode executar o linux dentro do seu windows 7, sem precisar formatar! Se quiser conferir, procure por **VMWare**, ou **VirtualBox** no Google. Esses dois softwares são máquinas virtuais, e funcionam perfeitamente para fazer análise de malware. Eu particularmente prefiro o VMWare, já que ele consome bem menos CPU e memória, além da performance do sistema a ser emulado ser muito boa.

A análise de malware consiste basicamente em duas etapas:

**10.1 Análise estática:** Ocorre antes da execução do malware, quando o analista observa as strings do arquivo a procura de informações sobre o criador do malware, estuda sobre como ele atua no sistema operacional, que tipo de técnicas de ofuscação são utilizadas, quais fluxos de execução levam ao comportamento principal planejado, se há operações de rede, download de outros arquivos, captura de informações do usuário ou do



sistema, utiliza ferramentas como disassemblers, decompilers e debuggers para entender como funciona o código(essas ferramentas serão explicadas mais tarde), procura por informações de compilação, como a data em que foi criado, a linguagem utilizada, e se há algum tipo de compactação/criptografia...

Abaixo um print do software ExeInfoPE, que serve para mostrar o compilador usado:



Imagem: Crimes Cibernéticos

**10.2 Análise dinâmica:** Ocorre na hora execução do malware, feito em uma máquina virtual, onde o analista consegue informações como arquivos criados e interagidos pelo/com o malware, monitoramento de bibliotecas e *systemcalls*, chaves no registro criadas e modificadas, conexão com a internet (*callhome*), e informações sobre as funções reais do malware.

Veja um print do programa Process Monitor mostrando dois arquivos criados por um trojan:

Operation	Path	Result	Detail
QueryOpen	C:\WINDOWS\system32\kernel32.dll	SUCCESS	CreationTime: 13/4/2008 16:20:30, La
CreateFile	C:\WINDOWS\inf\asynceql.inf	SUCCESS	Desired Access: Generic Read/Write,
CreateFile	C:\WINDOWS\inf	SUCCESS	Desired Access: Synchronize, Dispositi
CloseFile	C:\WINDOWS\inf	SUCCESS	
WriteFile	C:\WINDOWS\inf\asynceql.inf	SUCCESS	Offset: 0, Length: 16.384
CloseFile	C:\WINDOWS\inf\asynceql.inf	SUCCESS	

Imagem: Crimes Cibernéticos

Mas a análise não se limita apenas à observação dos componentes e do comportamento do malware na máquina. Alguns malwares são propagados via email, com links contendo engenharia social, e por esses emails é possível chegar ao cracker. Geralmente os assuntos dos emails são algo como “Boletim de ocorrência”, ou “pagamento do boleto”, etc...

Veja abaixo a screenshot que acabei de tirar com alguns spams que recebi:

<input type="checkbox"/> Nickguitar .dll	▶ RES: URGENTE - PROTOCOLO 03644549 26/11/2014 08:32:07	26/11/2014
<input type="checkbox"/> Nickguitar .dll	▶ Fwd: Lembrete : Boleto Vencido. 24/11/2014 13:33:25 0,1677904	26/11/2014
<input type="checkbox"/> Nickguitar .dll	▶ RES: ALERTA 54713562 24/11/2014 07:38:48	24/11/2014
<input type="checkbox"/> Nickguitar .dll	▶ RES: INFOMAIL 07278050 19/11/2014 02:11:04	19/11/2014
<input type="checkbox"/> Nickguitar .dll	▶ RES: INFOMAIL 93206438 18/11/2014 05:23:57	18/11/2014
<input type="checkbox"/> Nickguitar .dll	▶ 18/11/2014 06:00:06	18/11/2014
<input type="checkbox"/> Nickguitar .dll	▶ RES: CONTATO 09836050 14/11/2014 06:59:52	14/11/2014
<input type="checkbox"/> Nickguitar .dll	▶ RES: ATENDIMENTO 29536986 13/11/2014 10:53:08	14/11/2014
<input type="checkbox"/> Nickguitar .dll	▶ RES: URGENTE 07/11/2014 06:06(52014Fri, 07 Nov 2014 06:06:55 +0100Fri, 07 Nov 2014 06:06:55...	07/11/2014
<input type="checkbox"/> Nickguitar .dll	▶ RES: ITAU URGENTE 07/11/2014 01:35(52014Fri, 07 Nov 2014 01:35:06 +0100Fri, 07 Nov 2014 01:...	07/11/2014
<input type="checkbox"/> Nickguitar .dll	▶ RES: URGENTE - PROTOCOLO 75664675 29/10/2014 03:59:52	29/10/2014
<input type="checkbox"/> Nickguitar .dll	▶ ITAU BANKLINE 21105735 13/10/2014 11:08:53	14/10/2014
<input type="checkbox"/> Nickguitar .dll	▶ 27/08/2014 07:25:34	27/08/2014
<input type="checkbox"/> Banco BMG	▶ Segurança BMG	13/08/2014
<input type="checkbox"/> Cielo	▶ Parabéns, você foi contemplado!	10/08/2014

*Imagem: Acervo pessoal.*

Vários desses emails continham malwares para serem baixados, sendo a maioria keylogger banker. O keylogger banker é um variante do keylogger comum, e tem como principal característica a obtenção seletiva de dados. Falamos dele no último capítulo.

Esse tipo de KL é interessante para ser analisada, porque ela tem o que nós chamamos de *callhome*, quando o malware retorna os dados para a casa do cracker (*Na verdade todos os Keyloggers tem, mas analisar esse tipo é mais interessante porque estamos lidando com um estelionatário*). Assim nós podemos obter o endereço IP dele, ou o site isca que ele pode ter usado.

#### **“Site isca? O que é isso, Nick?”**

Alguns crackers criam scripts que fazem parte do KL, ou que armazenam os dados ele, e hospedam em sites – provavelmente hackeados – para que o KL funcione normalmente. Já teve um caso em que o Keylogger banker enviava todos os dados que ele capturava para uma página em txt, em uma pasta com o nome de *“shellfile”*. Deduzi que naquela pasta estavam os arquivos com os logs do KL e a shell que o cracker hospedou.

Então eu fui colocando combinações de nomes de arquivos naquela pasta, e acabou que consegui encontrar a shell. O cracker não tinha só aquele KL que eu recebi por email, mas tinha um esquema com vários. E lá estava o arquivo .txt, com todas as senhas. Apaguei tudo, deletei a shell e avisei para o admin que um bandido estava usando o site dele para roubar pessoas.

Veja abaixo as strings de um KL Banker, e note os sites em que ele atua:



Address	Length	String
"..." .text:00401A20	00000009	Project1
"..." .text:00401B0C	0000001C	www.bb.com.br
"..." .text:00401B2C	0000001E	200.98.201.19
"..." .text:00401B50	00000014	bb.com.br
"..." .text:00401B68	00000032	www.bancodobrasil.com.br
"..." .text:00401BA0	00000028	www.bradesco.com.br
"..." .text:00401BCC	00000020	bradesco.com.br
"..." .text:00401BF0	00000020	www.itau.com.br
"..." .text:00401C14	00000018	itau.com.br
"..." .text:00401C30	0000002A	www.santander.com.br
"..." .text:00401C60	00000022	santander.com.br
"..." .text:00401C88	00000030	www.santandernet.com.br
"..." .text:00401CBC	00000020	www.real.com.br
"..." .text:00401CE0	00000018	real.com.br
"..." .text:00401CFC	00000038	www.itaupersonnalite.com.br

*Imagem: Crimes Cibernéticos*

# 11. Engenharia Reversa



Nesse capítulo falaremos um pouco sobre engenharia reversa, bem como suas utilidades, funções, etc...

Primeiro, o que é engenharia reversa?

Engenharia você sabe que é o estudo que constrói as coisas. Engenharia civil, naval, etc.. Tudo isso visa a construção de coisas para melhorar as condições da utilização delas.

A engenharia reversa surgiu com a ganância em poder do ser humano, e foi usada para desconstruir equipamentos bélicos e entender como eles funcionam, para que o exército pudesse construir equipamentos melhores e mais sofisticados, e assim ter uma certa superioridade militar. Atualmente a engenharia reversa é muito usada pelas grandes empresas para saber como seus concorrentes estão usando a tecnologia para conseguir dinheiro.

Por exemplo, a fábrica da Samsung compra vários iPhones 6 e os desmonta para ver os componentes que tem dentro dele, estudar os chips e etc... Tudo isso para aumentar a possibilidade de fazerem algo melhor na próxima versão.

O nome “reversa” já diz tudo. A engenharia reversa é o estudo de um objeto, seja um processador, celular, relógio, ou algum software. Esse estudo é realizado quando o objeto é desmontado para suas peças, componentes e comandos possam ser analisados.

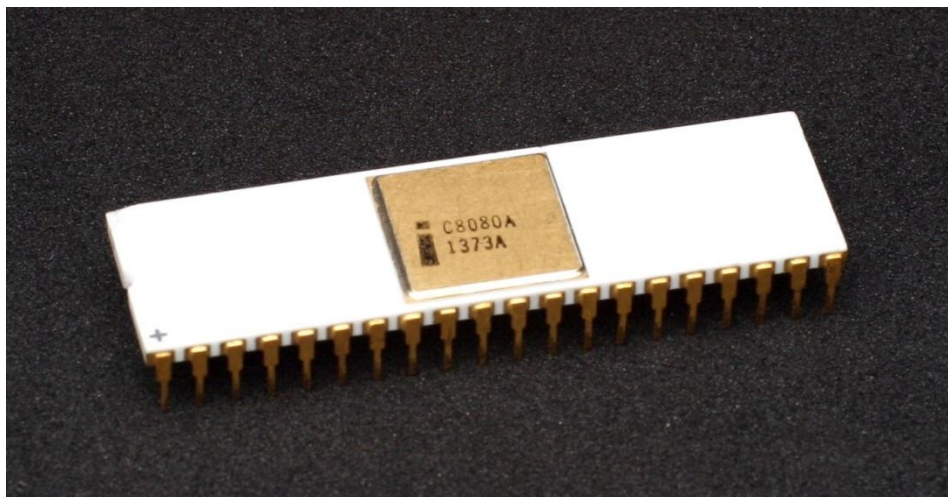
***“Qual a utilidade disso, Nick?”***

Isso é feito para descobrir como ele foi fabricado, como ele poderia ser melhorado e que outras funções ele poderia realizar.

Foi através da engenharia reversa, por exemplo, que a fabricante de processadores **AMD** conseguiu fazer uma cópia exata do processador Intel 8080, no ano de 1975.

A empresa provavelmente comprou alguns processadores da Intel, abriu-os e os estudou detalhadamente, conseguindo criar uma cópia perfeita através de tentativa e erro.

Abaixo vemos uma foto do processador Intel 8080:



*Imagem: Wikipédia*

A engenharia reversa está mais próxima de você do que você imagina. A pirataria de CDs e DVDs utiliza a engenharia reversa para descobrir como funciona a proteção anticópia dos discos, e assim removê-la.

### **Engenharia reversa no software**

A engenharia reversa de software é a mesma coisa que na vida real. Usa-se de programas para conseguir o código primitivo de compilação do programa, pausar a execução do mesmo para analisar seu comportamento no sistema, etc...

Uma cena bem comum de acontecer um software ter sido usado por muitos anos. Durante esses anos, ele foi corrigido, adaptado e aprimorado diversas vezes. Porém, a cada vez que o programa era aprimorado e adaptado, outros problemas apareciam, causando efeitos na hora da execução.

Agora o software está instável, porém, continua sendo usado, e precisa ser reformulado e reescrito para que funcione normalmente, junto com as tecnologias atuais.

Utilizando-se da engenharia reversa, o hacker consegue entender como o software foi programado, com qual motivo o programador fez determinado código, entre várias outras coisas... E assim ele pode reconstruir o programa usando o antigo como base.

### ***“Mas, como isso tudo é feito em um computador?”***

Usando ferramentas que foram criadas especialmente para isso. Existem diversos tipos de ferramentas para a análise de software, cada uma com uma função específica. Vou citar algumas das mais utilizadas e explicar o que elas fazem.

**Disassembler:** É um programa que converte o código de máquina (binário) para uma linguagem de programação de baixo nível, como assembly, por exemplo. Quando o programa está executando, o processador interpreta os códigos em binário (*apenas 0 e 1*), e o disassembler se responsabiliza por transformar o binário para uma linguagem de entendimento humano.

**Debugger:** É um software usado para a depuração de erros no programa. Ou seja, tem um programa que está dando erro e você quer saber qual o problema. Você usa um debugger, ele irá travar o código do programa na linha em que o erro aconteceu, mostrando a instrução em Assembly que contém o erro. São usados por analistas de software.

**Decompiler:** Aprendemos que o compilador transforma a linguagem de programação em código de máquina. O decompiler faz o contrário. Linguagens que usam máquinas virtuais (como .NET e Java) são mais fáceis de descompilar, porque as máquinas virtuais contém informações que facilitam o processo de descompilação.

## 11.1 Software Cracking

Bem provável que você esteja lendo isso em um sistema operacional pirateado, como o Windows. Isso só é possível porque uma equipe de crackers se dedicou a entender como funciona o sistema de verificação de serial do Windows, para conseguir quebrá-lo e fazer com que ele possa ser distribuído gratuitamente na internet.

Como os crackers usam sempre o conhecimento para realizar ações ilícitas, eles também usam conhecimento de engenharia reversa para desenvolver cracks/patches/keygens para programas pagos, causando muito prejuízo para os autores.

Geralmente, quando você baixa ou compra um software com um período de testes (Trial) ou pago, é possível inserir um número serial para validá-lo, e assim poder usar até expirar a licença. A maioria deles, esconde os seriais válidos dentro do próprio código-fonte, facilitando o trabalho dos crackers, enquanto outros fazem a validação pela Internet, tornando tudo mais difícil. Mas como vocês sabem, **nada é tão seguro que não possa ser quebrado**. Na teoria, crackear um software pago é simples. O cracker pode fazer de três formas:

**1** – Descompilando o software e obtendo a chave de ativação que está salva nele mesmo. Nesse caso, o software pede para que você insira a key de ativação, e ele entra em uma estrutura condicional com a key que você entrou, e verifica se a key entrada bate com a key original. Se bater, ele deixa você usar, se não, ele exibe uma mensagem de erro. Dessa forma o cracker pode obter a key original com a qual o software faz a verificação, distribuindo publicamente o serial de ativação.

**2** – Criando uma rotina que aceite qualquer tipo de serial entrado. Por exemplo, o software faz a seguinte validação: *“Se a chave entrada for igual a essa: ‘123’; Então, deixe-o usar o programa”*. Então o cracker modifica essa condição, para que o programa aceite qualquer key entrada. Ficaria assim: *“Se alguma chave for entrada; Então, deixe-o usar o programa”*. Logo, qualquer chave entrada – válida ou não – seria aceita, e o programa funcionaria normalmente.

**3** – Removendo totalmente qualquer tipo de verificação que o programa venha a fazer, e ativando-o automaticamente. Antes do bloco de código em que o programa faz a verificação sobre a chave utilizada o cracker coloca um desvio, que envia o fluxo para depois do código, e então a ativação é feita. Ou seja, o programa executaria, mas, quando chegasse a hora de exibir a pop-up pedindo a chave de ativação, ele pularia para a etapa depois dessa verificação, e o valor que define se o programa está ativo ou não seria alterado, para fazer como se ele estivesse ativo. O nome dessa técnica é patching.

Lembrando que essas técnicas só funcionam com programas que fazem ativação por meio de comparação de strings, comparando a key que você submeteu com a key original, que está contida no programa. Programas que fazem ativação online são mais complicados de serem quebrados.

# 12. Criptografia



Vamos falar um pouco sobre Criptografia. Primeiro vamos desmistificar os significados por trás desse nome lindo.

**Criptografia:**Essa palavra vem do grego *kryptós*, "escondido", e *gráphein*, "escrita", e significa literalmente escrever de uma forma oculta. As primeiras formas de criptografia datam de 1900 A.C, no Egito, onde um escrivão utilizou de hieróglifos fora do padrão para escrever uma carta. Essa carta estava criptografada, porque a mensagem era ilegível para quem não tinha conhecimento da criptografia utilizada. A criptografia também foi muito utilizada na primeira e segunda guerra mundial, para enviar mensagens sobre estratégias e outras informações confidenciais.

Atualmente, a criptografia está integrada em quase todos os sistemas virtuais que conhecemos, tendo seu uso nas mais variadas áreas da informática. Por exemplo: Você quer realizar uma compra em um site, e precisa colocar suas informações bancárias nele, mas você corre o risco de estar sendo monitorado por algum cracker, só esperando você digitar sua senha para que ele possa roubar. É aí que entra a criptografia. Quando você entra com seus dados bancários, o servidor se responsabiliza por criptografar os dados de uma forma que apenas ele consiga fazer o processo reverso.

Abaixo vemos a string "123" criptografada com o MD5:

**202cb962ac59075b964b07152d234b70**

A criptografia pode ser utilizada para proteger arquivos também. Um programa bem conhecido é o TrueCrypt, que criptografa todos os arquivos de uma determinada partição do HD, e eles só podem ser utilizados se forem descriptografados. Como nós falamos no capítulo de malwares, os RansomWares usam dessa tecnologia para obter proveitos sobre a vítima.

A unidade de nível de segurança da criptografia é o bit, e ele é dado pelo tamanho das chaves. Por exemplo, uma chave criptográfica de 8 bits pode gerar até 256 combinações diferentes ( $2^8$ ). É um número bem pequeno de combinações, já que existem programas que conseguem descobrir as senhas pelo método de tentativa e erro (Brute Force). Já uma chave de 128 bits tem  $3,4 \times 10^{38}$  combinações diferentes ( $2^{128}$ ). Um número tão grande que levariam décadas, talvez séculos para um computador quebrá-la pelo método de brute force.

O **bruteforce** é uma técnica em que é criptografado várias strings, e o hash originado dessas criptografias é comparado com o original, até que a comparação bata, e a criptografia seja quebrada. Não entendeu? Veja:

Temos a palavra "alicate", e queremos criptografá-la. Ela então criptografada fica assim: **Bk3j#1qalj5Gi96vSS\$**. Essa é a nossa hash. Vamos supor que nós não sabemos que essa hash significa "alicate" e queremos descobrir. Nós submetemos essa hash para um programa que vai gerar vários caracteres diferentes e vai criptografando o resultado.

Haverá um momento em que o programa – por uma questão matemática – vai gerar a palavra "alicate", ela será criptografada. O hash gerado será o mesmo hash que foi submetido, e então a criptografia será quebrada. É assim que os sites conseguem "quebrar" a criptografia que você submete à eles.

Uma chave criptográfica é um valor secreto que modifica um algoritmo de encriptação. A fechadura da porta da frente da sua casa tem uma série de pinos. Cada um desses pinos possui múltiplas posições possíveis. Quando alguém põe a chave na fechadura, cada um dos pinos é movido para uma posição específica. Se as posições ditadas pela chave são as que a fechadura precisa para ser aberta, ela abre, caso contrário, não.

As criptografias atuais podem ser de dois tipos: De **chave simétrica** e **assimétrica**.

Nas criptografias de **chave simétrica**, a mesma chave é utilizada tanto pelo emissor quanto por quem recebe a informação. Ou seja, a mesma chave é utilizada para codificação e para a decodificação dos dados. É usada nos casos em que a informação é codificada e decodificada por uma mesma pessoa, sem haver o compartilhamento da chave secreta.

Porém, quando estas operações envolvem pessoas ou computadores diferentes, é necessário que a chave secreta seja previamente combinada por meio de um canal de comunicação seguro (*para não comprometer a confidencialidade da chave*). Não é recomendado seu uso para guardar e enviar informações muito importantes. Exemplos de métodos criptográficos que usam chave simétrica são: **AES, Blowfish, RC4, 3DES e IDEA**.

Nas criptografias de **chaveassimétrica**, são utilizadas duas chaves distintas: uma pública, que pode ser livremente divulgada, e uma privada, que deve ser mantida em segredo por seu dono. Quando uma informação é codificada com uma das chaves, somente a outra chave do par pode decodificá-la. Qual chave usar para codificar depende da proteção que se deseja.

A chave privada pode ser armazenada de diferentes maneiras, como um arquivo no computador, um smartcard ou um token. Exemplos de métodos criptográficos que usam chaves assimétricas são: **RSA, DSA, ECC e Diffie-Hellman**.

Para aproveitar as vantagens de cada um destes métodos, o ideal é o uso combinado de ambos, onde a criptografia de chave simétrica é usada para a codificação da informação e a criptografia de chaves assimétricas é utilizada para o compartilhamento da chave secreta (neste caso, também chamada de chave de sessão). Este uso combinado é o que é utilizado pelos navegadores Web e programas leitores de e-mails. Exemplos de uso deste método combinado são: **SSL, PGP e S/MIME**.

O **SSL**(*Security Socket Layer*) é uma camada de segurança que permite que o cliente e servidor possam se comunicar, visando a segurança na informação e integridade dos dados.

Uma conexão utilizando SSL é sempre iniciada pelo cliente. Quando um usuário solicita a conexão com um site seguro, o navegador web (Firefox, Internet Explorer, Opera, Chrome, etc.) solicita o envio do Certificado Digital e verifica se:

- a) *O certificado enviado é confiável.*
- b) *O certificado é válido.*
- c) *O certificado está relacionado com o site que o enviou.*

Uma vez que as informações acima tenham sido confirmadas, a chave pública é enviada e as mensagens podem ser trocadas.

Na imagem abaixo é possível entender como o SSL funciona:

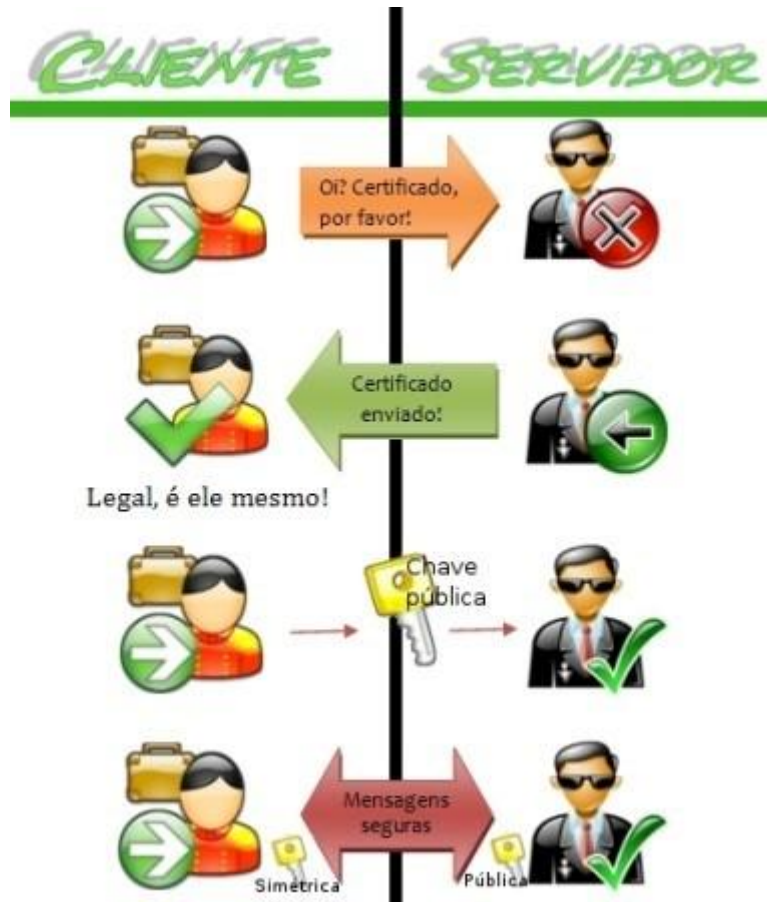


Imagem: Tecmundo

O SSL é responsável pelo "S" depois do "HTTP" em alguns sites. Por exemplo, na página de login do Facebook o link é <https://facebook.com>.

Esse "S" significa *Secured* (seguro), e quer dizer que os dados enviados pelo formulário dessa página estarão seguros, pois serão criptografados pelo SSL antes de chegar no servidor.

Falar de criptografia a fundo é bem complicado, porque exige muito conhecimento matemático sobre o assunto, e não é bom falar sobre esse tipo de coisa em um livro para iniciantes, isso pode confundir a cabeça de vocês.

Nos vemos no próximo capítulo, onde falaremos sobre segurança e anonimidade na web, tal como as ferramentas, técnicas e cuidados que devem ser tomados.

# 13. Segurança e anonimidade na web

Quando falamos de segurança em hacking, deveremos também falar de anonimidade. Mas vamos falar sobre cada item desse separadamente.

Antes de invadir um site, ou atacar qualquer outra coisa, é necessário ter certeza que seu IP não fique nos logs, e é sempre bom utilizar um IP Spoofado para isso. Já expliquei no capítulo de DoS o que é spoof. Trata-se de uma técnica onde o atacante modifica o IP da requisição, fazendo com que seus passos fiquem marcados com um IP falso.

Isso pode ser feito usando uma VPN ou proxy. As duas ferramentas tem finalidades parecidas: Alterar o IP do cliente. A diferença é a aplicação delas. **Lembrando que falarei apenas sobre Proxys WEB nesse capítulo.**

Primeiramente vamos entender o que é Proxy e VPN.

**Proxy** é um servidor que trabalha como um intermediário nas conexões. Ou seja, ao invés de você se conectar diretamente ao servidor do Facebook, você se conecta ao servidor do proxy que envia os dados para o Facebook, e vice-versa. O IP que ficará registrado nos logs quando você acessar uma página é o IP da proxy, e esse IP pode ser de origem estrangeira, ou seja, você pode acessar um site como se fosse um usuário americano.

Ao contrário de servidores VPN, servidores proxy não dedicam seus recursos para criptografar todo o tráfego que passa por eles, e, portanto, pode aceitar conexões simultâneas de um grande número de usuários.

Veja o diagrama abaixo para entender o funcionamento das proxys.

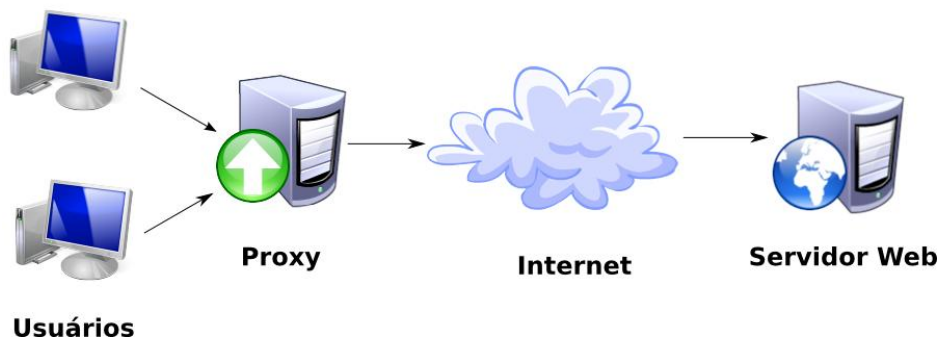


Imagem: Viva o Linux

**VPN** (Virtual Private Network) é – como o próprio nome diz – uma rede virtual privada. Com ela você pode acessar redes locais remotamente (*um tanto quanto irônico, não?*), por exemplo, pode acessar a rede interna do seu trabalho direto da sua casa.

A utilização da internet como forma de conexão entre dois computadores é uma ótima solução em termos de custo, já que é barato obter acesso à internet, mas não é uma boa ideia em termos de privacidade, pois a internet é uma rede pública, onde os dados que transitam nela podem ser lidos por qualquer pessoa.

Essa questão pode ser resolvida simplesmente com o uso da criptografia, que estudamos no último capítulo.

É possível entender o funcionamento de uma VPN com o diagrama abaixo.



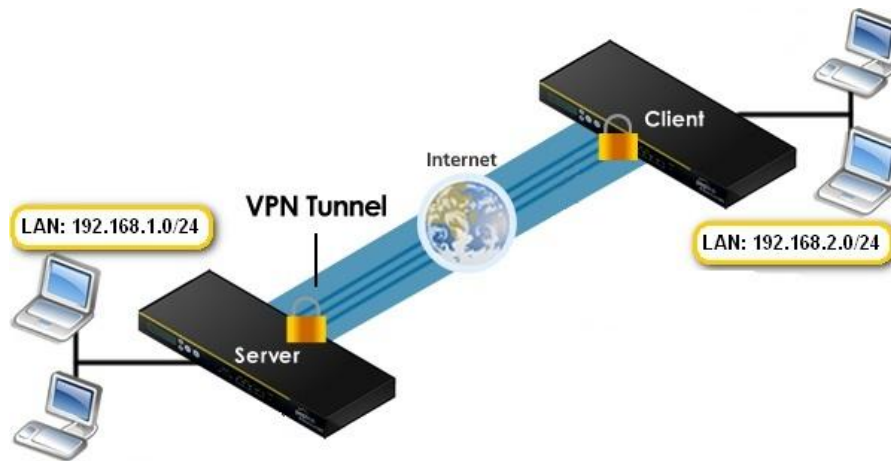


Imagem: VPN Revista

O proxy web é utilizado para alterar o seu IP de acesso em algum site, mas essa modificação só é feita no navegador. O IP de outras conexões continua normal. Isso é útil porque você pode ativar e desativar a proxy de uma hora para outra, sem precisar ficar esperando a internet reconectar e etc, diferente do VPN, que altera o IP para toda a rede. Isso pode fazer com que outros programas que usam internet também fiquem lentos, como Torrent ou Skype, por exemplo.

Ou seja, usando um proxy, o IP que você se conecta aos servidores do Skype é diferente do IP que você usa para acessar algum site. Já quando você usa VPN, o IP muda para todos os programas que se conectam à internet.

Há diversas maneiras de invadir uma máquina rodando algum programa de proxy. O método para cada tipo de proxy depende de sua versão e do sistema operacional no qual roda. Mas há algumas vulnerabilidades que funcionam em qualquer plataforma, pois são inerentes à tecnologia e não à implementação da tecnologia por um determinado desenvolvedor.

### **Anti-Virus**

Impossível falar de segurança virtual e não falar sobre antivírus. Todos usam, dos mais diversos tipos. Mas, você já parou para pensar como funciona o antivírus?

Como já foi falado antes, os anti-virus atuais contam com um gigantesco banco de dados com vários tipos de malwares que são detectados e cadastrados lá a cada atualização. Nesse banco de dados tem várias strings e registros suspeitos que são feitos pelos malwares. Por exemplo, os trojans geralmente criam uma entrada no registro em *HKLM/Software/Microsoft/Windows/CurrentVersion/*, ou seja, qualquer arquivo que tenha o código para modificar essa chave já é detectado como sendo um trojan (mesmo que não seja), com exceção dos falsos-positivos.

Os falsos-positivos são programas que contém alguma parte do código considerada suspeita, e os antivírus mandam-no para quarentena para ser analisado. É constatado então que se trata de um arquivo não malicioso, e então uma nova lista com os malwares é atualizada, dessa vez, sem esse falso-positivo na lista.

Apesar do método de reconhecimento citado acima ser o mais utilizado e efetivo na maioria dos casos, existem outras formas de um antivírus encontrar acabar com essas

ameaças no sistema. O método é chamado de **análise heurística** monitora constantemente as atividades do computador e entra em ação quando algum programa tenta modificar configurações do sistema ou arquivos importantes.

Pelo fato de vírus inéditos surgirem todos os dias, a biblioteca pode deixar alguma ameaça passar batida pela segurança inicial. Com a análise heurística, esse invasor pode ser descoberto enquanto age silenciosamente.

Outra lógica usada para acabar com os malwares é a utilização de sandbox (*caixa de areia*). Nesse caso, o antivírus simula um ambiente (*como se fosse uma máquina virtual, emulando acesso ao registro e componentes*) para avaliar o comportamento de alguns arquivos e executáveis. Se a resposta for positiva, um alerta é disparado e a ameaça é enviada para análise (*já que não havia sido detectada antes*).

O antivírus que eu recomendo você usar é o Avira, mesmo que seja o free. Digo isso porque por algum tempo eu utilizei crypters, e acabei usando algumas técnicas para fazer os meus próprios, e o Avira era o AV mais difícil de ser removido. Recomendo também o Malwarebytes, que também é bem difícil de ser removido por algum crypter. Já utilizei os dois juntos, e não ocorreu nenhum conflito.

Para ficar seguro, um antivírus apenas basta, no máximo dois. Não adianta encher o PC de antivírus, isso só vai deixar a máquina lenta, e pode até fazer com que eles não funcionem direito, já que um entrará em conflito com o outro.

Mas a segurança não depende apenas do antivírus, você também deve tomar cuidado. Você deve se policiar na hora de baixar algum arquivo ou entrar em algum site suspeito. Por isso eu recomendo que você aprenda a analisar malwares. Como eu disse na primeira página do livro, escreverei uma segunda edição com os ensinamentos na prática, tudo explicadinho como na teoria. Mas isso é coisa mais para frente, porque isso dá bastante trabalho.

## **Firewall**

Outro software para aumentar a segurança em seu computador é o Firewall. Eles são bem úteis, e é super recomendado utilizá-los juntamente com um anti-virus.

Os antivírus são bons, mas eles podem não bloquear a infiltração de malwares pela rede. Alguns anti-virus até vêm com firewall próprio, mas a maioria é pago, como Avast ou Avira.

Um firewall bem configurado pode bloquear tentativas de conexões externas, registrar as tentativas de acesso aos serviços habilitados no seu pc, bloquear a saída indevida de dados do seu computador para algum site ou servidor (*isso é feito porque o malware pode capturar seus dados e enviar para o cracker via internet*), evitar que um malware já instalado no seu PC se propague, infectando arquivos e outros computadores da rede.

O próprio Windows já vem com um firewall próprio, mas não é muito bom. Eu recomendo o uso do Comodo Firewall. Ele é um ótimo firewall, e se for bem configurado, pode até 'blindar' o seu PC, de certa forma. Se um arquivo tentar modificar alguma chave no registro, ele vai dizer. Se tentar se conectar com algum servidor, ele vai avisar também, e a conexão só será feita se você permitir. Ele é um ótimo software, e é gratuito.

O Comodo possui um filtro anti phishing também, que é útil para não cair em páginas fakes.

Eles já vêm configurado 'de fábrica', mas é sempre bom você dar uma atualizada nas configurações para deixá-lo mais ativo. As configurações recomendadas pela Cartilha de Segurança são: liberar todo tráfego de saída do seu computador (*ou seja, permitir que seu computador acesse outros computadores e serviços*) e bloquear todo tráfego de entrada ao seu computador (*ou seja, impedir que seu computador seja acessado por outros computadores e serviços*) e liberar as conexões conforme necessário, de acordo com os programas usados.

Se você não sabe ao certo como configurar seu firewall, seja ele por software ou hardware, **NÃO** utilize o método das tentativas, pois isso pode colocar seu computador em grande risco, causando perda de dados, tráfego de informações indevidas e grandes dores de cabeça. Digo isso por experiência própria. Quando instalei o Comodo pela primeira vez, não sabia como configurar, e fui mexendo em várias coisas. Ele estava bloqueando tudo e não pedia permissão. Então eu removi, sem voltar às configurações padrão. Resultado: Fiquei sem internet por um bom tempo, até fazer a restauração do sistema e voltar ao que era antes. Prefira pedir ajuda para alguém que entende um pouco mais do assunto.

Já que estamos no embalo de segurança e anonimidade, no próximo capítulo falaremos um pouco sobre a deep web.

# 14. Deep web



Muito se fala hoje em dia sobre Deep web. Alguns têm medo, outros entram e acessam vários sites, outros sequer entram e falam como se já tivessem anos de experiência. Meu objetivo nesse capítulo é explicar de fato o que é Deep web, que na verdade é um termo que perdeu o significado por conta de sua polêmica.

***O conceito original de Deep web é: Conteúdo não indexado pelos buscadores.***

***“Como assim, Nick?”***

Se você entrar no Fórum Guia do Hacker, entrar em algum tópico, copiar o conteúdo dele e pesquisar isso no Google, ele vai retornar o link do tópico que você retirou o texto. Isso acontece porque o Fórum está indexado no Google, e em outros buscadores, como Bing, Yahoo e etc. Agora faça o teste: Entre no facebook e abra algum grupo secreto/privado. Faça a mesma coisa, copie o conteúdo de algum post e cole no Google. Nenhum resultado daquele grupo será retornado. Isso acontece porque o Facebook é configurado para não exibir o conteúdo de grupos secretos. Não só esse conteúdo, como também vários outros. É por esse motivo que não se pode encontrar conversas via chat de outras pessoas no Google.

Então, todo o conteúdo desse tal grupo privado está na Deep web, apenas por não ser indexado nas ferramentas de busca.

Estranho? Pois é, isso é para você ver como as pessoas modificam o conceito de uma coisa apenas para gerar polêmica.

Redes intranet com sistemas próprios também estão na deep web, pois seu conteúdo é acessível apenas para quem estiver na rede, e como os buscadores não estão, é impossível de encontrarem algo.

Sabe aquela frase “se não tá no google, não existe”? Pois é, ela está errada, e a prova disso está aí.

***“E como os sites fazem para algo não ser indexado?”***

Bom, existe uma ferramenta chamada robots.txt. É apenas um arquivo de texto localizado na raiz dos sites que permite que os motores de busca saibam o que fazer com determinado site. Veja você mesmo, abra o link <http://facebook.com/robots.txt>

Note que esse arquivo de texto contém os comandos “**User-agent: Buscador**”, seguido de vários “**Disallow: Pasta**”. O User-agent serve para definir qual engine de busca será responsável pela próxima sequência de comandos. Ou seja o comando

*User-agent: Googlebot*

*Disallow: /ajax/*

Faz com que o Google não indexe a pasta /ajax/ em suas pesquisas.

Notem que há várias pastas e arquivos como /feed/, /album.php, /photos.php, /hashtag/, de vários buscadores diferentes. Ou seja, nenhuma dessas pastas e nenhum conteúdo proveniente delas será indexado nesses buscadores. Tente fazer uma pesquisa em algum deles por algum conteúdo que esteja dentro dessa pasta. Você não vai achar, pois está na Deep web.

**“Mas e a Deep que nós conhecemos, com gore, black market e outras coisas?”**

Esse é a chamada rede TOR. O TOR é uma rede de proxys interligadas umas às outras, e faz com que todos os computadores sejam conectados entre si. Ou seja, para acessar um site nessa rede, o pacote sai do seu PC, viaja através da proxy e chega até outro PC, esse outro PC envia seu pacote para outro, e assim vai enviando até chegar no site que você queria, e o mesmo acontece ao contrário. Isso faz com que a rede fique muito lenta.

TOR significa The Onion Router, que traduzindo ao pé da letra seria algo como “O roteador da cebola”. Mas por que ele recebe esse nome? Bom, certamente se você já cortou uma cebola ao meio (e chorou), deve ter notado que ela contém várias camadas, como se fossem folhas, uma sobrepondo a outra, até formar a cebola completa. É mais ou menos isso que acontece dentro do tor, mas espere, explicarei sobre isso melhor agora.

Na verdade não existem camadas, nunca existiram. Esse termo é muito, muito utilizado na internet, de forma errônea, principalmente pelos que não entendem de redes. As camadas na qual eles se referem são na verdade as linhas de conexão das proxys.

Veja o desenho que montei abaixo para exemplificar algumas “camadas”.

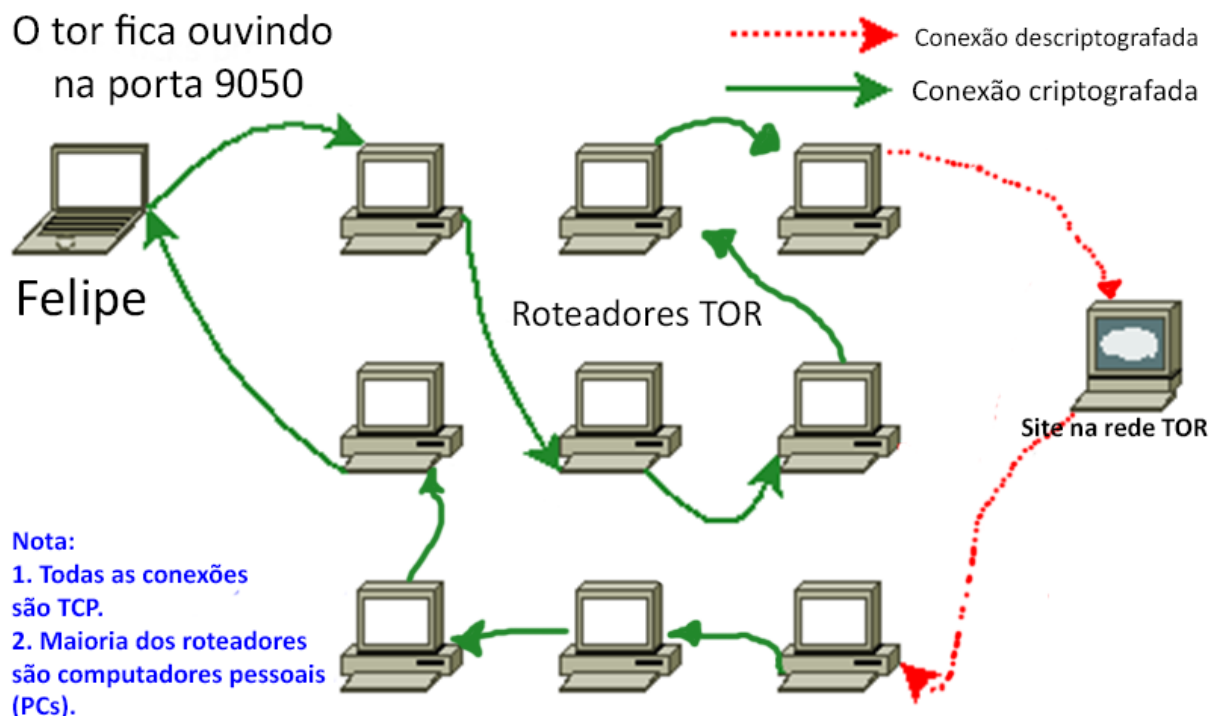


Imagem: rMonteux. Edição: Nicholas Ferreira.

Como você pode ver, quando Felipe deseja se conectar a um site na rede TOR, sua requisição passa por vários computadores, com faixas de IP diferentes, até chegar no site, que retorna então o conteúdo para Felipe, que novamente é passado por vários outros computadores até chegar nele. Essas que são as camadas.

Então já sabe, quando ouvir alguém falar sobre “primeira camada, sétima camada, marianas web, blablabla”, já sabe que é tudo conversa fiada, nada disso existe.

Existem outras redes parecidas com o TOR, como a i2p, Freenet, Closs (Closed Shell System), e elas também são chamadas de “camadas”. São na verdade outros aglomerados de computadores interligados pela rede proxy. Algumas dessas redes são bem complicadas de se conseguir conectar, como a rede Closs. E como são mais difíceis, obviamente há mais conteúdo privado e ilegal lá dentro.

Na própria .onion do TOR tem bastante coisa, como o Black market (mercado negro, vende armas e drogas), Assassination Market (Contrate um hitman para matar alguém), Pedochan (Fórum de pedofilia), Além de vários sites de gore (fotos e vídeos de coisas bizarras, como um homem cozinhando um bebê), e etc... O que eu disse até aqui é experiência própria, coisas que eu mesmo vi, não estou inventando nada. Há também fóruns e sites hackers, com conteúdos bem interessantes, além de várias bibliotecas com GBs de livros em PDF para baixar. (*Quem sabe um dia esse livro vá pra lá, rs*)

Existe também o TorMail, um sistema de envio de email anônimo que utiliza criptografia pesada para garantir a segurança e integridade dos dados. Você tem que usar sua conta no tormail para se cadastrar em sites no TOR (*ou vai querer deixar seu email exposto? Nunca sabemos as técnicas descobertas que não foram divulgadas ainda*). O cadastro é gratuito, e o modelo é [email@tormail.org](mailto:email@tormail.org).

Há quem diga que o governo de alguns países escondem informações confidenciais lá (principalmente os EUA, com assuntos sobre Área 51, caso Roswell, etc), mas eu não acredito muito. Seria muita ingenuidade da parte deles hospedar conteúdo confidencial em uma rede com vários hackers/crackers, dos mais diversos países.

Mas há bastante conteúdo, tanto sobre hacking quanto sobre outras coisas. Já baixei vários livros na TorLibrary, todos em PDF. Mas você deve tomar cuidado ao baixar alguma coisa, não pode sair baixando tudo que tem lá, por isso, mantenha seu antivírus atualizado e seu firewall bem configurado.

## 15. FAQ



Não escreverei o FAQ aqui porque ficaria muito grande, então disponibilizarei o link para que vocês acessem. O tópico foi escrito por **Void\_Witch**. Link do tópico abaixo.

<http://forum.guiadohacker.com.br/showthread.php?t=36173>

Se não funcionar, copie e cole no navegador.

Então é isso. Eu espero de coração que todos tenham entendido tudo que eu expliquei aqui, espero que não fiquem com raiva por eu não ter ensinado nada na prática (*farei isso no próximo livro*).

Qualquer dúvida sobre algum conteúdo, escrevam no tópico em que este livro foi postado que eu tentarei responder assim que possível.

Deu muito trabalho para escrever tudo de forma explicativa, formatar, inserir imagens e deixar tudo com uma boa linguagem para que iniciantes possam entender, então eu peço encarecidamente que, por favor, **se for copiar qualquer parte deste livro, deixe os devidos créditos**.

Se você não gostou do livro porque acha que o conteúdo é bem básico, bom, sinta-se feliz, você já passou da fase difícil que é entender o básico. E se você gostou, eu me sinto feliz por ter te ajudado a iniciar nos estudos.

Apenas ler este livro – ou outros do mesmo conteúdo – não te faz ser um hacker. Você precisa estar estudando sempre, para que esteja em constante evolução. Nunca deixe para trás só porque você não consegue. Não sou muito adepto à frases motivacionais, mas sei como é se sentir frustrado por não conseguir fazer algo que quer muito. Mas já pensou se Newton ou Nikola Tesla tivessem desistido nos primeiros erros? Pois é, a história seria outra, e talvez você nem estaria lendo isso.

Sou ruim para finalizações e conclusões, mas vou ficando por aqui. Mais uma vez, se você está lendo isso, obrigado pela paciência. Qualquer erro, desculpe-me, é o primeiro grande 'artigo' que escrevo (72 páginas!), então tenho um desconto, rsrs. **Até a próxima, tchau e bons estudos!**

# FIM

# 16. Referências bibliográficas



BestVPN: Proxies & VPNs  
Blog Inurl: Google hacking dorks  
Brutal Security: Footprint x Fingerprint  
CanalTech: Packers  
CM: The 10 most destructive PC viruses  
ComodoBR: SSL  
Crimes Cibernéticos: Análise de malware  
Digitro: Linguagens de programação  
Dipartimento di Informatica: Rete di computer  
Exploit-db: About Oday exploits  
Fergonez: Ferramentas para Engenharia reversa  
Fórum Guia do Hacker: "Como Crackers conseguem acesso ao seus dados" by Hackuv  
Fórum Guia do Hacker: "Como funciona um KL Bank feito em Delphi" by Hackuv  
Fórum Guia do Hacker: "Como os Ladrões agem ao Clonar seu cartão" by Hackuv  
Fórum Guia do Hacker: "FAQ" by Void\_Witch  
Fórum Guia do Hacker: "Para roubar um caixa eletrônico, basta teclado e pendrive" by Hackuv  
IANA: Service Name and Transport Protocol Port Number Registry  
Infowester: Bluetooth  
Kaspersky: Tunneling Protocol  
Keelog: Hardware Keylogger  
Kevin Mitnick: A arte de enganar  
Kevin Mitnick: A arte de invadir  
Mashable: From Heartbleed to the NSA  
NSForus: DDoS Attacks  
Oficina da Net: Malwares  
Oficina da Net: Tipos de criptografia  
Olhar digital: Segurança em VPNs  
OYS: Análise de malware  
OYS: Exploiting  
Paradigmas de Linguagens de Programação  
PCWorld: Researches about CryptoLocker  
Rapid7: A guide for exploits  
Rapid7: Logs  
RSWebMarketingDigital: No Tech Hacking  
TechTudo: Emuladores virtuais  
Tecmundo: Carregador de iPhone modificado  
Tecmundo: Engenharia reversa  
Tecmundo: Vulnerabilidades no SSL  
Viva o Linux: A arte de desconfigurar sites  
Viva o Linux: Backdoor  
Wikipédia: IPSec to IPv6  
Wikipédia: Universal Serial Bus  
Wingwit: Descompiladores  
Wingwit: Networking