O slackware é a distribuição linux mais antiga ainda em atividade. Tendo sido criada por Patrick Volkerding em 1993, a partir da SLS.

Em todos esses anos, a distro conquistou ardorosos utilizadores, principalmente graças à sua filosofia de simplicidade e estabilidade.

Um produto de extrema qualidade para usuários com esta mesma característica. E este zine é de slacker para slacker.

slackware zine

30 de Setembro de 2005 – Edição #11

Editorial

De novo no último segundo! SlackwareZine, o zine que vive perigosamente -;). Falando sério, esse foi um bimestre bastante animado, com o SlackwareShow e o lançamento do **slackware** 10.2.

Vou fazer um jabá aqui e agradecer às empresas que apoiaram o evento: LinuxMall, Linux Magazine Brasil, TempoReal e a FIAP que cedeu o espaço e infraestrutura para o evento.

Vale a pena agradecer também a todos os palestrantes, sendo que alguns ainda pagaram a viagem do bolso e a todos que participaram do evento! Espero que todos tenham gostado e muito. Pela gente da zine, ano que vem tem mais -;)

Falando dessa edição. Temos dois artigos de configuração de hadware (um de tablets e outro tratando de uma placa Gigabit), dois para interfacear com dispositivos externos (iPod e Palm) e um sobre o funcionamento do SSH.

Meus agradecimentos aos autores e faço aqui um “chamamento” para que nossos leitores mandem mais e mais artigos. Pois é com a participação de vocês que podemos ir para frente. E tornar a publicação cada dia melhor.

Boa Leitura!

Piter PUNK

Índice

slack users

Configurando Tablets no X

Ok, esse não é um hardware dos mais comuns, mas quem tem acesso a um deles vive encafifado sobre como fazê-lo funcionar. Do que estou falando? Estou falando de mesas digitalizadoras (ou tablets, para os íntimos).

No meu trabalho temos dois desses, um da Genius (NewSketch 1212HRIII) e um da Wacom (Graphire3). E isso é particularmente legal, já que cada um deles utiliza um driver e uma interface de conexão diferente -:). E ambos funcionaram no Linux!

Nenhum dos configuradores (xorgcfg, xorgconfig, xorgsetup, etc...) dá suporte a tablets, o que é bem estranho, já que o usuário de tablets costuma ser um designer e não um técnico em computação. Ou seja, aparecer no configurador ia ser bem legal. A configuração básica é simples, veja abaixo:

1. Genius NewSketch 1212HRIII

Esse tablet é ligado na porta serial, com a alimentação vindo pelo conector de teclado. Ele vem com uma caneta e dois "mouses".

A primeira coisa a fazer é conectar tanto a caneta quanto um dos "mouses" no tablet. Se for ligado com os dois conectados, o tablet entra no modo compatível com o "Summa" que é exatamente o que queremos.

O truque de conectar o "mouse" foi o primeiro para fazer com que esse tablet funcione. O segundo é um pouco mais complexo. Precisamos que o GPM faça a inicialização do tablet, antes do X. Para isso, edite o arquivo /etc/rc.d/rc.gpm trocando seja lá o que estiver depois do "-t" por summa e o /dev/mouse para /dev/ttyS0 (ou ttyS1 se o seu tablet estiver na segunda serial). Por exemplo, onde está:

```
/usr/sbin/gpm -m /dev/mouse -t imps2
```

Troque para

```
/usr/sbin/gpm -m /dev/ttyS0 -t summa
```

Nada que uma substituição rápida no VI não faça. Com o rc.gpm editado, reinicie o GPM:

```
# /etc/rc.d/rc.gpm restart
```

Em seguida, abra o /etc/X11/xorg.conf, para fazermos a configuração propriamente dita (afinal, ninguém quer um tablet para ficar copiando e colando textos com o gpm).

Na seção "Module", bem no começo do arquivo, adicione:

```
Load "summa"
```

Logo após a configuração do mouse, adicione mais uma seção do tipo "InputDevice" (dessa vez é para o nosso tablet):

```
Section "InputDevice"
        Identifier "EasyPen"
        Driver "summa"
        Option "Device" "/dev/ttyS0"
        Option "InputFashion" "Tablet"
        Option "Mode" "Absolute"
        Option "Name" "EasyPen"
        Option "Compatible" "True"
        Option "Cursor" "Stylus"
        Option "Protocol" "Auto"
        Option "SendCoreEvents" "on"
        Option "Vendor" "GENIUS"
EndSection
```

Lembre-se que onde está /dev/ttyS0 deve ficar a serial em que o seu tablet está conectado. ttyS0 refere-se à primeira serial e ttyS1 à segunda.

Agora, na seção "ServerLayout" inclua:

```
InputDevice "EasyPen" "SendCoreEvents"
```

Basta reiniciar o X e o seu tablet já estará funcionando! -:) São possíveis várias outras configurações. Utilizar as opções "MinX", "MaxX", "MinY" e "MaxY" para delimitar a área em que se pode escrever na tablet são algumas das melhores, já que sem elas o mapeamento fica meio alienígena, de uma área quadrada (o espaço do tablet) para uma retângular (o monitor).

2. Wacom Graphire3

Esta é uma placa bem mais nova, com bem mais recursos e que usa interface USB. Para "sentir" a diferença, basta conectar e (se o seu X lê o mouse através do /dev/input/mice) usar! Claro, com isso já dá para ter uma idéia da facilidade. Enquanto o tablet da Genius foi testado tanto no kernel 2.4.31 como no 2.6.13, este foi testado apenas utilizando o 2.6 (aquele que se encontra no /extra do **slackware** 10.2).

Antes de configurar o X, devemos apenas dar uma olhada no arquivo /proc/bus/input/devices o que interessa para a gente é um trecho semelhante a este:

```
I: Bus=0003 Vendor=056a Product=0014
Version=0312
N: Name="Wacom Graphire3 6x8"
P: Phys=usb-0000:00:10.0-1/input0
H: Handlers=mouse1 event2
B: EV=f
B: KEY=1c43 0 70000 0 0 0 0 0 0 0 0
B: REL=100
B: ABS=3000003
```

Repare no "Name", ali sabemos que estamos falando do tablet. Devem haver mais outros dois dispositivos nesse arquivo, o teclado e o seu mouse. Na linha em que está "Handlers" existe uma informação importante, qual o evento que está associado a ao tablet: event2.

Com o nome do evento em mãos, vamos editar o /etc/X11/xorg.conf. É bem simples e rápido. Devemos inserir três novos InputDevices, deixe-os próximos da seção que trata do mouse, para manter o /etc/X11/xorg.conf organizado.

```
Section "InputDevice"
    Identifier  "Mouse2"
    Driver      "wacom"
        Option "Device" \
                  "/dev/input/event2"
        Option "Type" "cursor"
        Option "USB" "on"
        Option "Vendor" "WACOM"
EndSection

Section "InputDevice"
    Identifier  "Mouse3"
    Driver      "wacom"
        Option "Device" \
                  "/dev/input/event2"
        Option "Type" "stylus"
        Option "USB" "on"
        Option "Vendor" "WACOM"
EndSection
```

```
Section "InputDevice"
    Identifier  "Mouse4"
    Driver      "wacom"
        Option "Device"
"/dev/input/event2"
        Option "Type" "eraser"
        Option "USB" "on"
        Option "Vendor" "WACOM"
EndSection
```

Perceba que indicamos na opção "Device" o /dev/input/event2, já que event2 é o evento em que o tablet "fala".

Depois de incluídas as três novas entradas, vá para a seção “ServerLayout” e insira as seguintes linhas:

```
InputDevice "Mouse2" "SendCoreEvents"
InputDevice "Mouse3" "SendCoreEvents"
InputDevice "Mouse4" "SendCoreEvents"
```

Agora é só iniciar o X e ir para o abraço. Sério.

Existem utilitários específicos para regular e fazer o ajuste fino do seu tablet, você pode encontrá-los em http://linuxwacom.sourceforge.net/, junto com uma farta documentação. Boa Sorte e bons desenhos!

Piter PUNK <piterpk@terra.com.br>

Configurando uma Placa Gigabit

Recentemente, durante mais um processo de "evangelização" realizado aqui onde trabalho, um amigo teve alguns problemas com um notebook Toshiba M45 S311.

Todo o hardware da máquina foi reconhecido, o único problema foi a placa Ethernet, uma SysKonnect SK-98xx Gigabit. As versões mais recentes do kernel já possuem um módulo para placas SysKonnect/Marvell denominado `sk98lin`, no entanto, o modelo em questão, uma `Marvell Technology Group Ltd. 88E8036 Fast Ethernet Controller (rev 10)`, não é suportada pelo release mais atual do kernel, pelo menos não até o momento em que esse texto estava sendo redigido.

Bem, felizmente, existe uma atualização desse módulo que permite a utilização da placa em questão, vamos ao trabalho:

```
root@papaleguas:~$ wget \
      http://www.syskonnect.de/\
      syskonnect/support/driver/\
      zip/linux/install-8_23.tar.bz2
```

Também existe uma documentação oficial, fornecida pelo fabricante em:

```
http://www.syskonnect.de/syskonnect/\
      support/driver/readme/\
      linux/sk98lin.html
```

E um guia de instalação em:

```
http://www.syskonnect.de/syskonnect/\
      support/driver/readme/\
      linux/README.htm
```

Segundo o fabricante, o `patch` é compatível com as séries 2.6 e 2.4 do `kernel`, esse último a partir do release 2.4.13.

Instalando:

```
root@papaleguas:~$ tar -xvjf \
          install-8_23.tar.bz2
root@papaleguas:~$ cd DriverInstall
root@papaleguas:~$ ln -s \
      /usr/src/linux_versão_do_kernel \
      /usr/src/linux
root@papaleguas:~$ ./install.sh
```

Vale lembrar que o comando:

```
root@papaleguas:~$ ln -s \
      /usr/src/linux_versão_do_kernel \
      /usr/src/linux
```

Cria um link simbólico dos fontes de seu `kernel` atual no diretório `/usr/src/linux`, permitindo ao `script` funcionar de maneira genérica, independente da versão do `kernel` utilizada, desde que a mesma seja suportada.

Desse modo, os fontes de seu kernel são obrigatórios para esse processo, informação óbvia, no entanto, não custa lembrar...

Se tudo correu bem, você irá receber a informação que para utilizar sua placa de rede, basta um "modprobe sk98lin".

Bem, isso pode ser ou não verdade. No caso do notebook em que realizei a instalação não foi o suficiente. Vejamos:

```
root@papaleguas:~$ modprobe sk98lin
root@papaleguas:~$ ifconfig eth0 up
root@papaleguas:~$ ifconfig eth0 \
      192.168.0.5 netmask 255.255.255.0
root@papaleguas:~$ route add \
      default gw 192.168.0.1
root@papaleguas:~$ vim /etc/resolv.conf
```

(Adicione os servidores DNS do seu provedor)

```
root@papaleguas:~$ lynx
```

Deveria funcionar, mas não funciona... :(

O ifconfig retorna as informações que deveria retornar, um cat /proc/net/sk98lin/eth0 também...

Bem, ao que parece, nosso sistema não consegue se comunicar de fato com o dispositivo. Vamos ver:

```
root@papaleguas:~$ dmesg | more
PCI: Using ACPI for IRQ routing
PCI: if you experience problems, try using
option 'pci=noacpi' or even 'acpi=off'
```

Não custa nada tentar. Principalmente sabendo que o suporte a ACPI é bem “estranho”. Para editar a configuração do LILO utilizando o VIm, digite:

```
root@papaleguas:~$ vim /etc/lilo.conf
```

Adicione a seguinte linha após as configurações referentes ao seu kernel:

```
append = "pci=noacpi"
```

Atualize o LILO::

```
root@papaleguas:~$ lilo
```

Caso use o GRUB:

```
root@papaleguas:~$ vim \
                  /boot/grub/grub.conf
```

Adicione o seguinte parâmetro no final da linha referente às configurações do seu kernel:

```
kernel /boot/vmlinuz-2.x.xx-x... \
      --> Adicione isso: pci=noacpi
```

Reinicie a máquina...

Repita os passos anteriores:

```
root@papaleguas:~$ modprobe sk98lin
root@papaleguas:~$ ifconfig eth0 up
root@papaleguas:~$ ifconfig eth0 \
            192.168.0.5 netmask \
            255.255.255.0
root@papaleguas:~$ route add \
      default gw 192.168.0.1
```

E pra fechar com chave de ouro:

```
root@papaleguas:~$ lynx \
      www.slackwarezine.com.br
```

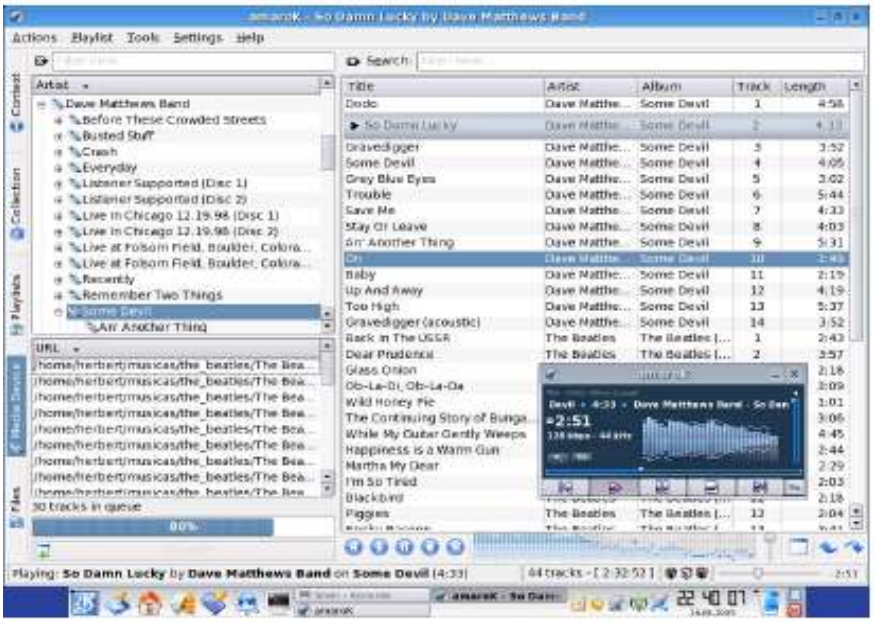
Navegue a vontade!!!;)

Agora basta inserir os parâmetros em seu /etc/rc.d/rc.inet1.conf manualmente ou via netconfig e pronto, sua rede irá funcionar automaticamente. Lembre (é claro) de colocar o módulo para ser carregado no /etc/rc.d/rc.modules.

Clayton <clayton@slackpoint.com.br>

Como utilizar o iPod no **slackware**

iPod é um player portátil de audio digital projetado pela Apple. Todos os modelos recentes do iPod oferecem suporte a portas FireWire, USB 2.0 e USB 1.1.



Os dados são armazenados em um disco rígido interno formatado como FAT32, portanto utilizar o iPod no **slackware** é bem simples, basta montarmos o dispositivo e manipulá-lo através de qualquer software já existente, como o `gtkpod` ou o `amaroK`

Ao plugarmos o cabo USB do iPod o `Kernel` (no meu caso o 2.4.29) detecta o dispositivo e consequentemente o respectivo módulo (`usb-storage`) é carregado:

```
# dmesg | grep -i usb
hub.c: new USB device 00:1d.7-1, \
                  assigned address 5
USB Mass Storage device found at 5
```

Caso isto não ocorra faça o seguinte:

```
# insmod usb-storage
Using /lib/modules/2.4.29/\
           kernel/drivers/\
           usb/storage/usb-storage.o.gz
```

Agora basta checar qual partição no respectivo dispositivo de disco contém os dados do iPod:

```
# fdisk -l /dev/sda | grep sda
Disk /dev/sda: 4095 MB, 4095737856 bytes
/dev/sda1   *   1   5   40131   0  Empty
/dev/sda2   *   6 497 3951990   b  W95 FAT32
```

Ou seja, o dispositivo em questão utiliza o `/dev/sda2` (o iPod em questão é o mini, de 4G). Finalizando, basta montá-lo e acessá-lo através de um dos softwares citados:

```
# mkdir /mnt/ipod
# mount -t vfat /dev/sda2 /mnt/ipod
# umount /mnt/ipod
```

Podemos também adicionar uma entrada ao `/etc/fstab` para qualquer usuário ser capaz de utilizá-lo:

```
$ cat /etc/fstab | grep ipod
/dev/sda2 /mnt/ipod vfat noauto,user 0 0
```

Para checar se tudo está funcionando como previsto:

```
$ mount /mnt/ipod
$ mount | grep -i ipod
/dev/sda2 on /mnt/ipod type vfat \
    (rw,noexec,nosuid,nodev,user=herbert)
```

Agora basta usar o seu player de som favorito para escutar as músicas do seu iPod, como no screenshot acima (em que o AmaroK está tocando e transferindo músicas direto para o iPod).

Herbert <herbert@faleiros.eti.br>

Usando o Palm Tungsten E2 no **slackware**

Recentemente adquiri um Palm Tungsten E2 da palmOne e como de costume, todos os softwares que acompanham o produto só possuem versão para "aquele" "Sistema Operacional" (Não pude evitar as aspas no "Sistema Operacional"). ;)

Bem, felizmente, a comunidade não depende dos fabricantes para utilizar seus hardwares em Sistemas Operacionais livres, desse modo, existem várias alternativas existentes, como por exemplo:

- `kpilot` (disponível nos CD's de instalação do **`slackware`**);
- `jpilot` (disponível em: `http://www.jpilot.org/download.html`).

Para fazer com que os modelos mais novos, baseados em conexão USB, possam ser sincronizados utilizando o `kpilot/jpilot`, são necessários alguns passos. Vamos ao trabalho!!!

Primeiro, devemos "subir" dois módulos, o `usbserial` e o `visor` (presumindo que o `hotplug` não os levantou):

```
root@papaleguas:~# modprobe usbserial
root@papaleguas:~# modprobe visor
```

O módulo `usbserial` provê conexão entre um dispositivo USB e o micro, provendo transferência de dados de maneira semelhante a uma conexão serial.

Já o módulo `visor`, é referente a dispositivos palmOne, Sony Clie e Handspring Visor. Também existem módulos para Compaq iPAQ, HP Jornada e Casio EM500 Driver, entre outros. Verifique se o seu `kernel` possui suporte ao seu hardware na seção `USB Support/USB Serial Converter support`.

Vejamos quais são os módulos USB que estão ativos:

```
root@papaleguas:~# lsmod | grep usb
usbserial               21404   0  [visor]
usb-storage             65216   0  \
(unused)
usb-ohci                19496   0  \
(unused)
usbcore                 62508   1  \
[visor usbserial usb-storage usb-ohci
ehci-hcd]
```

Tudo certo!!!

Agora vamos remover o link simbólico que o `kpilot` cria e criar um novo:

```
root@papaleguas:~# rm /dev/pilot
root@papaleguas:~# ln -s /dev/ttyUSB1 \
                            /dev/pilot
```

Vejamos se está tudo ok:

```
root@papaleguas:~# ls -la /dev/pilot
lrwxrwxrwx  1 root root 7 2005-09-29 \
            21:13 /dev/pilot -> ttyUSB1
```

Em seguida, conecte seu Palm na porta USB, aperte o botão de sincronismo do cabo de conexão e abra o `kpilot`.

Em um terminal digite:

```
root@papaleguas:~# lsusb
Bus 003 Device 001: ID 0000:0000
Bus 002 Device 001: ID 0000:0000
Bus 002 Device 013: ID 0830:0061 \
                         Palm, Inc.
Bus 001 Device 001: ID 0000:0000
Bus 001 Device 002: ID 0f2d:9308 \
                         ViPower, Inc.
```

Tenha certeza que o seu device está "linkado" com a porta USB correta.

Agora é só esperar o backup terminar... 8^)

Até o próximo artigo,

`Clayton <clayton@slackpoint.com.br>`

Controlando o PowerDownload da Telefônica

Power Download é um serviço do Speedy da Telefonica (aproximadamente R$35,00 mensais) que "turbina" a conexão do Speedy durante a noite e finais de semana. Por exemplo, meu Speedy é o 600 (agora 750), ao ativar o Power Download (após as 20hs e finais de semana) o mesmo passa a funcionar como se fosse um Speedy de 1Mbit

Como era de se esperar a Telefonica sempre dificulta as nossas vidas e este serviço deve ser ativado manualmente (todos os dias) pelo Speedyzone (a interface web deles `http://www.speedyzone`). Curiosamente, mesmo ao ativarmos o serviço religiosamente (todo santo dia) às 20hs, muitas vezes ele deixa de funcionar do nada durante à noite, nos obrigando a reativarmos o serviço sempre que isso acontece.

Para resolver este problema criei um script que automatiza todo este processo, ou seja, ativa automaticamente o serviço (todos os dias) por volta das 20hs e verifica de hora em hora se o mesmo está realmente funcionando, caso negativo (parou de funcionar misteriosamente) o script ativa novamente o serviço.

Uma outra vantagem deste esquema é termos controle sobre o que estamos pagando (pelos logs gerados).

O script é bem simples:

```
#!/bin/bash
#
# /usr/libexec/pdownload
# Herbert Alexander Faleiros
# <herbert@faleiros.eti.br>
#

# Insira um usuário com UID alta (não use
# o root). Esta é uma medida de segurança
# adicional, ou seja, evita que o root
# acesse a net desnecessariamente.
USER=

if [ "${USER}" == "" ]; then
  clear && echo "Configure o \
            script antes de executá-lo!"
  exit 1
fi
# 1024 = 1M
# Speedy 600/750 => Power Download: 1M.
SPEED=1024
BASE="http://www.speedyzone/\
                    speedywebapp/servlet"
EXEC="${BASE}/\
            StartStopService?servicio\
            =speedynight${SPEED}&accion"

# Exibe o status atual do serviço.
function status() {
  echo -n "Power Download \
      ${SPEED} \
      ($(date '+%Y-%m-%d %H:%M:%S')): "
  su ${USER} -c \
      "lynx --dump \
      ${BASE}/activacionServicios " | \
      grep -q green.gif && \
      echo ativado || echo desativado
}

# wget - opções utilizadas (não alterar).
OPTS="--quiet --spider"

START=activar
STOP=desactivar

# Ativa o Power Download.
function start() {
  su ${USER} -c \
      "wget \"${OPTS} ${EXEC}=${START}\""
  . $0 status
}

# Desativa o Power Download.
function stop() {
  su ${USER} -c \
      "wget \"${OPTS} ${EXEC}=${STOP}\""
  . $0 status
}

case $1 in
  status)
    eval $1
  ;;
  start)
    eval $1
  ;;
  stop)
    eval $1
  ;;
  *)
    echo "Usage: $0 {start|stop|status}"
    exit 1
  ;;
esac
```

Feito isto criamos um wrapper para evitarmos a inserção dos parâmetros do script no crontab.

```
#!/bin/bash
#
# /usr/bin/pdownload
# Herbert Alexander Faleiros
# <herbert@faleiros.eti.br>
#

# Script original.
EXEC=/usr/libexec/pdownload

# Verifica o status do serviço, caso o
# Power Download esteja desativado, ativa
# o serviço.
(${EXEC} status | \
      grep -q desativado && \
      ${EXEC} start) &>/dev/null

# Verifica novamente o status e grava o
# resultado no arquivo de log
# especificado pelas regras no cron.
. ${EXEC} status
```

Após os scripts serem criados e alocados corretamente nos locais especificados, basta inserirmos as seguintes regras no crontab:

```
# segunda à sexta, de hora em hora,
# das 20:05 à 7:05
5 20-7 * * mon-fri pdownload >> \
                  /var/log/messages 2>&1

# sábado e domingo, de hora em hora, o
# dia todo
5 * * * sat-sun pdownload >> \
                  /var/log/messages 2>&1
```

Feito isto nos certificamos de que as permissões/donos dos dois scripts estão corretamente configurados.

```
# chown 0.bin /usr/bin/pdownload
# chown 0.0 /usr/libexec/pdownload
# chmod 755 /usr/{bin,libexec}/pdownload
```

Exemplo dos logs gerados:

```
# cat /var/log/messages | \
                  grep -i download
Power Download 1024 (2005-09-11 \
                  13:05:01): ativado
Power Download 1024 (2005-09-11 \
                  14:05:03): ativado
```

Com isto evitamos (re)ativarmos o serviço manualmente e passamos a ter certo grau de controle sobre o que estamos pagando. :)

Herbert <herbert@faleiros.eti.br>

Autores

Clayton Eduardo dos Santos, Trabalha com Linux desde 2003 e com Slackware desde 2004. Atualmente desenvolve seu projeto de pesquisa de Doutorado no Departamento de Engenharia Elétrica na USP de São Carlos e é um dos administradores do slackpoint, portal 100% voltado à comunidade slackware brasileira.

Diego Fiori de Carvalho, é aluno do Bacharelado em Informática do ICMC-USP, SãoCarlos S.P.,utiliza Linux desde 2002, desenvolveu um sistema de multimídia interativa para Treinamento em Linux, tem grande interesse por desenvolvimento de aplicações gráficas e atualmente é desenvolvedor da empresa 3WT de São Carlos.

Herbert Alexander Faleiros (aka ratmmmam), programador, graduando em Física pela UFSCar. Seu primeiro contato com o Slackware ocorreu em 2000. Atualmente trabalha desenvolvendo soluções em Java para servidores de aplicações, nas horas vagas escreve artigos sobre criptografia e build scripts para o Slackware.

Piter PUNK, é mantenedor e principal desenvolvedor do slackpkg. Possui experiência com UNIX e Linux desde '96 tendo escrito diversos artigos em revistas da área, atualmente, trabalha como desenvolvedor de jogos na 3WT Corporation

Criptografia no SSH

Criptografia no SSH (Secure Shell)

Secure Shell é um programa/protocolo para acesso remoto de computadores, utilizando por padrão a porta 22 de sua máquina. Originalmente ssh é um
produto comercial, mas graças a nosso amigos do OpenBSD (`www.openbsd.org`), foi criado uma versão livre, chamada openssh, utilizado pelos nossos Linux amplamente.

O SSH difere-se principalmente, do Telnet (programa de acesso remoto muito utilizado, há pouco tempo atrás), por ter uso de criptografia na seção, corrigindo possíveis problemas de "sniffer"(escuta do canal), onde a senha do usuário passava sem nenhum tipo de criptografia no modo plain(texto puro), possibilitando a sua captura por possíveis invasores.

Outra vantagem do SSH está na implementação de trocas de chave públicas e privadas com algoritmo de assinatura digital para autenticação entre cliente e servidor.

Como funciona a criptografia no SSH?

Bom, para responder esta pergunta vamos primeiro entender os conceitos.

A criptografia é um ramo da matemática aplicada que pretende abordar a segurança na cifragem de mensagens sendo estas de qualquer tipo. Outra definição mais amigável, diz que é entendido como o estudo dos princípios e técnicas pelas quais as informações podem ser transformadas, dificultando a leitura por pessoas não autorizadas.

Primeiramente, é interessante saber como um algoritmo de criptografia é considerado seguro. Antigamente, até o fim da década e, tem-se mudado de 70, todos os algoritmos criptográficos eram secretos, guardados seus segredos funcionais e implementacionais a sete chaves. Mais recentemente este conceito de resguardar informações e os algoritmos são publicados mundialmente emconferências como a CRYPTO da International Association for Cryptologic Research (www.iacr.org). Este intuito de disseminação do estudo de algoritmos criptográficos, objetiva a análise de pesquisadores especializados que conheçam métodos sofisticados para atacá-lo. Caso seja negativo a maioria dos resultados de ataques, a sociedade aceita o algoritmo como seguro.

Neste artigo vamos entender os algoritmos de criptografia usados no SSH. O protocolo 2.0 do SSH utiliza o algoritmo DSA(Digital Signature Algorithm) baseado no RSA para autenticação de senhas. Para autenticação de mensagens ele utiliza uma criptografia com hash, usando SHA e MD5. E não acabou....ainda tem criptografia da seção utilizando o algoritmo BlowFish, DES, TwoFish.. Calma....Vamos com Calma...tudo será detalhado neste artigo a seguir...

Chaves Assimétricas

O RSA foi publicado em 1978 e seus autores são: Ron Rivest, Adi Shamir e Len Adleman. Este algoritmo é baseado na dificuldade computacional de fatorar um número inteiro em primos. Ele utiliza duas chaves diferentes para criptografia, uma para criptografar e outra para descriptografar, introduzindo o conceito de chave pública e privada. Tendo as duas chaves em mãos, uma mensagem que foi criptografada com uma, só pode ser descriptografada com outra. Mais especificamente, só pode ser descriptografada com a chave privada uma mensagem que foi criptografada com sua respectiva chave pública.

Como sugerem os nomes, a chave pública pode ser distribuída a vontade, enquanto que a privada deve ser guardada e protegida. Este processo chama-se troca de chaves assimétricas.

Existe uma limitação neste processo, pois o desempenho é lento para criação destas chaves a todo momento, por este motivo que elas são geradas somente quando necessário e armazenadas no arquivo / home/YOUR_USER/.ssh/know_hosts

A primeira vez que uma máquina conecta-se a outra, por meio do protocolo ssh, o cliente recebe a chave pública da máquina servidor e pede a confirmação da conexão. Desse modo, a máquina servidor é adicionada à tabela do arquivo know_hosts(nós conhecidos) e é pedida a senha do usuário, para estabelecer a conexão.

Nas próximas conexões só será solicitada a senha do usuário, pois a chave pública do host remoto já está guardada na máquina local.

Assinatura Digital

O SHA(Secure Hash Algorithm) é umalgoritmo hash do tipo Message Digest, desenvolvido pela NSA Security. Parecido com o famoso MD5, o SHA transforma uma mensagem em uma string de tamanho fixo de 128 bits. Este resultado dá-se o nome de hash(embaralhamento), o qual tem 2 propriedades:

1. Praticamente impossível encontrar duas mensagens que resultam no mesmo hash.

Pois o valor é obtido através de vários cálculos contendo operações lógicas(XOR, OR, AND), sempre visando atenuar próximo do limite zero as colisões da função de espalhamento, é o estado da arte em criptografia de apenas uma via, ou seja, uma vez criptografado você não tem a descriptografia.

2. Se a comparação entre dois hashs for positiva, temos a garantia da procedência da mensagem.

Com estas propriedades o hash serve para fornecer uma espécie de identificação única da mensagem.

Faça um teste, vamos utilizar o MD5 para isto.. crie um arquivo texto simples chamado teste com o seguinte conteúdo: `slackware`. Logo após, crie outro arquivo simples, chamado teste1 com o seguinte conteúdo: `slackwarezine`.

Vamos utilizar o comando md5sum, para gerar um hash de 128 bits dos arquivos, como descrito a seguir:

```
diego@osiris:~$ md5sum teste
0c775fcbacc4054457ace755b12e06ab teste
diego@osiris:~$ md5sum teste1
58df1dcb93b788d5f954d62255870aeb teste1
```

Perceba que os valores hash gerados, são totalmente diferentes para palavras tão parecidas como `slackware` e `slackwarezine`. Criptografia da seção (algoritmos de chave simétrica).

Além dos algoritmos de chaves assimétricas e assinatura digital, o ssh também utiliza criptografia da seção através de algoitmos de chave simétrica. Neste caso vários algoritmos são recorridos para esta finalidade, entre eles, podemos citar: DES, 3DES, BlowFish, TwoFish.

Para exemplificar, relataremos neste artigo o algoritmo Blowfish. Este algoritmo foi desenvolvido por Bruce Schneier em 1993, e muitos dizem que
é uma versão melhorada do algoritmo DES. Ele tem processamento mais rápido que algoritmos similares de chave secreta como DES(Data Encyption Standard) e IDEA(International Data Encryption Algorithm).

Blowfish é um algoritmo de cifragem simétrica de blocos que pode utilizar chaves de 32 a 448 bits, no caso do ssh é utilizado 128 bits. Este algorimto é utilizado para encriptar o conteúdo que trafega pela rede entre as máquinas que comunicam via ssh.

```
Assinatura Digital + \
Chaves assimétricas + \
Criptografia da seção = \
                SSHCrypto-Security
```

Para exemplificar o processo de assinatura digital com autenticação comchaves assimétricas do SSH e criptografia da seção, é ilustrado todo este processo em uma figura. Utilizaremos 2 personagens hipotéticos para ilustrar o processo de autenticação e verificação da procedência do SSH e comunicação.

Na figura, em 1, José D. conecta-se na máquina de Luís Inácio via ssh e recebe a chave pública(RSA). Em 2, podemos verificar que a máquina de José D. gera um número aleatório com hash que será criptografado junto com seu login e senha e enviado para a máquina de Luís Inácio.

A máquina de Luís Inácio descriptografa o conteúdo com a chave privada(RSA) e verifica consistência. Caso positivo Luís Inácio verifica que a procedência está correta, pois foi gerada por José D. e que não foi adulterada durante a transmissão.

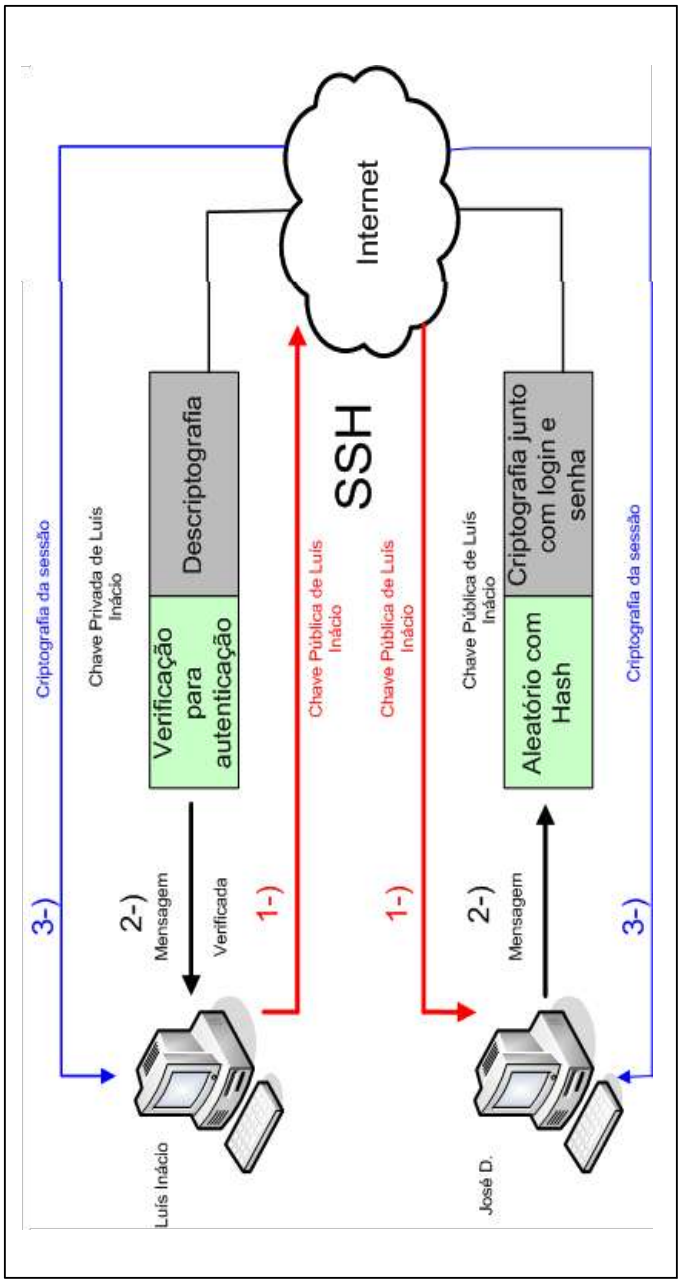
Caso contrário 3 possibilidades podem ter ocorrido:

1. Mensagem corrompida;

2. Não foi assinado por Luís Inácio;

3. Foi adulterada no percurso.

Após o processo 2 da figura ter ocorrido com sucesso, a conexão pode ser estabelecida com garantia de privacidade. Na figura, verificamos em 3, que uma vez estabelecida a conexão com confiabilidade em duas vias, a seção é criptografada, com algoritmo Blowfish, 3DES, etc...



Verifique você mesmo.

Para verificar passo a passo a conexão em máquinas remotas com SSH e o chamado de seus algoritmos de criptografia, utilize o parâmetro -v (verbose):

```
# ssh -v dfiori@smac.lcad.icmc.usp.br
OpenSSH_3.9p1, OpenSSL 0.9.7d 17 Mar 2004
debug1: Reading configuration data \
                  /etc/ssh/ssh_config
debug1: Connecting to \
            smac.lcad.icmc.usp.br \
            [143.107.232.169] port 22.
debug1: Connection established.
debug1: identity file \
      /home/dfiori/.ssh/identity type -1
debug1: identity file \
      /home/dfiori/.ssh/id_rsa type -1
debug1: identity file \
      /home/dfiori/.ssh/id_dsa type -1
debug1: Remote protocol version 2.0,\
      remote software version \
      OpenSSH_3.7.1p2
debug1: match: OpenSSH_3.7.1p2 \
      pat OpenSSH*
debug1: Enabling compatibility mode \
      for protocol 2.0
debug1: Authentications that can \
      continue: publickey,password,\
      keyboard-interactive
debug1: Local version string \
      SSH-2.0-OpenSSH_3.9p1
debug1: SSH2_MSG_KEXINIT sent debug1: \
      SSH2_MSG_KEXINIT received
debug1: kex: server->client \
      aes128-cbc hmac-md5 none \
      debug1: kex: client->server \
      aes128-cbc hmac-md5 none
debug1: SSH2_MSG_KEX_DH_GEX_REQUEST\
      (1024<1024<8192) sent
debug1: expecting \
      SSH2_MSG_KEX_DH_GEX_GROUP
debug1: SSH2_MSG_KEX_DH_GEX_INIT sent
debug1: expecting \
      SSH2_MSG_KEX_DH_GEX_REPLY
debug1: Host 'smac.lcad.icmc.usp.br' \
      is known and matches the RSA \
      host key.
debug1: Found key in \
      /home/dfiori/.ssh/known_hosts:2
debug1: ssh_rsa_verify: signature correct
debug1: SSH2_MSG_NEWKEYS sent
```

E várias outras mensagens semelhantes, até que finalmente:

```
debug1: Next authentication \
      method: password
dfiori@smac.lcad.icmc.usp.br's password:
```