

BSD MINI GUIDA HOW TO

Le pagine di questa breve guida provengono dal sito: www.paolodistefano.name
www.distefanopaolo.it

The real question is not: "*Is FreeBSD good enough for you?*"

The real question is: "*Are you good enough for FreeBSD?*"

SOMMARIO:

[Premessa;](#)

[Configurazione hardware;](#)

[Partizioni e devices in FreeBSD;](#)

[Procedure post-installazione: ports e filosofia d'utilizzo;](#)

[Installazione server grafico \(Xorg\) e desktop manager \(KDE, Gnome, Xfce\)+ \[login grafico: KDM;\]\(#\) \[Compiz ed effetti 3D\]\(#\)](#)

[Il file /etc/rc.conf; il file /boot/loader.conf;](#)

[Montare e vedere le partizioni Windows e/o Linux e viceversa;](#)

[Firewall metodo IPWF;](#)

[Firewall metodo OpenBSD packet-filter;](#)

[Gestione e interventi vari:](#)

[Stampanti USB;](#)

[Scanner USB;](#)

[Schede rete;](#)

[Altri dispositivi USB \(Pendrive, HD USB, Smartphone/Tablet\)](#)

[Installazione Apache+MySQL+PHP+PhpMyAdmin](#)

[Gestione binari Linux;](#)

[Samba \(con particolare riferimento a MacOSX\)](#)

[Caratteri true type MS;](#)

[Tastiera italiana nella console;](#)

[Lista dei dispositivi USB;](#)

[Verifica della memoria RAM gestita;](#)

[Plugin flashplayer;](#)

[Plugin multimediali in Firefox;](#)

[Riproduzione CD audio;](#)

[Supporto Java;](#)

[VirtualBox;](#)

[Collegamento DLNA/UpPnp con dispositivi multimediali \(Playstation, TV, ecc.\);](#)

[Vedere le dirette RAI con FreeBSD;](#)

[Usare la rete Tor con FreeBSD;](#)

[Come accedere al sistema da console per riparare errori;](#)

[OCR su *BSD;](#)

[Upgrade ad una versione successiva](#)

[Compilazione di un nuovo kernel](#)

PREMESSA

E' possibile un uso desktop dei *BSD? Sommarariamente si può dire:

OpenBSD e **NetBSD** sono *BSD specifici per uso **server** e non possono essere facilmente gestiti per un uso desktop, poi tutto è possibile;

FreeBSD è un *BSD più abbordabile che permette, con un certo impegno, di avere un desktop funzionante. La presente guida è basata sulla **versione 15.0**

Per quelli che vogliono un sistema al boot già con ambiente grafico (MATE) e parecchie cose configurate esiste un'altro *BSD uso desktop più recente (anche Live) molto consigliato e sempre basato su FreeBSD: [GhostBSD](#).

La scelta d'installare GhostBSD o FreeBSD è alquanto soggettiva. GhostBSD sicuramente offre una semplicità di installazione e anche uso in ambiente Desktop. Il principale vantaggio è che si installa con interfaccia grafico e all'avvio parte già con un ambiente grafico (MATE). Inoltre alcune operazioni sono facilitate come: gestione degli aggiornamenti, riconoscimento partizioni Windows (anche in scrittura con fuse-ikl) e Linux, configurazione stampanti e scanner, ecc. Sono tutte cose non di poco conto ma in seguito sarà necessario però apprendere come gestirlo e quindi la seguente guida è utile anche per GHOSTBSD. Ultimamente GHOSTBSD usa come filesystem solo ZFS e questo può creare problemi nell'installazione. **Importante:** la differenza principale tra i due sistemi è nella **gestione dei servizi e nell'installazione degli applicativi**. Per i servizi GhostBSD usa nettamente **openRC** e quindi tutto ciò che attiene alla gestione dei servizi va fatta con comandi da terminale un po' diversi. Con GhostBSD si usa il comando da amministratore `rc-service <nome servizio> (opzioni: start, stop, status)`. Ad

esempio con `rc-service cupsd stop` si ferma il server di stampa Cups mentre su FreeBSD si usa generalmente `service cupsd stop`. Qui un [Video](#) sulle differenze (in inglese). Per gli applicativi si consiglia di usare con GhostBSD i due tool grafici a disposizione e relativi ai pacchetti e al loro aggiornamento.

Un'alternativa è [DragonFly BSD](#)

. Si segnala che **TrueOS (ex PC-BSD)** un *BSD pensato anch'esso per un uso desktop, non è più supportato.

Lato utenza, si può dire che un **utente Linux** (specie se ha avuto trascorsi su distribuzioni come Slackware, Archlinux o Gentoo) può affrontare con una certa sicurezza l'installazione e la gestione di un *BSD. Un **utente Windows** può tentare d'installare FreeBSD o meglio GHOSTBSD con l'accortezza di saper almeno come ritornare a ripristinare il MBR, nel caso qualcosa andasse storto.

Lo stesso dicasi per un **utente Mac**, ammesso e non concesso che voglia uscire dal guscio (peraltro ottimo) del MacOSX che è poi, a grandi linee, proprio un *BSD con in più Aqua. Detto in soldoni, una FreeBSD richiede una disciplina che in Linux ha forse paragone solo nella distribuzione Gentoo/ArchLinux. Con la differenza, questa è una mia personale opinione, che le guide Gentoo sono insuperabili per chiarezza anche nell'affrontare temi difficili. Lo stesso non si può dire sempre per le guide o i tutorial in rete su FreeBSD che risultano spesso non aggiornate.

Questa mini-guida prende ad esempio FreeBSD e si rivolge ad un'utenza di livello medio. Installazioni su macchine virtuali (VirtualBox, ecc.) non forniranno un test completo ed autosufficiente e non potranno l'utente di fronte alla prova di settare e configurare un ambiente desktop completo dall'inizio.

E' d'obbligo in ogni caso dare un'occhiata prima alla [guida ufficiale di FreeBSD](#) e specificatamente alla parte riguardante l'allocazione delle partizioni e l'installazione. In questa miniguia mi limito a suggerire alcune cose.

Infine, oltre a questa mia guida, consiglio la lettura integrativa di quest'altro [howto \(in inglese\) ufficiale](#) e dedicato ad un'installazione in ambiente desktop.

CONFIGURAZIONE HARDWARE

In questa guida consiglio l'installazione di un *BSD soprattutto a chi ha schede grafiche **Nvidia**, per un pieno **supporto 3D**. Ovviamente, dato che il supporto driver è un tantino carente pure rispetto a Linux, evitare di installare il sistema operativo su macchine che abbiano dispositivi di ultimissima generazione. Le **schede wireless** comuni presenti nei laptop più usati vengono riconosciute quasi tutte, ormai.

Per il resto dovrebbe essere normale amministrazione da affrontare caso per caso e la guida

di seguito riporta come risolvere alcuni problemi con l'hardware.

Un **avvertimento** magari non necessario ma che potrebbe essere utile. Se avete connesso un HD esterno USB c'è il rischio, se siete sfortunati, che **l'installazione di FreeBSD si blocchi appena partita**. Un'esperienza che castra fin dall'inizio e potrebbe far rinunciare all'installazione. In tal caso è sufficiente spengere l'HD USB esterno e rilanciare l'installazione. Per **risolvere il problema in via definitiva** è probabile che dobbiate disabilitare nel BIOS del computer il **Legacy USB support**. Andate con il comando previsto dal vostro computer (tasto DEL, ecc.) nel BIOS ed in una sezione dove avete la configurazione per i devices USB **disabilitate l'opzione Legacy USB support**. Dovrebbe funzionare.

INSTALLAZIONE: PARTIZIONI E DEVICES

Cosa importante da ricordare: un *BSD vuole una **partizione primaria**, meglio se installato nella prima partizione in sequenza di un HD. Comunque la regola della prima partizione può anche essere derogata senza particolari problemi ma non quella della partizione primaria. Scordatevi quindi le partizioni estese. ***Inoltre è fortemente raccomandato che il disco su cui installerete sia in formato GPT e non MBR.***

Breve cenno sul modo con cui i *BSD definiscono i dispositivi. Rivolgendomi a chi viene da Linux, quello che lì è un /dev/hda (o /dev/sda/) diviene in OpenBSD /dev/wd0 ed in FreeBSD **/dev/ad0** oppure **/dev/ada0**. Anche le partizioni sono descritte in modo diverso. Un device /dev/sda1 diviene un **/dev/ad0s1** oppure **/dev/ada0s1** in **FreeBSD** se il partizionamento è **MBR**.

Se il sistema di partizionamento è **GPT** i devices saranno invece riconosciuti come nel seguente esempio; **/dev/ada0p1** e così via. Nella guida presente sono riportati i parametri MBR e quindi vanno cambiati in caso di partizionamento GPT (in pratica 's1' diviene 'p1' ecc...) I dispositivi USB esterni SCSI (HD esterni) sono chiamati in FreeBSD **/dev/da0** e le loro partizioni **/dev/da0s1**, ecc. I CD e DVD sono **/dev/cd0**, mentre Scanner USB e stampanti USB sono rispettivamente **/dev/ulpt0** e **/dev/uscanner0**. Per gli scanner però leggetevi la sezione apposita della guida ufficiale e comunque attenzione perchè ormai FreeBSD li denomina **/dev/ugenX.X** (dove X=numero). Per le schede di rete, ethernet è **ale0**.

Con **GhostBSD** l'installazione è fatta con un'interfaccia grafica che ha somiglianza con le installazioni delle distro Linux, mentre con **FreeBSD** l'interfaccia è minimalistica. Inoltre con GhostBSD al riavvio sarete in un'ambiente DE conosciuto agli utenti Linux: MATE.

Dopo l'installazione ed il riavvio passate direttamente alla fase descritta qui sotto nella sezione **ports**. Una volta installati i ports, installarete, seguendo la guida ufficiale FreeBSD server grafico e Desktop manager e altre cose essenziali. Questo metodo è già usuale per chi viene da installazioni linuxiane toste (ArchLinux e Gentoo). Però, per esperienza e lo dico soprattutto a quelli che vengono da Linux, FreeBSD risulta più ostico e meno diretto

nell'aggiunta dell'ambiente grafico ed altro. Tuttavia è possibile più facilmente installare il sistema grafico ed un Desktop Manager senza compilazione con i pacchetti attraverso il comando `pkg` di `pkgng`.

Altra cosa: se non siete proprio dei novellini cercate sempre di agire manualmente all'atto dell'installazione nella **scelta delle partizioni**, per attenersi quanto più rigorosamente alla scelta delle subpartizioni consigliate sul manuale ufficiale (`/`, `/var/`, `/tmp/`, `/usr` e opzionalmente `/home`). Comunque si può anche far fare la partizione in automatico, sempre controllando il device scelto. La quantità di spazio dedicato alle varie partizioni sarà a vostro piacimento, tenendo presente che quella più grossa (e di parecchio) va riservata a `/usr`. Attenzione anche a quella di `/root`. Dategli sempre uno spazio ragionevole (1,5/2 giga sono congrui).

Noterete che c'è la tendenza ad esagerare nello spazio **SWAP**. Ora, in presenza di memorie RAM più capienti si può tranquillamente diminuire di parecchio la SWAP.

Tenete presente che, saranno i ports installati, saranno altre cose, comunque un *BSD uso Desktop non è mai tanto leggero (in questo assomiglia alla Gentoo). Con gli HD di oggi non ci saranno problemi ma tenetevi sempre sui **20 giga minimi per tutto il sistema**.

Altra cosa da ricordare con FreeBSD: potete scegliere come filesystem **ZFS** (di default su GHOSTBSD) oppure **UFS**. Tenete però presente che per usare ZFS bisogna avere una macchina a 64bit e almeno 4 giga di memoria RAM. Inoltre, con ZFS, potreste avere più problemi di riconoscimento delle partizioni *BSD da ambiente Linux. Conviene ZFS? E' sicuramente un filesystem potente ma dalla gestione più complessa. Ha il problema poi di dover essere gestito, in presenza di altri sistemi operativi, con Grub 2 e non sempre è facile settare il boot loader. In fondo, se proprio siete incuriositi da ZFS, potete sempre in un secondo momento settare un pool ZFS nell'ambiente *BSD e divertirvi a vedere le potenzialità di questo file-system.

Per far partire *BSD con filesystem UFS con altri sistemi operativi presenti sul computer è necessario un **boot loader**. Se venite da Windows e state provando il *BSD preparatevi a saper tornare indietro ripristinando il MBR originario, nel caso qualcosa non funzionasse. Se avete già un boot loader, magari installato con un'altra distro Linux, tenete quello. Poi, per lanciare FreeBSD con filesystem UFS sarà sufficiente inserire, se usate il **Grub2** dovete creare se non c'è già un file **40_custom** nella directory `/etc/grub.d` ed inserire le seguenti righe:

```
menuentry 'FreeBSD' {
insmod ufs2
set root=(hdx,x)
kfreebsd /boot/loader
}
```

Con sistemi a partizionamento GPT (raccomandato) mettere queste righe nel Grub2 sempre nel file 40_custom:

```
menuentry 'FreeBSD' {  
  insmod part_gpt  
  insmod ufs2  
  set root=(hdx,gptx)  
  kfreebsd /boot/loader  
}
```

In entrambi i casi le 'x' vanno sostituite con i numeri della partizione in cui si trova FreeBSD

Con sistemi UEFI provare:

```
menuentry "FreeBSD" {  
  insmod part_gpt  
  insmod fat  
  set root=(hdx,gptx)  
  chainloader /efi/freebsd/loader.efi  
}
```

Con filesystem ZFS le cose si complicano. Un workaround è installare il Grub non su MBR ma sulla partizione dove risiede il *BSD. Attenzione poi che ZFS tenderà ad occupare tutto lo spazio sul device ed è difficile installarlo su una partizione.

POST-INSTALLAZIONE

Ognuno ha le sue procedure dopo che vede girare al riavvio il sistema. Qui segnalo alcune cose che bisogna fare per avere il sistema il più funzionante possibile.

Può sembrare strano ma la prima cosa da fare è impratichirsi subito con il modo usato da FreeBSD per installare le applicazioni e aggiornare il sistema (in questo ArchLinux e soprattutto Gentoo gli assomigliano). Specie se installate FreeBSD, alla fine dell'installazione avrete pochi applicativi pronti all'uso e dovrete aggiungere molti programmi rispetto ad una distro Linux più user-friendly. Ecco le principali cose da fare.

Installazione dei programmi, ports e filosofia d'utilizzo

Le **applicazioni** sono gestite in due modi. Uno è simile al sistema portage di Gentoo, che infatti lo ha ripreso in parte. Cioè si scaricano i sorgenti e si **compilano**. Avete sentito bene. Si compilano. Questo significa sistema ottimizzato ma anche **tempi lunghi**, specie se avete computer non proprio schegge. Questo metodo fa capo ai cosiddetti **ports**.

Poi ci sono i **pacchetti binari** che si autoinstallano senza ricorrere alla compilazione ed usano

un comando simile all'`apt-get` di Debian, ovvero **pkg** (che ha ormai sostituito il precedente e non più supportato **pkg_add**) e relative opzioni.

In genere sarebbe consigliato di **ricorrere quanto più possibile alla compilazione**. Tuttavia questa scelta comporta spesso tempi biblici, anche su computer potenti. Diciamo che, con l'adozione del metodo a pacchetti pkgng, è ormai accettabile ricorrere ai pacchetti, velocizzando molto il tutto. **Attenzione:** una via ibrida d'installazione e gestione, fatta sia con i pacchetti binari che con i ports può comportare alla lunga problemi di stabilità al sistema e problemi soprattutto con gli aggiornamenti.

Il meccanismo gestionale degli applicativi da installare tramite compilazione e ports è considerato molto potente dai fans di FreeBSD. Senza dubbio sarà efficace ma ad un utente poco esperto risulta piuttosto macchinoso. Poi, questa è una mia considerazione, l'accavallarsi dei metodi (si possono usare varie procedure per gli aggiornamenti) creano alquanto confusione, disorientando l'utente.

E' praticamente impossibile sintetizzare qui tutte le procedure possibili. Qui indico la via che ritengo più facile.

Tutti i comandi vanno dati da console e da amministratore.

Attenzione: **GhostBSD** usa dei propri ports, seguire le istruzioni [QUI](#)

Gestione attraverso sorgenti e compilazione

La gestione attraverso **portsnap** dei ports (sorgenti) è considerata deprecata e sarà dismessa

Al suo posto usare **SVN** come è descritto [QUI](#)

Riassumendo dare da amministratore il comando **pkg install subversion** e poi **svn checkout https://svn.FreeBSD.org/ports/head /usr/ports**

Dopo un po' di tempo avrete installati i sorgenti da installare sotto la directory `/usr/ports`

Dovete essere consapevoli che qualsiasi aggiornamento del sistema potrebbe comportare tempi lunghi, dato che il sistema dovrà ricompilare parecchie cose.

Con **svn update /usr/ports** aggiornate i ports (con un tempo di esecuzione inferiore rispetto all'installazione)

Per **installazione/aggiornamento del software** usare il comando **portmaster**.

Una volta installato con un comodo **pkg install portmaster**, questo nuovo tool permette a grandi linee:

con **portmaster -L** di fare un elenco di tutto ciò che è installato e relativi aggiornamenti

disponibili;

con **portmaster -a** l'aggiornamento di tutto l'installato (aggiungendo anche il suffisso "f" avremo un comodo rebuilding dell'installato nel caso qualcosa andasse storto);

con **portmaster <nome pacchetto>** installeremo un nuovo applicativo (il nome del pacchetto sarà comprensivo della directory come nel caso di portmanager);

con **portmaster -check-depends** fate un check di tutte le dipendenze.

Per renderlo compatibile con **pkgng** (il gestionale per i pacchetti) vedere la prossima sezione sul comando pkg e pkgng. Si segnala che il comando va usato comprensivo del percorso /usr/ports/ dove si trova il programma per installarlo (ad es.: portmaster /usr/ports/editors/libreoffice).

Per installare tramite compilazione una volta aggiornati i ports, cercare il software con il comando whereis <nome programma> e saprete in quale subdirectory di /usr/ports/ c'è il vostro applicativo. Andate dentro la directory e date da root **make install clean** (potrebbe essere usato in alcuni casi il comando **make config** prima ma solo se sono necessarie le configurazioni).

Questo compilerà il tutto, chiedendo magari quali opzioni di compilazione volete. In questi casi se non sapete esattamente cosa fare lasciate le scelte di default. In alternativa usare il più comodo comando **portmaster** seguito dalla subdirectory con l'applicativo da installare.

Gestione pacchetti binari

Con FreeBSD e derivate è possibile usare in modo stabile il metodo **pkgng** basato sul comando **pkg**, che ha sostituito pkg_add. Usando **pkg** la gestione di un sistema FreeBSD si è di molto semplificata per l'installazione e l'aggiornamento del software installato, per cui è **vivamente raccomandato ai meno esperti**.

Non solo, a partire da FreeBSD 15 conviene scegliere il cosiddetto metodo PkgBase (cioè l'intera gestione del sistema è gestita da pkg) dato che con la versione 16 è previsto l'abbandono del metodo freebsd-update.

Per avere compatibilità tra pkgng e portupgrade/portmaster bisogna aggiungere una riga al file /etc/make.conf: WITH_PKGNG=yes. Inoltre, se non vi funziona ancora con portmaster, disinstallate portmaster e reinstallatelo di nuovo.

Inoltre, può capitare che, per evitare conflitti nella ricerca del repository, di dover controllare che nelle rispettive directory utente e root, nel file nascosto di configurazione relativo ai terminali (se usate bash=.bashrc mentre con cmd=.chsrc), sia commentata con un # davanti la riga: setenv PACKAGESITE <path al repository>.

I comandi più in uso con **pkg** sono **pkg search** <nome pacchetto> (cerca i pacchetti anche con riferimento parziale al nome). Mentre **pkg install** <nome pacchetto> serve sia ad installare che ad aggiornare il singolo pacchetto con tutte le dipendenze. Poi: pkg delete

<nome pacchetto> (rimuove il pacchetto). In caso di dipendenze che bloccano la rimozione bisogna forzare con `pkg delete -f <nome_pacchetto>`. Con **pkg update** (aggiorna l'elenco pacchetti dai repositories); `pkg upgrade` aggiorna tutti i pacchetti alla versione più recente. Attenzione: può capitare che durante l'upgrade `pkg` segnali di dover rimuovere alcuni pacchetti. Qualche volta questo può essere rischioso. Eventualmente, si può dare allora il comando `pkg upgrade -f` che reinstallerà i programmi per risolvere i conflitti. Utile anche il comando **pkg lock** <nome pacchetto> che esclude dell'aggiornamento un singolo pacchetto (es.: `pkg lock ffmpeg`). Per sbloccarlo il comando è: `pkg unlock <nome pacchetto>`.

Con il comando `pkg info` si visualizza l'elenco dei pacchetti installati); `pkg info -f <nome pacchetto>` fornisce invece informazioni sul singolo pacchetto. Infine con `pkg clean` fate una pulizia dei vecchi pacchetti che avete aggiornato e che non vengono rimossi, con aumento considerevole della memoria occupata. Con il comando `pkg autoremove` disinstallate i pacchetti non più necessari per le dipendenze (usare con accortezza). Affidarsi a `pkg`, ormai consolidato, è consigliato per i meno esperti e per chi non vuole perdere tempo con la compilazione. In effetti, usando `pkg` dopo l'installazione del sistema base, in poco tempo si avrà una FreeBSD uso desktop perfettamente funzionante.

Infine, capita talvolta che dopo il comando `pkg upgrade` risultano messaggi del tipo: “`pkg: Cannot solve problem using SAT solver`”. In tal caso provate a dare il comando **pkg check -Ba** e aspettate la fine della procedura (ci mette un pò di tempo). Alla fine ridate `pkg upgrade` e probabilmente potete procedere con lo stesso.

Per quelli che non amano particolarmente la riga di comando, incredibile a dirsi, FreeBSD ha a disposizione anche una GUI per il package manager e si chiama [octopkg](#). Si installa con 'pkg install octopkg' e poi il suo uso è intuitivo. Si cerca il pacchetto e sono elencati quelli installati (in verde) o quelli da installare, ecc.

Infine può capitare che sia necessario installare un pacchetto binario con una particolare configurazione diversa da quella di default del pacchetto scaricato, In tal caso si usano i ports in `/usr/ports` cercando il pacchetto e dando il solito 'make config' per personalizzare la configurazione e poi dare **make package**. Verrà creato un pacchetto con estensione **.txz** nella sottodirectory `/usr/ports/.../work/pkg`. Controllare se c'è e dare il comando 'pkg install `/usr/ports/.../<nome applicativo>/work/pkg/<nome pacchetto>`'.

ATTENZIONE: con **GhostBSD** è raccomandato usare gli applicativi grafici 'Software center' e per gli upgrade la 'Update station'.

Attenzione: è sconsigliato sovrapporre troppo il software installato con pacchetti binari con altro compilato. Le due procedure, alla lunga, se usate insieme possono creare confusione e problemi di dipendenze non facilmente risolvibili. In sostanza usare o l'uno o l'altro dei metodi (compilazione o pacchetti binari) ed evitare di mischiare installazioni fatte con i binari e con la compilazione.

Il sistema di aggiornamento del software nei *BSD sarà pure rigoroso ma risulta francamente macchinoso e addirittura impossibile (dati i tempi necessari) con la compilazione se il sistema contiene migliaia di pacchetti. Insomma, a parte ragioni di sicurezza, nei *BSD è bene essere un po' parsimoniosi sugli aggiornamenti. Se il sistema va bene e gli applicativi girano, non vi fate prendere troppo la mano ad aggiornare in continuazione. Un buon metodo è quello di controllare periodicamente con il comando **portmaster -L** quali ports siano da aggiornare e regolarsi manualmente e secondo esigenze personali. Con **portmaster** è utile per fare un aggiornamento di tutto dare un comando del genere: **portmaster -dBa --delete-packages** (le opzioni evitano procedure che allungano i tempi).

Infine, per le **patches della sicurezza** è opportuno dare periodicamente **freebsd-update fetch** e poi **freebsd-update install** seguito da un reboot del sistema. Se qualcosa è andato storto, con **freebsd-update rollback** si ritorna indietro. **Per quest'ultimo aspetto [vedi anche la guida ufficiale.](#)**

ATTENZIONE: questo comando non funziona più a partire dalla versione 15 se è in uso il metodo PkgBase da errore "freebsd-update is incompatible...". Poichè dalla versione 16 è probabile l'uso del solo metodo PkgBase lasciare al comando 'pkg update' tutta la gestione degli aggiornamenti

Infine un accenno al non indifferente spazio che occupano le directory dedicate alla gestione software. Se si rilevassero **problemi di spazio sull'HD** provate a pulire e liberare spazio con comandi come **portmaster --clean-distfiles** oppure **pkg clean**

Adesso che sappiamo più o meno come installare e aggiornare, vediamo i passi successivi da fare.

Prima cosa: **il server grafico.** Ahia! Con FreeBSD vi dovete imbarcare nel dover gestire direttamente l'installazione del server grafico e dell'ambiente grafico. Del resto, chi viene da Archlinux è già pronto a questo. Quanto segue interessa quindi **FreeBSD.** Con **GhostBSD** l'ambiente grafico di default MATE è già installato, anche se è probabile che sarà necessario comunque installare i driver della scheda grafica (ad esempio Nvidia) successivamente.

Installare server grafico e ambiente desktop su FreeBSD. Si da per scontato che avete già installato **Xorg** (pkg install xorg).

Una volta installato Xorg, bisogna configurarlo a seconda della vostra scheda video, soprattutto per l'uso del rendering. Un comando **Xorg -configure** da console vi prepara un file di esempio da rivedere e portare sotto /etc/X11 con il nome di xorg.conf. Tutto questo lo farete da terminale.

Se avete una **scheda nvidia** installerete con il pacchetto binario o dai ports i driver delle

schede. Attenzione a sapere quale scheda avete, perché esistono varie versioni dei driver. Installate anche **nvidia-xconfig** e date poi un nvidia-xconfig per creare in automatico il file di configurazione xorg.conf per le schede Nvidia. Sarà poi necessario caricare il modulo agendo in rc.conf e in bootloader.conf. In genere è sufficiente inserire in rc.conf la riga kld_list="nvidia" o aggiungere il nome nvidia agli altri moduli presenti nella riga. Nel caso la scheda NVidia sia un po' vecchiotta e non parte il driver con l'errore di ABI non supportato, il trucco è andare nel file di configurazione di nvidia e aggiungere queste righe:

```
Section "ServerFlags"
Option "IgnoreABI" "true"
EndSection
```

Altri piccole dritte: fare attenzione al riconoscimento del mouse che in BSD va nel /dev/sysmouse e che "Input Protocol" in genere accetta l'opzione "auto".

Una volta assicurati con comando da console **startx** che il server grafico parta, avrete il problema di far girare **KDE** (consigliato), **Gnome o Xfce**. La cosa più facile leggere la guida molto chiara, [questa](#). Una volta installato il DE per farlo partire al boot bisogna usare un login manager. Considerare che KDE5 preferisce come login manager **sddm** da abilitare nel file /etc/rc.conf con la riga **sddm_enable="YES"**. Funziona anche **gdm** (login manager di Gnome3) e in tal caso nella riga di rc.conf aggiungere **gdm_enable="YES"**. Tuttavia non sempre i login manager funzionano a dovere. Alternativa valida è affidarsi al file **.xinitrc** nella propria directory home. Anche qui seguire la guida segnalata prima. Comunque in sintesi bisogna dopo l'avvio del sistema inserire il proprio nome utente e password e poi dare 'startx', dopo aver inserito una di queste righe nel file citato .xinitrc (a seconda del DE scelto):
exec dbus-launch --exit-with-x11 ck-launch-session mate-session (per Mate)
exec dbus-launch --exit-with-x11 ck-launch-session startplasma-x11 (per KDE5)
exec gnome-session (per GNOME)
. /usr/local/etc/xdg/xfce4/xinitrc (per XFCE4)

Con **GNOME** potrebbe capitare **che il DE non cambia la lingua in italiano**, nonostante il settaggio consigliato nelle guide ufficiali. In tal caso provare a modificare la lingua desiderata nel file /usr/local/etc/gdm/locale.conf così:

```
LANG="it_IT.UTF-8"
LC_CTYPE="it_IT.UTF-8"
LC_MESSAGES="it_IT.UTF-8"
```

Con **MATE** invece provate a cambiare il file .bashrc (se usate la bash shell) nella vostra home directory inserendo:

```
export LANG=it_IT.UTF-8
```

Editare e/o modificare il file `/etc/rc.conf`

Sul file `rc.conf`. E' molto importante nei *BSD (chi conosce ArchLinux lo sa) e si trova sotto la directory `/etc`. Poiché, nonostante la sua importanza, si trova scarso materiale in giro nella rete per configurarlo, a titolo d'esempio se ne fornisce uno:

```
hostname="localhost"
keymap="it.iso.kbd"
ifconfig_ale0="DHCP"
ifconfig_ale0_ipv6="inet6 accept_rtadv"
sshd_enable="YES"
kld_list="nvidia fusefs linux linux64"
moused_enable="YES"
ntpd_enable="YES"
ntpd_sync_on_start="YES"
nptd_config="/etc/ntp.conf"
powerd_enable="YES"
powerd_flags="-a hiadaptive"
# Set dumpdev to "AUTO" to enable crash dumps, "NO" to disable
dumpdev="NO"
dbus_enable="YES"
hald_enable="YES"
avahi_daemon_enable="YES"
avahi_dnsconfd_enable="YES"
sddm_enable="YES"
cupsd_enable="YES"
devfs_system_ruleset="system"
apache24_enable="YES"
mysql_enable="YES"
ftpd_enable="YES"
#enable ftp-proxy for pf firewall rules
ftpproxy_enable="YES"
linux_enable="YES"
pf_rules="/etc/pf.conf"
pf_enable="YES"
pf_flags=""
pflog_enable="YES"
pflog_logfile="/var/log/pflog"
pflog_flags=""
privoxy_enable="YES"
```

```
tor_enable="YES"
vboxnet_enable="YES"
# clear the /tmp directory at boot
clear_tmp_enable="YES"
```

Come si vede è una lunga sfilza di comandi YES/NO che attivano servizi e quant'altro all'avvio. Una volta che ci si impratichisce nell'uso è molto comodo. La sintassi è abbastanza elementare e la riga d'aggiungere viene in genere suggerita dalle guide o dagli articoli in giro per la rete. In questo caso, basta ricopiarli uguali ed inserirli. Qui si evidenzia la presenza, aggiunta in un secondo momento, dell'abilitazione all'avvio del server Apache e di Mysql. Da segnalare anche la settatura dei parametri della rete che, nell'esempio riportato, si riferiscono all'assegnazione al computer in uso di un indirizzo IP statico; per un indirizzo IP dinamico la riga è: **ifconfig_ale0="DHCP"** e va cancellata (o commentata) la riga con **defaultrouter="..."** . Utile segnalare anche il linux_enable settato a YES che serve a utilizzare i binari dei programmi Linux.

Comunque, anche in questo caso, installando **GHOSTBSD** è quasi tutto pronto (tranne Apache e Mysql), mentre con FreeBSD dovrete aggiungere manualmente parecchie cose ed è quindi utile fare riferimento a questo esempio.

Il file /boot/loader.conf

Anche questo file è piuttosto importante perchè carica moduli e quant'altro, necessari e vitali per molti devices, compresa la scheda audio. A parte quando trovate nelle guide l'apposito accenno ad inserirvi alcuni comandi, ritengo utile riportare anche in questo caso un file di esempio adeguatamente configurato:

```
# Kernel Options
linux_load="YES"
nvidia_load="YES"
fuse_load="YES"
hw.ata.atapi_dma=0
snd_hda_load="YES"
accf_http_load="YES"
sem_load="YES"
hw.usb.no_shutdown_wait=1
vboxdrv_load="YES"
# only zfs filesystem
zfs_load="YES"
```

Come si vede, anche in questo caso la sintassi è piuttosto semplice mentre i nomi dei vari moduli rimandano a parecchi supporti necessari in un desktop.

Vedere le partizioni Windows e/o Linux sullo stesso computer

GhostBSD in automatico mostra le partizioni Windows (lettura/scrittura) e Linux (sola lettura). Quindi, a meno di problemi riscontrati, è possibile saltare questa sezione (tranne la parte dedicata a come leggere partizioni FreeBSD sotto Linux).

Per **FreeBSD** le partizioni FAT32 e NTFS non in scrittura dovrebbero essere montate in automatico su KDE (Plasma). Tuttavia, FreeBSD ha strani comportamenti nei permessi non sempre facilmente risolvibili. Ricordate, a tal proposito, che con la riga **vfs.usermount=1** aggiunta al file **/etc/sysctl.conf** darete i permessi all'utente per il montaggio/smontaggio. Non vale però per fusefs-ntfs (ntfs-3g) che qualche volta funziona qualche altra no. Se vedete però che le partizioni FAT32 si comportano in modo strano lato permessi, conviene montarle in **/etc/fstab** con una riga analoga a questo esempio:

```
/dev/ada(x)s(x) /mnt/fat msdosfs rw 0 0
```

Dato che è supportato anche **ntfs-3g** (si trova sotto il pacchetto o ports **fusefs-ntfs**), conviene usarlo per le partizioni NTFS. Se per caso non vedete le partizioni Windows o non ci potete scrivere qui riporto alcuni consigli su ntfs-3g. [Riguarda soprattutto la configurazione in FreeBSD.](#) Siccome il sistema di denominazione dei dispositivi dei *BSD è un po' ostico si può dare un comando da console **ls /dev** che elenca tutti i dispositivi attivi trovati. Utile soprattutto per montare altre partizioni presenti nell'HD, specie se Linux.

Con **KDE Plasma** in genere le cartelle Windows in lettura/scrittura sono montate in automatico, sempre dopo aver installato con **pkg install fusefs-ntfs** e il [pacchetto essenziale automount](#) con **pkg install automount**, poi caricare il modulo con **kldoad fusefs** (eventualmente controllare con **kldstat** se è già caricato). Se dà errori provate a riavviare direttamente il computer per far caricare il modulo, dopo aver messo in **/etc/rc.conf** **fusefs_enable="YES"**.

Comunque da console la partizione si monta dando un comando da terminale come **ntfs-3g /dev/ada0s1 /mnt/win**. Inoltre è bene creare un link con questo comando da terminale: **ln -s /usr/local/bin/ntfs-3g /usr/sbin/mount_ntfs-3g**. Altrimenti, agire manualmente creando nella directory **/media** tante directory quante sono le partizioni da montare e dare **chmod 777** alla directory in modo da renderla accessibile in tutto agli utenti. Modificare poi il file **etc/fstab** inserendo una riga così (sostituire le 'X'):

```
/dev/adaXsx /media/windows fuse mountprog=/usr/local/bin/ntfs-3g,rw,late 0 0
```

Dare poi un **mount -a** e vedere se funziona. Se non vedete ancora niente, provate a riavviare il computer.

Un discorso a parte meritano le **partizioni Linux** eventualmente presenti sul vostro Hard disk. Per vedere da *BSD una directory Linux bisogna intervenire sul file **/etc/fstab**. Prima però bisogna installare **fusefs-ikl** (*e sempre installare anche automount come visto prima*). Con questo pacchetto sarà possibile vedere e anche scrivere (quasi sempre) sulle directory di una partizione Linux con filesystem ext4, BTRFS e XFS. Anche in questo caso KDE Plasma in Dolphin le monta in automatico. Se volte farlo manualmente, create le cartelle necessarie in /mnt o /media (comando `mkdir /mnt/<nomecartella>`).

Poi editare il file /etc/fstab scrivendo per ogni partizione Linux da vedere la riga:

```
/dev/adaXsx /media/linux fuse mountprog=/usr/local/bin/lklfuse,type=ext4,rw,allow_other 0 0
```

Anche in questo caso, le X stanno al posto dei numeri che indicano le partizioni. Per sapere quali sono vi potete far aiutare dal comando **ls /dev** che vi farà vedere tutti i dispositivi attivi nel computer.

Qualcuno si chiederà se è possibile fare il contrario e vedere le partizioni *BSD da Linux. Da Linux è possibile vedere partizioni *BSD editando il file /etc/fstab:

```
/dev/[device] /media/[directory] ufs auto,ro,ufstype=ufs2,nodev,nosuid 0 0
```

Però così facendo potreste vedere solo la partizione /root. Infatti le subpartizioni restano nascoste a Linux, almeno che non usiate i moduli particolari (nel kernel) per fargliele vedere. Il Kernel deve essere stato compilato con l'opzione **UFS file system support (read only)** nel menù **Miscellaneous filesystems**. Però, per esperienza, ormai i kernel installati di default dovrebbero avere questa opzione attivata. Sennò dovete ricompilare il kernel, non c'è nulla da fare. Dando il comando da Linux `fdisk /dev/sdx` (x=lettera della vostra partizione dove c'è *BSD), entrate in un dialogo in cui prima inserite l'opzione b) e poi l'opzione p) che vi dovrebbe far vedere tutte le subpartizioni *BSD precedute da una lettera.

Dando il comando `cat /proc/partitions`, dovreste avere la conversione delle lettere nel numero allegato alla partizione sdx (esempio: /dev/sda = partizione *BSD in Linux ---> /dev/sda1 = subpartizione lettera a) *BSD in Linux, /dev/sda5 = subpartizione lettera b) *BSD in Linux, ecc.). A questo punto sarà sufficiente creare tante cartelle in /mnt quante sono le subpartizioni elencate e montarle.

Esempio: se /mnt/bsd1, /mnt/bsd5, /mnt/bsd6 sono le tre cartelle create per le tre subpartizioni di *BSD, allora in /etc/fstab, avremo le righe:

```
/dev/sda1 /mnt/bsd1 ufs auto,ro,ufstype=ufs2,nodev,nosuid 0 0
/dev/sda5 /mnt/bsd5 ufs auto,ro,ufstype=ufs2,nodev,nosuid 0 0
/dev/sda6 /mnt/bsd6 ufs auto,ro,ufstype=ufs2,nodev,nosuid 0 0
```

Possibile problema: se dando il comando **dmesg | grep bsd** invece della giusta stringa che identifica le subpartizioni *BSD (tipo: [23.054946] sda2: <bsd: sda11 sda12 sda13 sda14 sda15, che vi dice che le subpartizioni sono sda11, ecc.) vi esce una stringa con: **sdb1: <bsd:bad subpartition - ignored sdb7>** allora la cosa si complica e di parecchio. Sostanzialmente significa che il vostro Linux non riesce a leggere le subpartizioni. In tal caso la questione si fa delicata e non ho una soluzione, almeno finora, a questo problema, anche dopo aver cercato a lungo nella rete. Forse una soluzione è imbarcarsi in una delicata procedura di ridenominazione delle etichette (label) con tutto quello che ciò comporta. Se qualcuno ha risolto un problema anche analogo, sarebbe gradita una informativa.

Qui va inserito un riferimento al fatto che soprattutto in FreeBSD potrebbe capitare che le partizioni sia su HD che in dispositivi USB si possano montare solo da root. Per poter provare a **montarle come user** bisogna:

- 1) assicurarsi che ci sia un file /etc/devfs.rules e che ci sia sotto all'intestazione [system=10] una riga tipo: add path 'da*' mode 0660 group operator (dispositivi USB) e una riga con add path 'ada*' identica per i dispositivi HD interni
- 2) l'user deve essere aggiunto al gruppo operator
- 3) aggiungere al file rc.conf una riga con devfs_system_ruleset='system'
- 4) inserire una riga al file /boot/sysctl.conf con vfs.usermount=1
- 5) dare comando sysctl vfs.usermount=1 per rendere effettivo il tutto subito e non al riavvio

Infine: è possibile **vedere partizioni *BSD da Windows?**

Se volete provare date un'occhiata a questo progetto, anche se sembra un po' vecchiotto e testato fino a Windows XP: <http://ffsdrv.sourceforge.net/>

Infine si ricorda che l'accesso a una o più directory tra computer in rete con sistemi operativi diversi è possibile via **Samba** o **ssh** con Windows, Linux e Mac e, in Linux, anche via **NFS (net file system)**.

Firewall

Potete scegliere nei *BSD tra due tipologie di **firewall**, tenuto conto che la sicurezza è ovviamente uno dei punti di forza di questo sistema operativo.

Metodo IPWF (usato di default da **GhostBSD**)

L'attivazione del firewall si ha, come al solito, nel file /etc/rc.conf aggiungendo la riga firewall_enable="YES". Per una semplice configurazione con il metodo seguire [questa guida](#). E' possibile settare già delle protezioni predisposte nel file

/etc/rc.conf/firewall_type="<select>" Le selezioni possibili sono leggibili nel file /etc/rc.firewall (open, client, simple le più usate su un desktop).

Altrimenti si può configurare un file di regole d'accesso (eseguibile e posizionato in genere sotto /etc/ipfw.rules) come spiegato in [questo articolo](#) Inoltre è necessario avere in rc.conf ipfw_enable="YES" e firewall_enable="YES" ed infine firewall_script="/etc/ipfw.rules".

Comunque su **GhostBSD** se si dovessero riscontrare problemi, aprire da amministratore il con le regole ipfw.rules ed inserire:

```
#!/bin/sh
ipfw -q -f flush # Delete all rules
# Set defaults
net="ale0" # out interface (change own inteface)
cmd="ipfw -q add " # build rule prefix
$cmd 00100 allow ip from any to any via lo0
$cmd 00200 deny ip from any to 127.0.0.0/8
$cmd 00300 deny ip from 127.0.0.0/8 to any
$cmd 00400 deny ip from any to ::1
$cmd 00500 deny ip from ::1 to any
$cmd 00600 allow ipv6-icmp from :: to ff02::/16
$cmd 00700 allow ipv6-icmp from fe80::/10 to fe80::/10
$cmd 00800 allow ipv6-icmp from fe80::/10 to ff02::/16
$cmd 00900 allow ipv6-icmp from any to any icmp6types 1
$cmd 01000 allow ipv6-icmp from any to any icmp6types 2,135,136
$cmd 00250 allow tcp from any to any 8895 in via $net
$cmd 00260 allow udp from any to any 1900 in via $net
```

Dove 00250 e 00260 sono 2 regole aggiunte dall'utente sulle porte tcp/8895 e udp/1900 dove \$net corrisponde al nome della vostra interfaccia di rete precedentemnte settata nello script.

Metodo OpenBSD packet-filter (consigliato)

L'altro firewall usa le regole dei packet-filter di **OpenBSD** (mitico!).

PF firewall richiede una certa abilità nell'uso, compensato dall'efficacia e dal fatto che si gestisce in modo molto articolato. Questo significa che in giro troverete esempi spesso diversi tra loro. La guida ufficiale su PF firewall in FreeBSD è [QUI](#). Se siete particolarmente paranoici, conviene includerlo nel kernel e questo significa che dovete compilare e lanciare un nuovo kernel. Per farlo, bisogna seguire la canonica procedura inserendo nel file di configurazione del kernel da compilare queste righe:

```
#PF firewall
```

```
device pf
device pflog
device pfsync
options ALTQ
options ALTQ_CBQ
options ALTQ_RED
options ALTQ_RIO
options ALTQ_HFSC
options ALTQ_PRIQ
options ALTQ_NOPCC
```

La procedura per la compilazione del kernel è [descritta in fondo alla guida](#).

Alla fine della compilazione, prima del reboot necessario, in **/etc/rc.conf** vanno aggiunte le righe:

```
pf_rules="/etc/pf.conf"
pf_enable="YES"
pf_flags=""
pflog_enable="YES"
pflog_logfile="/var/log/pflog"
pflog_flags=""
```

Per quanto riguarda la configurazione si fa creando un file di testo (sempre da amministratore) **/etc/pf.conf** ed editando manualmente le regole da aggiungere.

Qui di seguito un semplice esempio, utile per un uso su una rete domestica, che può essere considerato una base di partenza per prendere confidenza:

```
# PF firewall configuration
interface = "ale0"
local_host = "192.168.1.2"
udp_services = "{ntp}"
tcp_services = "{smtp,http}"
proxy="127.0.0.1" # ftp proxy IP
proxyport="8021" # ftp proxy port
#list for IP addresses blocked
#table <blockedips> persist file "/etc/blocked_ips.conf"
set block-policy return
set skip on lo0
scrub in all
#### NAT and RDR start
nat-anchor "ftp-proxy/*"
```

```
rdr-anchor "ftp-proxy/*"  
# Redirect ftp traffic to proxy  
rdr pass proto tcp from 192.168.1/24 to any port ftp -> $proxy port $proxyport  
# We need to have an anchor for ftp-proxy  
anchor "ftp-proxy/*"  
block all  
# block ip from list  
# block drop in log quick on $interface from <blockedips> to any  
# Enable ICMP for IPv4 IPv6  
pass proto icmp all  
pass proto icmp6 all  
pass out all keep state  
pass in quick on $interface inet proto tcp from any to $local_host port $tcp_services keep  
state  
pass in quick on $interface inet proto udp from any to $local_host port $udp_services keep  
state  
# Rules section  
# DLNA Serviio  
pass in quick on $interface inet proto tcp from 192.168.1/24 to $local_host port 8895 keep  
state  
pass in quick on $interface inet proto udp from 192.168.1/24 to $local_host port 1900 keep  
state  
# CUPS  
pass in quick on $interface inet proto tcp from 192.168.1/24 to $local_host port 631 keep  
state  
pass in quick on $interface inet proto udp from 192.168.1/24 to $local_host port 631 keep  
state  
# TOR  
pass in quick on $interface inet proto tcp from any to $local_host port 8118 keep state  
# SSH  
pass in quick on $interface inet proto tcp from 192.168.1/24 to $local_host port 2224 keep  
state  
# Restreaming RaiNixV3  
pass in quick on $interface inet proto tcp from any to $local_host port 8530 keep state  
# NGINX SERVER (RTMP)  
#pass in quick on $interface inet proto tcp from 192.168.1/24 to $local_host port 8554 keep  
state  
#pass in quick on $interface inet proto tcp from 192.168.1/24 to $local_host port 1935 keep  
state
```

eccetera..... qui un utile [articolo riepilogativo](#).

Per riavviare il firewall da terminale e da root caricando nuovi filtri dare il comando:

pfctl -f /etc/pf.conf

Per avviare solo il Firewall dare da root il comando **pfctl -e**, mentre per disabilitarlo dare **pfctl -d**

Con il comando **sudo pfctl -s all** si verifica lo stato di tutte le regole in azione

Potreste avere un errore al boot del firewall di questo tenore:

```
no IP address found for lo1:network
```

```
/etc/pf.conf:4: could not parse host
```

```
specification
```

```
no IP address found for lo1:network
```

```
/etc/pf.conf:5: could not parse host
```

```
specification
```

In tal caso dovrete dare da terminale, dopo aver stoppato il firewall con il comando **/etc/rc.d/pf stop**, il comando **/usr/local/etc/rc.d/portjail restart**, poi far ripartire il firewall con **/etc/rc.d/pf start** e ricaricare le regole.

Ovviamente, se avete compilato il kernel per includere PF, va dato un reboot per sincerarsi che tutto funzioni bene al riavvio.

Problema con FTP. Attenzione, perché se volete usare **FTP** con il packet filter firewall bisogna fare degli interventi suppletivi. In pratica passare per un server-proxy apposito (ftp-proxy) e poi settare adeguatamente il file pf.conf Per fortuna che il server proxy s'installa in auto con il firewall. Per sapere come fare [seguire questa guida](#).

Domanda: ma non c'è un'interfaccia grafica per il tutto? In teoria, sì. Si tratta di [FWBUILDER](#) in grado di impostare anche firewall PF. L'applicativo è fatto piuttosto bene e si installa facilmente con un pkg install fwbuilder ma non sempre è pienamente compatibile nell'importazione di un file di configurazione pf.conf già fatto e forse alla fine risulta meno complicato usare la configurazione manuale.

GESTIONE E INTERVENTI VARI

Stampanti USB

Con **GhostBSD** è necessario solo installare il pacchetto gutenprint (il caricamento del modulo all'avvio cupsd è già integrato) e poi passare alla configurazione della stampante con browser

all'indirizzo localhost:631

Per FreeBSD, dopo aver installato cups e gutenprint-cups (e hplip per stampanti HP), bisogna settare la stampante **CUPS** tramite il famoso localhost:631 da browser. Durante la scelta della stampante USB se avete problemi e vedete caratteri strani all'atto della stampa, provate a spuntare l'opzione "**USB (no reset)**" e non quella solo "USB". Potrebbe essere necessario per usare CUPS disabilitare **lpd** nel file /etc/rc.conf con una riga: **lpd_enable="NO"**. Altri problemi, potrebbero far capo a questione di **permessi**. In sostanza, potreste stampare da root ma non come semplice utente.

In tal caso provate ad:

- editare il file /etc/devfs.conf e aggiungere (o modificare la riga) perm /dev/ulpt0 0666
- editare il file /etc/group e alla riga daemon:*:1: aggiungere il nome utente desiderato.

Comunque, leggere attentamente quanto detto nel [Wiki ufficiale su CUPS](#). Attenzione che alla riga della guida "*add path 'usb/X.Y.Z'*" dovete cercare in /dev/usb il corrispettivo file verso cui punta in /dev il file ugenX.x segnalato dal comando **dmesg**

Far ripartire il server CUPS tramite il comando **/usr/local/etc/rc.d/cupsd restart** e.... sperate che funzioni!

Per **permettere le stampe da remoto** (ad esempio attraverso Samba) dalla stampante collegata al computer dove risiede *BSD, bisogna abilitare CUPS a farlo. Si andrà nella solita amministrazione da browser (localhost:631) ed oltre a spuntare le opzioni sotto Server settings (meglio spuntarle tutte se non si hanno problemi di sicurezza), è meglio aggiungere (editando il file di configurazione aperto cliccando sull'apposito tasto a destra) sotto alla riga con scritto Port 631 e prima della riga Listen /var/run/cups.sock le righe:

```
Listen localhost:631
```

```
Listen [IP della macchina]:631 ad esempio: 192.168.1.2:631
```

Scanner USB

Con **GhostBSD** alcuni passaggi non sono necessari, come quello relativo alla gestione dei permessi.

Premesso che bisogna sempre leggere la guida ufficiale su [questo link](#). Tutta la guida ufficiale si basa sull'uso di **SANE** e di X-sane come GUI. L'uso di SANE riguarda anche scanner presenti su **stampanti All-in-one**.

Attenzione: certe volte, il vero problema è che lo scanner potrebbe non esser proprio riconosciuto come dispositivo USB. Per vedere se è visto ci sono vari comandi da usare e riguardano tutti i dispositivi USB. Intanto, però un **dmesg** a scanner acceso ci dirà se qualcosa è visto o no. Il trucco sta nel dare il comando, accendere lo scanner e ridarlo. In fondo alla schermata dopo il comando dmseg, dovrebbe apparire qualcosa. Niente? Ahia!

Dopo parecchie peripezie, un magico comando è stato trovato nascosto in una delle miriadi di discussioni in giro per la rete. Sito in inglese, naturalmente. Ecco la risposta a uno che non vedeva un dispositivo USB su *BSD data da un laconico ma geniale interlocutore: prova con il comando **sysctl hw.usb.ehci.no_hs=1**. Non commentava altro: duro e puro.

Infatti è così; dato da root nel terminale ed editando poi **usbconfig list** è possibile che il vostro scanner appaia magicamente nell'elenco dei dispositivi USB, con il suo nome identificativo. Del resto il comando **syctl** è un comando molto potente, uno dei fiori all'occhiello degli Unix-like.

Diamo allora un comando **usbconfig dump_device_desc** e vedremo tutti i dettagli (ID, vendor, ecc.) del nostro scanner. A questo punto e solo adesso, possiamo seguire la guida citata e installare, se abbiamo bisogno di Sane, nell'ordine: **sane-backends, sane-frontends e xsane**. L'installazione può avvenire o tramite compilazione o molto più velocemente con **pkg install**. Controllare comunque con un **pkg info** prima se qualcosa è installato già. Quando installerete sane-frontends vi chiederà un sacco di opzioni. Se tutto è andato a buon fine, allora si da un **sane-find-scanner -q**. Se lo scanner è trovato, tutto ok. Si passa al comando **scanimage -L** e se anche qui tutto ok, significa che lanciando **X-sane** (o anche altre interfaccia grafiche per scanner) avrete l'interfaccia grafica dello scanner pronta all'uso. In caso contrario ci sarà da editare il file **/usr/local/etc/sane.d/<nome scanner>.conf** L'unica cosa è che la guida è un po' vecchia (difetto piuttosto frequente nelle guide *BSD) ed adesso gli scanner sono ormai riconosciuti sotto il nome di **/dev/ugenX.x** (esempio del mio: **/dev/ugen7.2**) e non più sotto **/dev/usbscanner**. Inoltre, per chi usa **scanner Epson**, spesso il file da modificare non è **epson.conf** ma **epson2.conf** (state attenti!) e basta inserire il codice ID e VEND trovato con i comandi precedenti.

Scanner WIFI

Con scanner WIFI è opportuno installare anche il pacchetto **sane-airscan**. In genere lo scanner viene rilevato a patto che nel file **/etc/rc.conf** sia aggiunta la riga:

```
avahi_daemon_enable="YES"
```

Inoltre, perché sia rilevato, è necessario aprire la porta di default 6566 di 'saned' sul firewall. Stampanti multifunzione Epson e HP WIFI generalmente funzionano bene con FreeBSD.

Anche nel caso dello Scanner potrebbero esserci **problemi di permessi**. Specie in FreeBSD vi trovate ad usare lo scanner solo come amministratore. Nel tal caso come amministratore creare un gruppo usb con **pw groupadd usb** e poi aggiungere l'utente desiderato con **pw groupmod usb -m <username>** poi in **/etc/devfs/rules** aggiungere queste righe:

```
[usbscanner=10]
```

```
add path 'ugen*' unhide
```

```
add path 'usb/*' unhide
```

```
add path 'ugen*' mode 0660 group usb
```

```
add path 'usb/*' mode 0666 group usb
```

infine in **/etc/rc.conf** inserire la riga **devfs_system_ruleset="usbscanner"** quindi riavviare il servizio con **service devfs restart** e per sicurezza rimuovere la configurazione di sane nella cartella dell'utente con **rm -rf /home/<username>/.sane** (verrà ricreata da sé). E' utile anche aggiungere l'utente al gruppo 'sane' editando il file **/etc/group** e inserendo il nome utente desiderato. Dare poi un reboot. Infine, solo nel caso si sia dovuto usare il comando **sysctl** per far riconoscere lo scanner, sarà necessario rendere permanente il tutto, dato che al riavvio del computer sareste costretti a ridare il comando **sysctl**. Basta aggiungere la riga **hw.usb.ehci.no_hs=1** (senza **sysctl** prima) al file **/etc/sysctl.conf**. Esempio:

```
....  
# Allow to recognize an usb-scanner  
hw.usb.ehci.no_hs=1
```

.....

E questo dovrebbe essere tutto.

Schede di rete

Può succedere con alcune schede di rete un crash dopo l'avvio o durante l'uso. Questo inconveniente può essere risolto con un workaround inserendo nel file **/etc/rc.local** queste righe:

```
/sbin/ifconfig ale0 down  
/sbin/ifconfig ale0 up
```

Dove è necessario sostituire eventualmente 'ale0' con il dispositivo in uso dopo averlo identificato con **ifconfig**.

Altri dispositivi USB

Qui ci si riferisce alle **pendrive USB** ed agli **HD esterni USB** ed in particolare al riconoscimento, montaggio e/o automontaggio da parte del sistema operativo, una volta inseriti nella/e porta/e USB. Con **GhostBSD** non dovrete avere problemi a vedere i contenuti dei dispositivi USB che vengono montati in automatico.

Con le ultime versioni di **FreeBSD** e con Plasma KDE o MATE dovrete vedere in automount i dispositivi. Se avete problemi nell'automount, le soluzioni sono in genere tre: **provare una sequela di configurazioni**, usare un'utility apposita chiamata '**automount**', adottare il classico **sistema manuale da riga di comando**. Le procedure di configurazione trovate sulla Rete sono un pò ostiche per gli utenti cui è dedicata questa guida e non è nemmeno garantito che funzionino ma comunque le potete leggere **QUI** se volete. Più semplice usare un'utility **automount** che si trova sotto **/usr/ports/sysutils/automount** e che s'installa con i soliti comandi **make install clean**. Poi si può settare il file in

/usr/local/etc/automount.conf.sample a proprio piacimento e lo si salva come **automount.conf**. Questo è un esempio:

```
USERMOUNT=YES
ATIME=NO
REMOVEDIRS=YES
FM="nautilus"
USER=<nome user>
ENCODING=it_IT.ISO8859-1
#CODEPAGE=cp852
```

Con FM scegliete il Window manager di vostro gradimento (dolphin, pcmanfm, ecc.), però tenete presente che non tutti vi faranno vedere i dispositivi montati se non andando direttamente nella directory **/media**. Ciò significa che avrete difficoltà a smontarli se non ricorrendo al terminale. E alla fine è proprio questo che dà più garanzie, come al solito. Infatti, tenete presente che se siete abituati ad avere l'HD USB inserito ed accesso insieme al computer, allora si ricorre facilmente alla configurazione in **/etc/fstab** secondo il tipo di filesystem sull'HD. Per cui se c'è Microsoft NTFS, la riga sarà analoga a questa (dopo aver creato la directory **/mnt/hdusb**): **/dev/da0s1 /mnt/hdusb ntfs-3g rw 0 0**

Ricordarsi che con **ls /dev** vedete i dispositivi riconosciuti e che con la riga **vfs.usermount=1** aggiunta al file **/etc/sysctl.conf** date i permessi all'utente al montaggio/smontaggio. Non vale però per fusefs-ntfs (ntfs-3g) che qualche volta funziona qualche altra no.

Se invece inserite in un secondo momento il dispositivo USB lo montate da terminale con i soliti comandi e a seconda del filesystem: **mount -t msdosfs -o rw /dev/da0s1 /mnt/usb** (usato per le penne USB), oppure **ntfs-3g -o rw /dev/da0s1 /mnt/usb** o, in caso di Linux, **mount -t ext2fs -o ro /dev/da0s1 /mnt/usb**. Potete, ovviamente, in parte automatizzare questa procedura con un'icona sul desktop che punta ad uno script scritto con questi comandi.

Capitolo a parte sono i collegamenti via USB con **dispositivi Mobile**, cioè **Smartphone** (no iOS) e anche Tablet. Per dispositivi **Android** e **WindowsPhone** si ricorda che ormai lo standard di connessione di questi tramite porta USB è il **protocollo MTP**, quindi i dispositivi non sono più riconosciuti come semplici mass storage. Per vedere i files in questi dispositivi in FreeBSD è possibile usare **simple-mtpfs**, installandolo con il solito pkg install fusefs-simple-mtpfs. Creando una cartella apposita nella propria directory **/home** e montando da utente il dispositivo con il comando da terminale **simple-mtpfs /home/<utente>/<dir>** si vedranno sul file manager consueto i contenuti del dispositivo connesso, compresa la scheda esterna SD, se presente. A quel punto il trasferimento dei files verso o dal dispositivo avviene normalmente. Il test è stato effettuato su un Windows Phone con Windows 10. Se Dolphin di KDE dà problemi, installate il filemanager di MATE Caja.

Installazione Apache+MySQL+PHP+PhpMyAdmin

Qui ci si rivolge ad un'installazione in locale, ad esempio per poter effettuare delle **prove di gestione di siti WEB dinamici**. Non ha quindi la pretesa di considerare gli aspetti della **sicurezza** tipici di un'installazione di un server WEB. Comunque, a parte l'aspetto della sicurezza da affrontare con le configurazioni, i passi fondamentali per l'installazione sono analoghi.

Poichè non è possibile installare (almeno per i comuni mortali) sotto *BSD il comodo pacchetto XAMPP (esistente per Win, Mac e Linux), bisognerà installare i tre componenti separatamente.

Qualcuno si è lamentato di questo fatto e della mancanza di un porting di questo pacchetto. La risposta è che non ce n'è bisogno perchè è facile installare le componenti separatamente. E' una tipica risposta del mondo *BSD.

Installare i singoli componenti non è semplicissimo, soprattutto per le configurazioni e le guide in giro sulla rete propongono di tutto e di più. [Questa](#) l'ho trovata sintetica ma efficace. Attenzione perchè ultimamente c'è parecchia confusione con i numeri delle versioni di **Apache, PHP e MySQL** (installabile al suo posto anche **MariaDB**).

Seguendo la guida segnalata la strada consigliata è da terminale come root:

installare Apache2x (ultima versione consigliata la 2.4). Installare con *pkg install apache24*. In alternativa usare **portmaster** per installare tramite compilazione.

Inserire in */etc/rc.conf* la riga *apache24_enable="YES"*. Avviare il server da amministratore con *service apache24 onestart*. Se, durante l'avvio, pur partendo il server, vi dice qualcosa sull'impossibilità di trattare il nome di dominio, nel file */usr/local/etc/apache24/httpd.conf* decommentare la riga con "ServerName" e mettere il nome del vostro server (ad esempio localhost). Se tutto è andato liscio con il browser inserendo <nome server> (ad esempio localhost) dovrebbe uscire la frase "it works!"

Attenzione: se notate strani messaggi (tipo: "Invalid argument: Failed to enable the 'httpready' Accept Filter...") lanciando il server Apache o dando il comando */usr/local/sbin/httpd* (con il server ovviamente non in funzione), è probabile che dobbiate inserire anche una riga *apache2x_http_accept_enable="YES"* al file */etc/rc.conf*

Inserire questa direttiva nel file */usr/local/etc/apache2x/httpd.conf*

```
<IfDefine !NOHTTPACCEPT>
AcceptFilter http httpready
AcceptFilter https dataready
</IfDefine>
```

Installare MySQL installate una versione a scelta di MySQL (ultima versione 8) usando **pkg install mysql80-client mysql80-server**

Su MySQL dovrete lavorare sulla sicurezza. Qui non mi dilungo, supponendo che state in ambiente locale e su computer sicuri e non avete database di particolare importanza da proteggere! Comunque se volete andare più sicuri con il comando *mysql_secure_installation* dopo l'avvio del server potrete settare una password a vari livelli di sicurezza.

I comandi da usare per avviare MySQL sono: **service mysql-server onestart** (avvia MySQL mentre con **onestop** e **onerestart** si ferma e si riavvia). Per farlo partire in automatico al boot inserire la riga *mysql_enable="YES"* nel file */etc/rc.conf*.

Installare PHP. Prima controllare le versioni a disposizione ma consigliate quelle dalla 7.3 in poi, quindi ad esempio *pkg install php8x php8x-mysqli mod_php8x*. (dove *x=versione recente*)
installare le estensioni ad esempio *pkg install php80-extensions* e per sicurezza anche questi altri moduli: *php73-mbstring* e *php73-gettext*.

Dare il comando: *sudo cp /usr/local/etc/php.ini-production /usr/local/etc/php.ini*

Con questo copiate il file .ini di php. Bisogna poi abilitare in */etc/rc.conf* PHP-FPM con *php_fpm_enable="YES"* e avviarlo con *sudo service php-fpm onestart*.

Poi aprire un file di configurazione */usr/local/etc/apache24/modules.d/001_mod-php.conf* con queste righe:

```
<IfModule dir_module>
DirectoryIndex index.php index.html
<FilesMatch "\.php$" >
SetHandler application/x-httpd-php
</FilesMatch>
<FilesMatch "\.phps$" >
SetHandler application/x-httpd-php-source
</FilesMatch>
</IfModule>
```

Quindi riavviare il server apache e per vedere se funziona lo scripting di PHP, aprite un file di testo e inserite:

```
<?php phpinfo(); ?>
```

salvate con il nome *test.php* e inseritelo nella directory ***/usr/local/www/apache2x/data***.

Ricordate: questa è la directory dove dovrete piazzare le vostre pagine PHP. Ora, riscrivendo sul browser **localhost/test** (oppure *localhost/test.php*), dovrete vedere delle pagine informative sulla versione di PHP e Apache installata nonché i parametri di configurazione presenti.

A questo punto possiamo installare il PhpMyAdmin che ci permette di gestire i databases MySQL (crearli, vederli, modificarli, ecc.). L'installazione di PhpMyAdmin, che si gestisce attraverso un browser richiamandolo in genere all'indirizzo "localhost/phpmyadmin", non è

difficile in sé ma può comportare malfunzionamenti dovuti alle configurazioni non proprio agevoli da fare. Per installarlo, cercate con *pkg search phpmyadmin* la versione adatta alla vostra installazione di PHP (le versioni devono coincidere). Poi seguire questo [WIKI](#) che dà un'idea delle cose da configurare. E' molto probabile che dovrete aggiungere dopo l'installazione di phpMyAdmin queste righe in fondo al file httpd.conf (proprio alla fine del file):

```
Alias /phpmyadmin "/usr/local/www/phpMyAdmin/"
<Directory "/usr/local/www/phpMyAdmin/">
Options None
AllowOverride Limit
Require local
Require host localhost
</Directory>
```

Poi fate un restart di Apache. Date quindi da amministratore il comando:

```
cd /usr/local/www/phpMyAdmin && sudo cp config.sample.inc.php config.inc.php
```

Restringete i permessi del file a 444 dopo aver modificato la linea `$cfg['blowfish_secret'] =` aggiungendo una serie di caratteri alfanumerici. Ad esempio:

```
$cfg['blowfish_secret'] = 'Ey0r*h!5g#oNf5WvkosW)*H$jasn%$!1'
```

Ora basta andare sul browser inserire ad esempio un localhost/phpmyadmin e s'accede alla schermata iniziale del tool che richiede l'utente e la password del gestore del database. Attenzione: se appare un warning relativo alla crittografia sha di mysql allora dovrete andare come amministratore dentro MySQL con il comando `'mysql -u root -p'` e inserire questo comando:

```
alter user 'root'@'localhost' identified with mysql_native_password by 'pwd';
```

naturalmente `'pwd'` va sostituito con la propria password. Poi si esce dando `'exit'`.

Ricordarsi comunque che potrebbe essere necessario nel file config.inc.php avere le due righe:

```
$cfg['UploadDir'] = '/tmp';
```

```
$cfg['SaveDir'] = '/tmp';
```

Dove la directory /tmp è aperta alla lettura e scrittura per gli utenti.

Attenzione: può darsi che PhpMyAdmin fornisca dei *warning* relativi a delle funzioni disabilitate. In tal caso per rendere completa l'installazione, è opportuno creare un user *"pma"* e un *database "phpmyadmin"* su MySQL. L'utente deve essere creato prima a livello di sistema (con il comando `adduser`) e poi aggiunto nella tabella user del database *mysql*. Controllate comunque se esista già un utente `_pma` (con l'underscore) installato di *default* durante la compilazione da FreeBSD. In tal caso modificherete solo il nome dell'utente in `_pma` nella tabella user e nel file config.inc.php. Il database va invece creato importando il file

create_tables.sql presente nella directory `/usr/local/www/phpMyAdmin/examples/`. Solo così le righe presenti nell'esempio a partire da quella con `$cfg['Servers'][$i]['pmadb'] = 'phpmyadmin'` e fino alla fine della configurazione del server avranno effetto. Ricordatevi di riavviare il server MySQL dopo queste operazioni.

Infine, specie su FreeBSD, se PhpMyAdministrator dà messaggi di file not found, potrebbe essere necessario fare un link da `/usr/local/www/<cartella di phpmyadministrator>` verso `/usr/local/www/apache22/data/`

Gestione binari Linux

FreeBSD ha la possibilità di usufruire dei binari Linux, per ovviare alla carenza di applicativi ed altro, spesso associata con lo scarso supporto delle aziende di software e hardware per i sistemi Unix. In sostanza, si ammette che per Linux c'è più roba in giro che per i *BSD.

Per farsi un'idea della gestione dei binari Linux, bisogna leggersi la [guida ufficiale](#)

In pratica, dato che **linux_base-c7** è considerato ormai obsoleto, installare Rocky linux base con **pkg install linux_base-r19**.

Se tutto è andato a buon fine aggiungerete al file `/etc/rc.conf` la riga:

`linux_enable="YES"`. Se ci sono problemi provate ad editare il file **`/etc/fstab`** e aggiungere questa riga:

```
linproc /usr/compat/linux/proc linprocfs rw 0 0
```

Tuttavia con dispositivi non recenti, Rocky linux base potrebbe non installarsi con un warning intorno ad errori glibc legati alla CPU che non supporta x86-64-v2. In tal caso c'è una opzione che permette di installare una vera e propria distro base linux debian/ubuntu dentro FreeBSD, con l'uso debootstrap (installare con **pkg install debootstrap**) e poi scegliere una distro ad esempio Ubuntu jammy con il comando **debootstrap jammy /compat/ubuntu**. Usando poi **chroot /compat/ubuntu /bin/bash** si entra nella distro linux dopo aver modificato il file `/etc/fstab` e eseguito il mount, come descritto nella guida citata prima.

Anche se tutto questo vi permetterà di poter installare applicativi Linux in FreeBSD, tenete presente che non è così semplice per frequenti problemi di incompatibilità di librerie,

Samba

Si presuppone che sappiate cosa sia Samba, come si installa e configura. Nei *BSD bisogna fare le stesse procedure, seguendo magari la [guida ufficiale](#)

Qui si affronta un caso particolare di post-installazione per un collegamento tra due computer, magari meno frequente ma penso utile: **collegare un *BSD con un computer**

Apple (MacOSX Leopard e Snow Leopard) via Samba.

Una volta installato Samba sul *BSD dovrete vedere le cartelle sull'Apple da *BSD (si dà per

scontato come già effettuata la procedura lato Apple per rendere pubbliche una o più cartelle nel MacOSX). Il modo migliore per farlo (non fallisce mai o quasi) è inserire l'indirizzo IP (qui un esempio supponendo che la macchina abbia IP 192.168.1.4) del computer Apple cui bisogna connettersi. Si può fare in vari modi (esempio in ambiente KDE4):

Dolphin ---> Rete --- Samba shares ---> inserire 192.168.1.4 in alto;

Dolphin ---> cliccare tasto sx in alto sotto alle icone (aprirete una finestrella per immettere indirizzi come con konqueror) e scrivere smb://192.168.1.4;

Aprire Konqueror (buon vecchio konqueror!) e scrivere l'indirizzo smb://192.168.1.4;

Aggiungere un'icona al desktop scrivendo come URL: smb://192.168.1.4

Il problema è che, nonostante abbiate settato correttamente i permessi in MacOSx, dalla macchina Apple potreste invece non vedere le cartelle condivise in *BSD.

Ci sono varie procedure lato *BSD per riuscirvi, provate nell'ordine:

- aggiungere (da root) un utente per Mac con il comando `adduser -a [nome utente]`. Se le cartelle da vedere appartengono ad un utente già esistente potete saltare questo passo, usando magari il nome dell'utente già esistente;
- aggiungere però lo stesso utente a SAMBA con `smbpasswd -a [nome utente]` dove la pwd è ovviamente uguale a quella impostata per l'utente creato o esistente. Cioè se devo vedere le cartelle di un utente già esistente "pippo" devo solo aggiungere a SAMBA l'utente con: `smbpasswd -a pippo` e inserire la pwd di quell'utente che già è in uso. Ritornate sul Mac, chiedere la connessione al server, scegliere l'opzione no ospite (utente registrato) e digitare nome utente e pwd;
- accertarsi che nel file di configurazione di Samba `/usr/local/etc/smb.conf` ci sia una riga con **encrypt passwords = true** e rilanciare Samba con **`/usr/local/etc/rc.d/samba restart`**;
- se ancora non vedete le cartelle forzate i permessi della cartella/e condivisa/e a `chmod 777`, in pratica aperta/e a chiunque (`chmod -R 777 /[directory]/[cartella]`);
- se non funziona ancora usare da console il comando **`pdbedit -a -u <username>`** con gli stessi criteri di `smbpasswd` (immettere una pwd di un utente esistente su *BSD) e rilanciare samba

In genere, applicando le **prime tre procedure** dovrete vedere la cartella condivisa su *BSD dal Mac!!

Leggersi comunque la [guida ufficiale](#).

Una volta viste le cartelle da entrambe le macchine, bisogna configurare su Apple la stampante collegata al *BSD. Per farlo, meglio scordarsi il comodo settaggio previsto in preferenze di sistema del vostro MacOSX. Aprirete invece il browser Safari (solo Safari non Firefox o altro altrimenti non vi farà accedere) e digiterete nella barra indirizzi **localhost:631**. Esattamente come negli Unix, perchè MacOSX è comunque un Unix-like e peraltro CUPS è un

prodotto della casa di Cupertino, pensate un po'! Cliccate su "Amministrazione" e poi "aggiungi stampante" sul menu e sceglierete la seconda configurazione ipp (ipp) e continuate. Nella casella seguente che apparirà scriverete questo indirizzo: **ipp://<indirizzo server>:631/printers/<nome stampante>**

Per esempio, supponendo che il computer *BSD abbia indirizzo 192.168.1.3 ed il nome della stampante nel file di configurazione **smb.conf** di *BSD sia "myprinter", l'indirizzo sarà `ipp://192.168.1.3:631/printers/myprinter`.

Continuate a configurare la stampante cercando il driver per il modello come fareste nell'ambiente usuale del MacOSX, riprodotto in modo analogo sul browser ora. Se tutto è andato bene, cercando normalmente nelle preferenze di sistema ---> stampanti e fax, dovrete vedere come inattiva ma pronta l'iconcina della stampante configurata... non è poi così difficile.

Ovviamente bisognerà poi configurare il file **/usr/local/etc/smb.conf**

Attenzione. Per la stampante controllare che il Firewall (specie con `OpenBSD packet-filter`) sia aperto alla porta 631 tramite le righe spiegate sopra nella sezione "Firewall". Inoltre controllare l'abilitazione nel file di configurazione di CUPS alla stampa da remoto come spiegato nella sezione "stampanti".

Caratteri TrueType MS

Se non vedete i caratteri TrueType di MS in LibreOffice leggere la [guida ufficiale](#) sotto la voce TrueType può non bastare, dato che è un tantino obsoleta. Leggete [questo post](#) che dovrebbe risolvere la questione. Meglio avere a disposizione i caratteri TrueType che servono tratti da un Windows (in genere `C:\Windows\Fonts`) per metterli in `/usr/local/share/fonts/TTF` e poi aggiungere al file di configurazione di Xorg nella Section "Files" il path `(/usr/local/share/fonts/TTF)`

Tastiera italiana nella console.

Se nella console non potete usare la tastiera in italiano dovete aggiungere al file `/etc/rc.conf` la stringa come: **keymap="it.iso"**

Visualizzare i dispositivi USB connessi. Molti comandi in *BSD sono diversi rispetto a Linux. Per vedere i dispositivi USB connessi in Linux si da `lsusb` o simili. In BSD il comando è **sysctl** (un po' diverso) ma dalla FreeBSD 8.0 in poi è **usbconfig**.

Controllo della quantità di RAM gestita

Dato che ormai i computer hanno RAM sempre più grandi, è sempre bene vedere quanta RAM il sistema operativo riesce a gestire e in che modo. Sotto Linux in genere si usa il comando **free**, seguito da opzioni (es.: `free -m`). Sotto *BSD si può usare il comando

freecolor (installare con `pkg install freecolor`). Ora dando il comando **freecolor -m -o** da terminale avrete la quantità di memoria RAM gestita e altri dettagli.

Riproduzione CD AUDIO

Mentre non dovrete avere problemi a riprodurre files .mp3 ed altri files multimediali, potrebbe accadere che con i **normali CD Audio** sia complicata la riproduzione. Purtroppo è un problema dei *BSD e in giro per la rete è pieno di richieste d'aiuto per questo problema. Ovviamente per risolverlo diamo per scontato che la scheda audio sia stata riconosciuta e funzioni. Ciò significa che potete ascoltare tranquillamente l'audio di altri files multimediali, tipo filmati, riproduzione .mp3, ecc. Altro presupposto è che il lettore CD/DVD funzioni correttamente e sia in grado di aprire i CD o DVD con altri contenuti, come files di dati, foto, ecc.

I problemi possono essere di due tipi: l'audio da un CD audio inserito nel lettore non va, nonostante il CD inserito sia riconosciuto dal lettore e si accende la spia apposita. E' un problema che su FreeBSD può accadere con una certa frequenza. La soluzione è quella di provare a far funzionare l'audio del CD musicale con programmi appositi. Qui si consigliano due soluzioni che in genere funzionano.

Prima soluzione: uso di programmi appositi. Nonostante sul Web si dica che **MPlayer** funziona, ho constatato che anche questo applicativo potrebbe non funzionare. L'alternativa è data da **VLC**. Installarlo con un **pkg install vlc**. Una volta installato, lanciarlo, andare sulla scelta "**Media**" e poi "**Apri disco**". Spuntare la **casella CD audio** e inserire il device corretto che, in caso di un solo lettore sul PC, è **/dev/cd0**. **Attenzione:** /dev/adc0 è obsoleto dalla versione 9. Dopodichè tasto in basso "**Riproduci**" e il CD si avvierà e ascolterete la musica dalla prima traccia (testato su FreeBSD e GhostBSD). Inoltre, con KDE (versione plasma 5) nel file manager Dolphin potete vedere in basso il contenuto del CD audio ma l'ascolto completo del CD audio non è molto agevole, meglio usare direttamente gli applicativi prima consigliati. Mentre è interessante usare Dolphin per trasformare le tracce in formati .mp3 o .flac direttamente copiando i files audio in altra destinazione.

Seconda soluzione: uso comando **cdda2wav** (installato con `pkg install cdrtools`). Si tratta di una soluzione a riga di comando e quindi non molto comoda ma è utile per testare se c'è soluzione nel vostro computer a questo problema. Bisogna dare da terminale prima il comando **sudo cdda2wav -scanbus**. Si otterrà un elenco analogo:

scsibus2:

2,0,0 200) *

2,1,0 201) 'HL-DT-ST' 'DVD-RAM GH22LS30' '1.00' Removable CD-ROM

2,2,0 202) *

2,3,0 203) *

2,4,0 204) *

2,5,0 205) *

2,6,0 206) *

2,7,0 207) *

Alla terza riga come si può vedere è stato individuato il lettore con tre numeri che lo identificano. A questo punto, dare il comando **cdda2wav -eN dev=2,1,0 -t1**, dove ovviamente i 3 numeri del dev= sono presi dall'output del vostro comando precedente relativi al dispositivo cdrom trovato. In genere il CD Audio dovrebbe partire e dovrete sentire la musica.

Si segnala che il comando **cdcontrol**, spesso suggerito nei Forum per trovare soluzione al problema, non funziona in quanto il CD si avvia ma non esce il suono. Meglio usare **cdda2wav**.

Se nonostante tutto il CD audio non si sente, spiacente ma ho esaurito le soluzioni...

Altro tipo di problema è più complicato ed è connesso a problemi di permessi. In sostanza, accedete solo da root alla lettura del CD audio. Per testare se questo è il problema basta vedere se il comando **cdda2wav** visto prima funziona da root ma fallisce da utente, oppure lanciare **audacious** come amministratore. Per tentare di risolvere, date il comando da terminale **camcontrol devlist** e dovrete vedere nell'elenco il riferimento al vostro dispositivo ottico, con una serie di parametri. Ad esempio: <ASUS DRW-24B3ST 1.00> at scbus3 target 1 lun 0 (pass1,cd0).

Andate a modificare il file **/etc/devfs.conf** che deve contenere questi parametri:

```
own /dev/cd0 root:operator
```

```
own speaker root:operator
```

```
perm /dev/cd0 0666
```

```
link cd0 cdrom
```

```
link cd0 dvd
```

```
perm speaker 0666
```

```
perm /dev/xpt0 0666
```

```
perm /dev/pass0 0666
```

```
perm /dev/pass1 0666
```

```
perm /dev/pass2 0666
```

Notare che deve essere presente un **perm /dev/passX** con il numero uguale a quello dato dal comando camcontrol in precedenza. Inoltre l'utente deve esser nel gruppo 'operator'.

Dare poi il comando da amministratore **/etc/rc.d/devfs onerestart** o riavviate il computer.

Supporto Java

Con **FreeBSD** bisogna installare il pacchetto **openjdk** da **/usr/ports/java/** oppure con **pkg install openjdk**

Per il **plugin nel browser** si deve poi installare il plugin **icedtea-web** o con i ports o con il comando *pkg install icedtea-web* e fare il link nella directory **/<nome utente>/.mozilla/plugins** al plugin con suffisso **.so** che si troverà nella directory **/usr/local/lib/IcedTeaPlugin.so**

Ulteriori spiegazioni al [link della guida ufficiale](#). Andare al capitolo 7.2.2 della guida.

VirtualBox (virtualizzare Windows su FreeBSD)

Anche sui *BSD è possibile rendere il sistema operativo un host per una macchina virtuale guest che virtualizza un altro sistema operativo a scelta. Qui si presuppone che abbiate una conoscenza minimale delle possibilità di virtualizzazione. Si è scelto come esempio **VirtualBox** (ex prodotto Sun ora sotto Oracle) perchè effettivamente è al momento uno dei più usati (almeno per un uso normale) software di virtualizzazione totalmente libero in circolazione.

Nel caso particolare, ci si riferisce alla possibilità di far girare **WindowsXP** e **Windows 7** dentro a FreeBSD esattamente come l'originale, come sa bene chi è avvezzo alle virtualizzazioni! E' appena il caso di ricordare che virtualizzare un Windows è oltremodo comodo dentro ad un *BSD, dato che tali sistemi soffrono ancor più delle distro Linux di alcune limitazioni nel parco software a disposizione.

Premessa importante è che sui *BSD è possibile installare solo la versione OSE di Virtualbox. Comunque, dato che nei ports ormai si trova la versione 4.x.x di Virtualbox, questa implementa anche il **supporto ai dispositivi USB**, a differenza della versione 3 che non li supportava.

Per installare Virtualbox-OSE su FreeBSD è meglio farlo con i pacchetti binari dando il comando:

pkg install virtualbox-ose-kmod e poi **pkg install virtualbox-ose**

Alla fine dell'installazione attenzione a fare un copia e incolla delle avvertenze che appaiono sul terminale. Infatti, dovrete:

- caricare il modulo vboxdrv scrivendo **vboxdrv_load="YES"** nel file **/boot/loader.conf**;
- aggiungere la riga **vboxnet_enable="YES"** nel file **/etc/rc.conf**;
- aggiungere gli utenti che useranno VirtualBox al gruppo "vboxusers" dando il comando **pw groupmod vboxusers -m <nomeutente>**;
- fare un reboot;

Si avverte anche che eseguire VirtualBox come utente non root potrebbe causare un errore

"NS_ERROR_FACTORY_NOT_REGISTERED". In tal caso cancellate il file /tmp/.vbox-*-.ipc

[Questa comunque è la guida ufficiale](#) di FreeBSD, anche se potreste non avere bisogno, per fortuna, di tutte le procedure descritte...

Ora dobbiamo avere solo una copia di WindowsXP o Seven a disposizione per installarlo nella VirtualBox.

La configurazione di VirtualBox è analoga a quella usuale per altri sistemi operativi host (compresa l'installazione delle guest additions), ricordando solo che per **Windows 7**, come spiegato anche nel sito ufficiale di Oracle, è necessario installare manualmente i driver audio, scaricandoli da [qui](#). Per risolvere eventuali problemi con l'audio [consultare anche la pagina del mio sito](#), appositamente dedicata alla questione audio con Windows 7 virtualizzato.

Come configurare la macchina guest Windows 7 è spiegato bene in [questo sito](#), dove troverete anche ulteriori notizie sulla configurazione da fare in FreeBSD nel caso quello che ho scritto prima non bastasse e ci fossero problemi.

Comunque, dato che si sono verificati qualche fastidioso **freeze** completo del computer con una **guest Windows 7 Ultimate**, riporto una configurazione di VirtualBox funzionante con una scheda grafica Nvidia (le altre opzioni lasciarle come da default):

Sistema – Scheda madre (**yes** solo so **IO/APIC**) – Processore (**no** a **PAE/NX**) – **yes** a **VTx/AMD-5** – **yes** a **paginazione nidificata**;

Schermo – potete abilitare l'accelerazione 3D e 2D altrimenti se crea problemi toglietela;

Archiviazione – Sotto Controller IDE ---> **Tipo ICH6**; **yes** a **Usa cache**;

Archiviazione – Sotto Controller SATA – **yes** a **Usa cache**;

Scheda di Rete – **NAT** – Avanzate ---> **Intel PRO/1000 MT Server (82545em)**

Ricordo solo il comando da dare in Windows per aprire una comoda **cartella condivisa** creata in VirtualBox e corrispondente ad una cartella presente in *BSD. In Windows da **Start (Avvio)** ---> **Esegui**, scrivere il comando **net use x: \\vboxsvr**<nome cartella condivisa>****

(attenzione allo spazio dopo la "x:" Si ricorda che in Windows 7 è scomparso da "Start" il comando "Esegui". Qui un [rimando per come fare ad attivarlo](#).

Collegamento DLNA/UpPnp con dispositivi multimediali (Playstation, TV, ecc.)

Con FreeBSD è possibile collegare dispositivi DLNA attraverso applicativi multipiattaforma.

Come [Serviiio](#). Usare i ports sotto **usr/ports/net/serviiio**. Compilarlo o meglio creare un pacchetto binario con il comando 'make package' (come fare è spiegato nella sezione 'installazione programmi') e poi installarlo. Aggiungere nel file /etc/rc.conf le seguenti righe:
serviiio_enable="YES"
serviiio_args='-Dserviiio.advertisementDuration="15" -Dserviiio.remoteHost="192.xxx.xxx.xxx"'

Sostituire l'indirizzo IP con quello della macchina dove gira Serviio.

Se usate un firewall potrebbe essere necessario aprire la porta 8895/TCP e 1900/UDP.

Riavviando la console di Serviio sarà disponibile all'indirizzo "http://<IP assegnato nel file rc.conf>:23423/console".

In genere non si dovrebbero avere problemi particolari di rilevamento. Può succedere che il comando 'service serviio status' restituisca un 'not running' nonostante il mediaserver sia avviato. In tal caso provare ad aprire il file 'serviio' in '/usr/local/etc/rc.d' e cambiare la riga dello script d'avvio che inizia con 'serviio_pid' così:

```
serviio_pid=$(ps -wwaU ${serviio_user} | awk '/java/&&/serviio/ {print $1}')
```

Da questo punto in poi, se tutto funziona, dovrete vedere l'icona del mediabrowser sul vostro dispositivo collegato per gestire i files multimediali che si trovano sul computer, compresi i flussi video (utile per la funzione restreaming nello script RaiNixV3 segnalato nel paragrafo qui successivo).

Vedere le dirette del sito della RAI

La RAI trasmette ormai le dirette sul suo sito anche per Linux e BSD, compresa RAI Replay (con qualche eccezione). Tuttavia un'alternativa, **svincolata dall'uso di qualsiasi browser** ed anche dalla pubblicità iniziale nonché dalla registrazione per vedere alcuni contenuti, è usare uno **script** per Linux il cui download si trova sul sito di questa guida: [RaiNixV3](#). Permette la visione e registrazione delle dirette (comprese le Radio RAI), Rai Replay e dei programmi RAI on demand. Inoltre è possibile fare il restreaming dei video su altri dispositivi in grado di leggere flussi multimediali su rete locale (smartphone, tablet, altri computer, android TV, ecc.). Con le SmartTV il flusso può essere anche gestito da un mediaserver come Serviio. Per eseguire lo script sotto FreeBSD è necessario avere installati **xterm, curl, wget, ffmpeg, dialog e jq** (facoltativo **youtube-dl e streamlink**). Attenzione: 'dialog' è stato sostituito con il pacchetto **cdialog**, quindi installate questo e fate un link simbolico così: ln-s /usr/local/bin/cdialog /usr/local/bin/dialog. Inoltre installare i players multimediali **mpv** (fork di Mplayer) e **VLC**. Se con ffmpeg installato con pkg non avrete a disposizione il player 'ffplay', bisogna compilare da ports **ffmpeg** con l'opzione **X264** e **SDL abilitate** (comando **make config** prima di compilare). Per le registrazioni modificare l'ordine nella riga 'Recorders=' mettendo ffmpeg o youtube-dl al primo posto. Lo script si lancia da terminale con nel solito modo e con **bash rainixv3.sh si ha il debug**. **CLICCA QUI per avere comunque spiegazioni ulteriori**.

Connessione Web con [TOR](#) (connessioni anonime)

Quest'operazione può essere eseguita per rendere più sicura la navigazione sul Web. La **rete**

TOR ci fornirà un indirizzo IP alternativo con cui navigare. Non è la sede per discutere i pro ed i contro e sull'effettiva sicurezza garantita: si presuppone che l'utente sappia già qualcosa sull'argomento. Naturalmente anche sotto un *BSD si può usare TOR. Tuttavia che io sappia non è tanto facile usare il comodo Tor-browser che si scarica dal sito TOR (previsto per Linux). L'installazione necessita quindi di un proxy, qui si usa in particolare '**privoxy**'. Per prima cosa andiamo ad installare 'TOR' e come proxy 'privoxy' dai ports. Il primo è in

/usr/ports/security/tor mentre il secondo si trova sotto **/usr/ports/www/privoxy**.

Entrambe le volte useremo da root il comando **portmaster**.

E' importante dare un'occhiata ai messaggi finali dopo l'installazione di entrambi che ci informano sulla settatura di alcune cose e di come avviare i servizi. In particolare, per TOR, bisogna in un terminale come amministratore dare in sequenza questi comandi:

```
rm -r /var/db/tor /var/run/tor (il più delle volte non necessario perché non esistono le directory)
```

```
mkdir -p /var/db/tor/data /var/run/tor
```

```
touch /var/log/tor
```

```
chown -R _tor:_tor /var/db/tor /var/log/tor /var/run/tor
```

```
chmod -R 700 /var/db/tor
```

Per far partire TOR dovete fare anche una copia del file **/usr/local/etc/tor/torrc.sample** in un file **torrc** (controllate se già non ci sia) e decommentare la riga 'RunAsDaemon 1' per farlo partire come demone. Poi avviate con **service tor onestart** ed al boot aggiungendo al file **/etc/rc.conf** la riga **tor_enable="YES"**.

Privoxy va avviato a parte prima con **service privoxy onestart**, oppure, ad ogni boot, inserendo nel file **/etc/rc.conf** la riga **privoxy_enable="YES"**. Il punto delicato è il file di configurazione denominato 'config' da avere nella cartella **/usr/local/etc/privoxy**. Infatti, c'è un file d'esempio ma se non ci sapete mettere le mani capirete ben poco. Comunque potete tranquillamente copiare il file d'esempio e lasciarlo così com'è tranne alla **sezione 5.2** (forward socks) dove dovete inserire queste righe (comprehensive del punto):

```
forward-socks5 / 127.0.0.1:9050 .
```

```
forward-socks4a .onion 127.0.0.1:9050 .
```

Per problemi con **Privoxy** seguire [questa guida](#).

Fatto questo, non resta che controllare eventualmente il Firewall. Se usate il file **/etc/pf.conf**, potreste avere bisogno di aprire la porta 8118 aggiungendo una riga analoga a questa e dopo rifare la ricarica delle regole con **pfctl -f /etc/pf.conf**:

```
pass in on ale0 proto tcp from any to (ale0) port 8118 keep state
```

A questo punto il funzionamento del tutto avviene come al solito. Avviato il proxy e Tor dovete abilitare ad esempio Firefox alla rete con proxy cliccando su Settings >Network Settings>Settings. Qui abilitare la casella della configurazione manuale del proxy ed inserire

per ogni preferenza 127.0.0.1 e porta 8118 ed infine abilitare socks5. Noterete subito se siete nella rete Tor andando all'indirizzo <https://check.torproject.org/> e quindi da questo momento la navigazione sarà relativamente anonima. Naturalmente che appaia un indirizzo IP di località diversa da quella italiana o viceversa può essere d'aiuto in molti casi e si ricorda è possibile settare il file di configurazione **torcc** presenta nella vostra directory user per forzarlo ad usare indirizzi IP (o anche uno solo) proveniente da zone geografiche specifiche. Questi argomenti però esulano da questa guida. Inoltre, ricordate che non state usando il tool "Firefox Tor Bundle" e quindi dovrete da voi disabilitare e/o abilitare varie opzioni o gli addons (in primis Adblock) necessari per una navigazione più sicura. Installando anche '**torsocks**' con *pkg install torsocks* come sapete c'è la possibilità di usare la rete Tor anche con gli applicativi, lanciando da terminale *torify <nome applicativo>*. Tuttavia non si danno garanzie che l'applicativo in questione stia usando la rete Tor. Potete comunque controllare in parte lanciando ad esempio da terminale *torify xterm* e scrivendo poi *curl -s api.infoip.io/ip* che vi restituirà un indirizzo IP facilmente controllabile se diverso da quello del vostro provider.

In alternativa, è possibile seguire [QUESTA guida](#) con la quale otterrete gli stessi risultati e cioè di un browser collegato alla rete TOR, senza usare 'privoxy'.

Nota: la scelta di privoxy è dovuta anche alla sua facile configurazione se volete usare anche la **rete i2p** (analoga a quella .onion di TOR). Tuttavia su FreeBSD potreste avere difficoltà ad usare insieme le due reti con privoxy anche inserendo la riga **forward .i2p localhost:4444** alla sezione 5.1 del suo file 'config'.

Accedere al sistema per riparare eventuali errori

Potrebbe capitare che, per qualche motivo, il sistema non parte e bisogna intervenire per risolvere errori da console, dopo il boot. La procedura in un *BSD è leggermente diversa da quella su Linux e potrebbe spiazzare il neofita. Se il boot fallisce e c'è necessità di entrare per controllare il sistema, bisogna lanciare l'opzione **single user mode** se viene presentato il menu di scelta all'avvio (in genere opzione 2). Qualora non si presenti il menu, bisogna editare **boot -s** al prompt. Terminato l'avvio dare ENTER (come consigliato dal messaggio che appare) per entrare nella shell. Per accedere come root editare **mount -uw /**
A questo punto, per poter avere accesso ai programmi e a tutti i comandi del sistema dovrete editare **mount -a** Attenzione: se vi trovate in un **filesystem zfs** il comando da editare è: **zfs mount -a**

Dopo aver dato questi comandi sarete in grado di avere accesso alle directory dell'intero sistema e quindi modificare files di configurazione, cancellare, copiare, ecc. (ovviamente da riga di comando!).

Se la connessione di rete non funziona provate a dare il comando **/etc/rc.d/netif start** ed

anche **service routing start**.

Altro problema che può accadere è che FreeBSD non trova più la partizione di sistema, causa modifiche alla geometria dell'HD (ad esempio in caso di installazione di sistemi Windows sullo stesso HD). In tal caso il boot fallisce e vi ritrovate con un **mountroot>** al prompt. La soluzione è rifare un reboot ed accedere come **single user mode**, poi scrivete '?' al prompt mountroot> e vedrete le partizioni a disposizione. Individuata la partizione cambiata dove probabilmente risiede il sistema, date **ufs:/dev/adaXsXX** (cambiando le x ovviamente). Al prompt date il solito comando **mount -uw /** e poi con un editor (vi, nano) modificate e salvate il file /etc/fstab con il /dev giusto, poi fate un reboot.

Qualora sia impossibile accedere alla schermata di boot si può ricorrere al DVD di FreeBSD che funziona anche come 'live' anche se solo da terminale. Conviene lanciare il DVD e scegliere alla schermata l'opzione 'SHELL'. Accedendo alla shell dei comandi è possibile montare la/e partizioni dove si trova la vostra versione di FreeBSD installata. Per conoscere le partizioni si può usare il comando 'gpart show'. Una volta conosciuta la partizione di FreeBSD si può montare con **mount /dev/adaXsXX /mnt** (dove x=numero e lettera della partizione e slice) ed accedervi per manutenzione. Se c'è bisogno di riparare il filesystem dare direttamente un **fsck -y /dev/adaXsXX**

Usare OCR con *BSD

E' possibile usare programmi OCR su *BSD come con Linux. La preferenza è qui accordata a **OCRFeeder + tesseract**. L'unico problema è che per essere sicuri di avere un OCR ben funzionante bisogna compilare e questo richiederà molto tempo a causa delle numerose dipendenze da risolvere. Meglio usare il comando pkg. Quindi installate prima **ocrfeeder**, poi **tesseract** ed infine **tesseract-data**

Finalmente dovremmo essere in grado di lanciare OCRfeeder. Si presuppone che abbiate già installato uno scanner funzionante sotto *BSD. Dopodichè OCRfeeder va usato nel solito modo, ricordandosi di modificare sotto **Strumenti --> Motore --> tesseract --> modifica**, la linea di comando aggiungendo un **-l ita** alla stringa **\$IMAGE \$FILE** (cioè farla divenire: **\$IMAGE \$FILE -l ita**). Questo per usarlo con la lingua italiana. Potreste avere un **problema di permessi**, nel senso che OCRfeeder fa il riconoscimento testi solo da root e non come user. In tal caso, **modificate i permessi** alla cartella **/usr/local/share/tessdata** dando un **chmod -R 777 /usr/local/share/tessdata**. Fatta questa operazione dovrete essere in grado di usare il riconoscimento testi come normale utente. Naturalmente potete usare anche altri motori di riconoscimento (Ocrad, GOCR) installandoli sempre con pkg.

Upgrade del sistema ad una versione successiva

L'upgrade non è difficile ma l'esito non sempre è privo di problemi. Infatti dopo l'upgrade in sé efficace, potrebbero manifestarsi errori soprattutto in riferimento agli applicativi quando

si passa ad una major release (dalla 13 alla 14 ad esempio) dato che l'**upgrade** **aggiorna solo il sistema operativo**. Con **FreeBSD** useremo la procedura [descritta nella sezione 25.2.3.2 del manuale ufficiale](#). In pratica si tratta di dare prima, da terminale e come amministratore, i due comandi **freebsd-update fetch** e poi **freebsd-update install** e poi il comando **freebsd-update -r 15.0-RELEASE upgrade**. Naturalmente se l'upgrade è ad una versione diversa bisogna scrivere il nome corrispondente. Partirà una lunga procedura e bisogna stare attenti a leggere in inglese attentamente i messaggi che ogni tanto compariranno. Dare "yes" ai messaggi che avvertono che alcuni componenti non sembrano installati. Successivamente apparirà il messaggio di **inspecting system** e poi **preparing to download**. Naturalmente i tempi dell'upgrade saranno condizionati dalla velocità della connessione. Dopo l'applicazione delle patches necessarie, si passa ad una **fase più delicata** in cui probabilmente **FreeBSD** chiederà di agire manualmente su alcuni files di configurazione attraverso l'**editor da terminale vi**. Se non sapete bene cosa fare è buona norma entrare in ognuno di essi, non modificare nulla e poi premere ESC e scrivere in basso a sinistra **:x** (inserire due punti e x) e così salverete il file. Successivamente **FreeBSD** potrebbe chiedere conferma se è ragionevole che esistano certi parametri in alcuni files. Anche qui se non sapete cosa fare dite sempre "yes". Alla fine, dovrebbe partire una lunga lista di files da upgradare che bisogna scorrere il più velocemente possibile (tasto pagina giù) per arrivare alla fine e seguire l'istruzione data, cioè digitare nel terminale: **/usr/sbin/freebsd-update install**. **FreeBSD** installerà il nuovo kernel e poi chiederà un **reboot** (digitare **reboot** e tasto invio). Al **reboot** starete già dentro al nuovo ambiente e sempre come root ridate esattamente il comando di prima. A questo punto **FreeBSD** installerà tutti i pacchetti necessari all'upgrade (non quelli degli applicativi trovati ma quelli del server grafico, ecc.). Di nuovo date un **reboot** ed avrete finito l'upgrade. Tuttavia questo è vero se l'upgrade è stato tra due minor release (ad esempio dalla 14.2 alla 15.0), altrimenti sarà necessario aggiornare i pacchetti e altri programmi compilati (ad esempio passando dalla versione 13 alla 14). In tal caso è bene seguire le **istruzioni del paragrafo 23.2.3.2** del link riportato prima e cioè aggiornare i pacchetti con un **pkg-static upgrade -f** ed eventuali compilazioni con **portmaster -a** cui seguirà alla fine un nuovo **freebsd-update install**. Per accertarsi di aver installato la versione nuova, da terminale: **freebsd-version**.

FreeBSD probabilmente non aggiornerà i **sorgenti** eventualmente presenti in **/usr/src** e dovrete, dove aver cancellato l'intera directory **/usr/src**, ricrearla e reinstallare i sorgenti con la procedura solita. Accertate con il comando **svn info /usr/src** quale versione dei sorgenti avete installati. La stessa cosa per **kernel customizzati** che dovrete ricompilare dopo l'upgrade. Inoltre dopo l'upgrade potrebbe non funzionare più bene il comando **pkg**. In tal caso, provate a dare un **pkg clean -a** e poi nuovamente **pkg update**.

Compilazione di un nuovo kernel

Può accadere che ci sia bisogno di un kernel customizzato. Ad esempio, come è stato detto

nell'apposita sezione, per includere il PF Firewall nel kernel di FreeBSD. La compilazione di un nuovo kernel che sostituirà quello presente all'atto dell'installazione è piuttosto semplice. **E' necessario avere i sorgenti installati.** [QUI](#) è spiegato bene come fare (l'unica differenza è l'installazione di subversion con 'pkg install'). In sostanza con il comando **uname -a** si ottengono le informazioni sulla versione esatta di FreeBSD installata e si sceglie in base al risultato il comando corrispondente. Se però la versione è RELEASE conviene usare il source della Release Engineering (releng) comprensivo delle patches, quindi il comando da dare sarà:

svn checkout <http://svn.freebsd.org/base/releng/14/> /usr/src

Il processo d'installazione durerà alquanto. Finito questo per prima cosa fare una copia del file di configurazione GENERIC che si trova in /usr/src/sys/amd64/conf chiamandolo diversamente (ad esempio MYKERNEL) e modificarlo secondo le opzioni necessarie. Poi procedere alla compilazione di un nuovo kernel [seguendo la guida ufficiale](#). La procedura è piuttosto semplice. FreeBSD dopo la compilazione mette il nuovo kernel nella cartella **/boot/kernel** e mantiene l'originale in **/boot/kernel.old**. Nel caso al riavvio qualcosa vada storto basta premere alla schermata di boot l'opzione 'escape to loader prompt' e dare il comando **boot kernel.old** come spiegato [QUI](#) alla sezione "The kernel does not boot". Di nuovo con il comando **uname -a** controlliamo se il kernel caricato sarà quello da noi compilato.

Paolo Di Stefano (update: 16 dicembre 2025)

www.paolodistefano.name

www.distefanopaolo.it

info@paolodistefano.name

E' ovviamente permessa la riproduzione parziale o totale; gradita la citazione della fonte