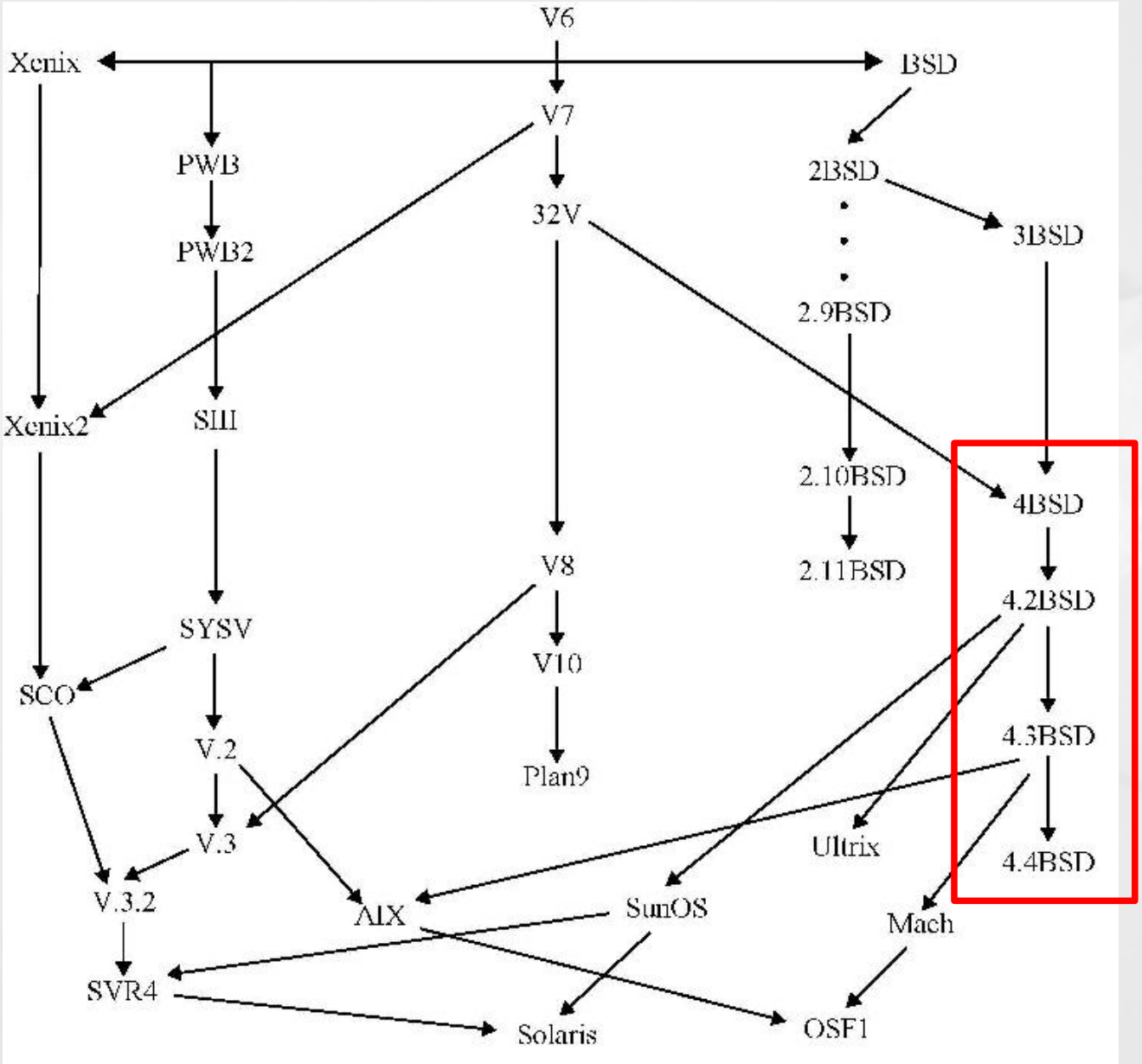
Michał Jaszczyk
Krzysztof Jakubowski
Krzysztof Kostałkowicz

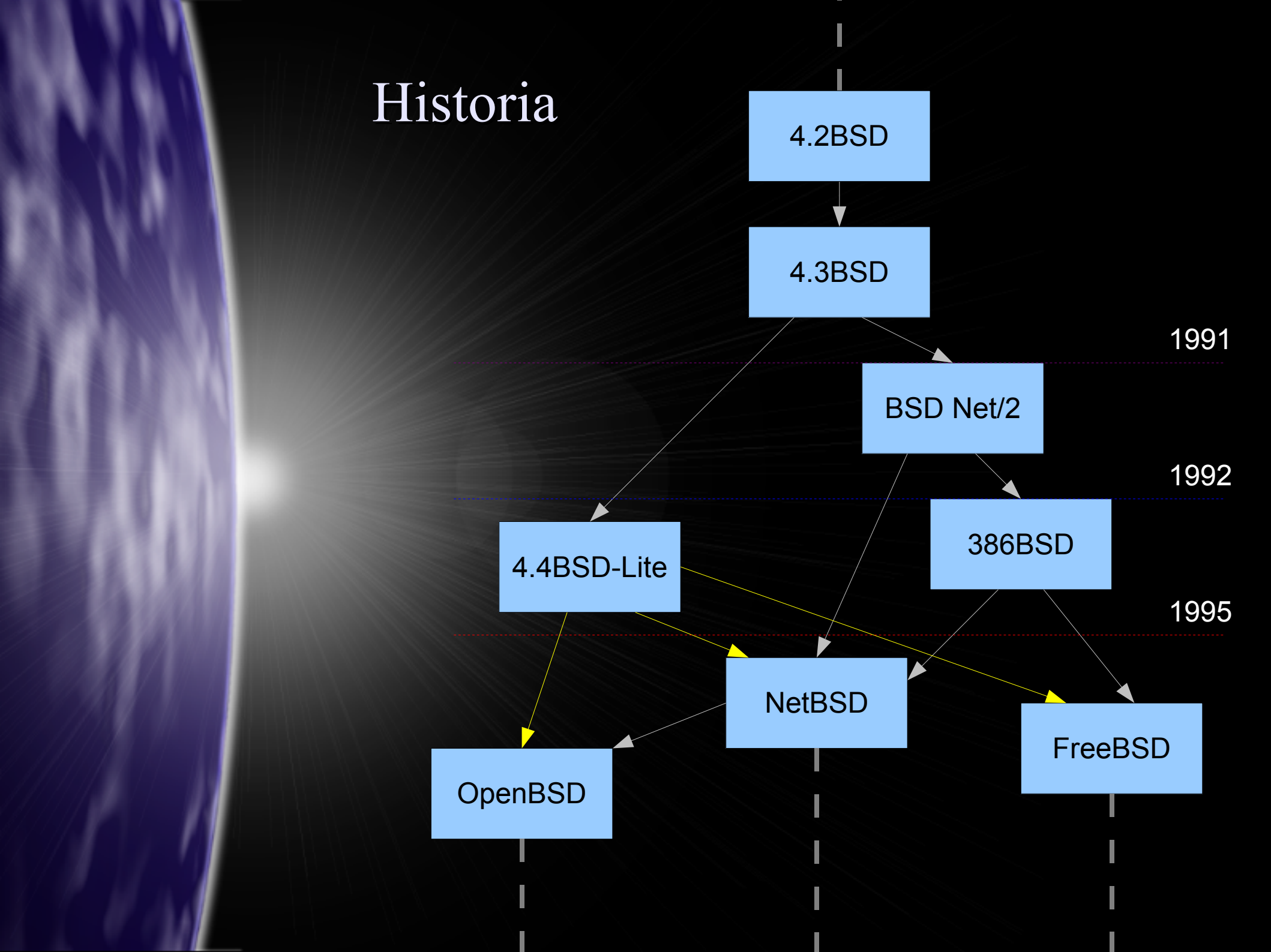
Berkeley Software Distribution

- System Unix-like
- Istnieje od lat 1970-tych
- Zapoczątkowany przez Uniwersytet Kalifornijski w Berkeley (studenci!)
- Architektury: PDP-11, VAX
- Alternatywna (w porównaniu z System V) gałąź rozwoju Unixa
- Główne nabytki: *(1BSD)* ex, *(2BSD)* vi, csh, *(4BSD)* delivermail (przodek sendmail'a), libcurses, *(4.2BSD)* sockets, FFS, daemon
- Wykorzystany przez np. Microsoft, Apple, Sun

Historia (przypomnienie)



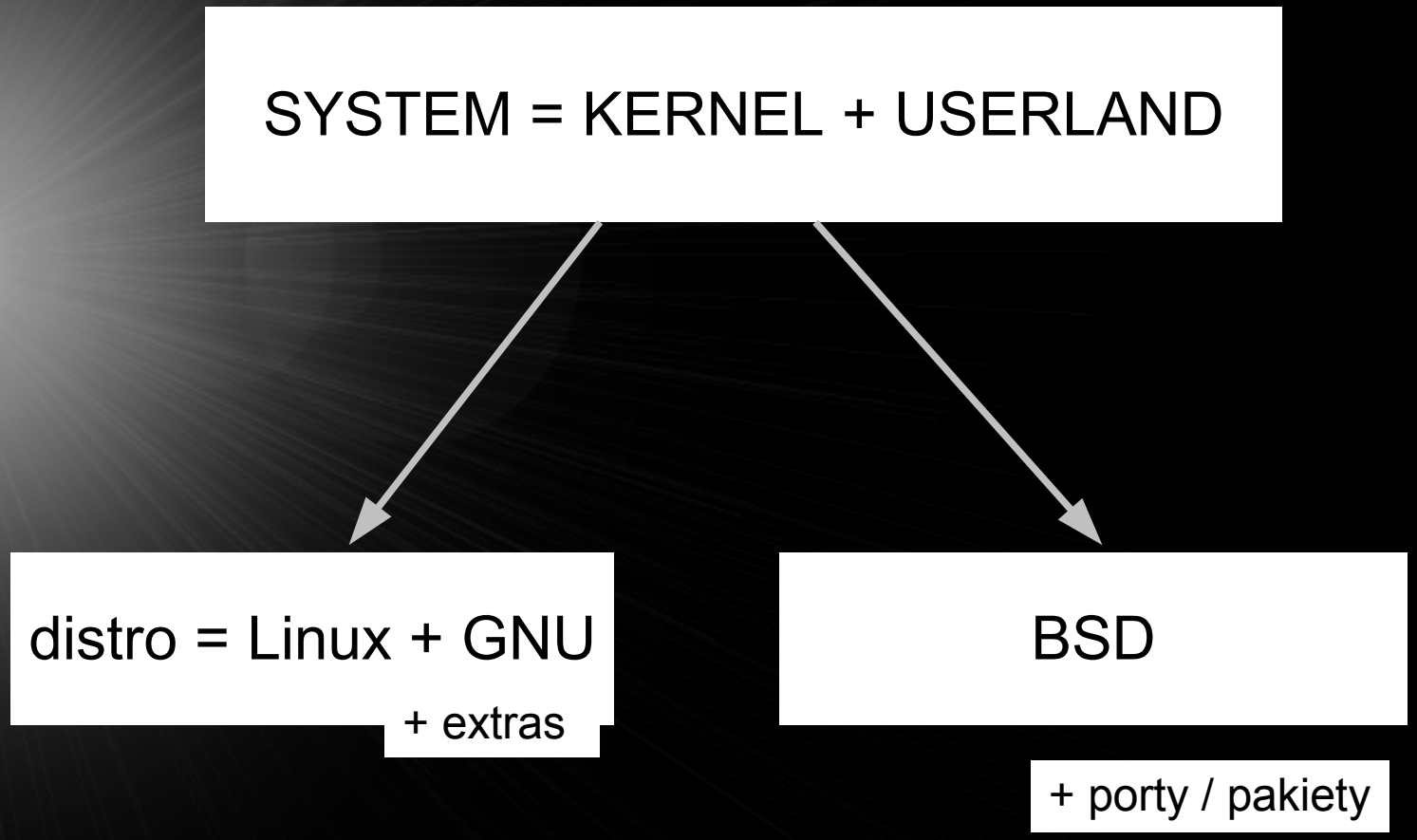
Historia


4.2BSD
4.3BSD
1991
BSD Net/2
1992
386BSD
4.4BSD-Lite
1995
NetBSD
FreeBSD
OpenBSD

Licencja BSD

- Dystrybucje źródłowe
  muszą zachowywać kopię licencji
- Dystrybucje binarne
  muszą odtwarzać kopię licencji w dokumentacji
- Advertising clause
  This product includes software developed by the University of California, Berkeley and its contributors.
  (w materiałach reklamowych)
- Promowanie produktu
  nie można powoływać się na autorów oryginału bez ich pisemnej zgody

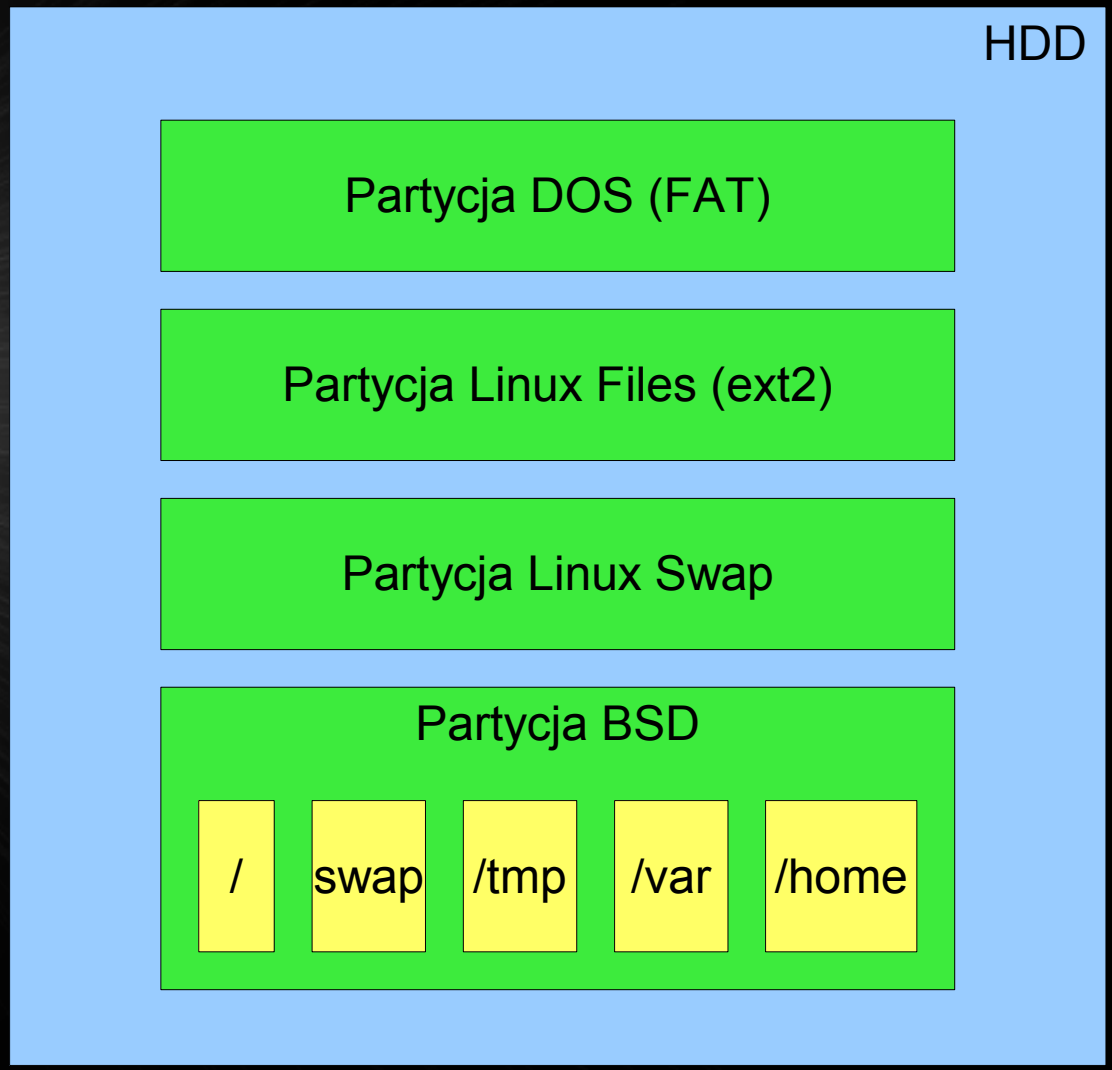
Co to jest system?



Wybrane cechy systemów

- Wieloplatformowość
- Rozwijane przy użyciu CVS
- Dobra dokumentacja (man pages)
- Własny system plików: FFS (UFS)
- Niekompatybilność licencji z GNU
- X11R6 (X.Org)
- Własny format binarny programów (ELF)
- Zgodność binarna z innymi Unixami (Linux)
- disklabel

disklabel


HDD
Partycja DOS (FAT)
Partycja Linux Files (ext2)
Partycja Linux Swap
Partycja BSD
/
swap
/tmp
/var
/home


NetBSD®


NetBSD

Główne cele projektu

- Portowalny system, działający na wielu platformach
- Tworzenie dobrze zaprojektowanego, stabilnego systemu BSD
- Kompatybilność z innymi systemami, zgodność ze standardami

Portowalny system, działający na wielu platformach sprzętowych

- Wyraźne oddzielenie kodu zależnego od platformy (MD) od kodu niezależnego (MI)
- Zestaw narzędzi w pełni obsługujących cross-kompilację
- Motto "Of course it runs NetBSD"; obsługa 53 różnych architektur

Tworzenie dobrze zaprojektowanego, stabilnego systemu BSD

- Wysoka jakość kodu
  - Portowalność wymusza dobrą jakość kodu
  - Motto "It doesn't work unless it's right"
- Szybkość działania poprzez porządny design, a nie drobne optymalizacje poszczególnych modułów

Kompatybilność z innymi systemami, zgodność ze standardami

- Emulacja binarek
  - Emulacja Linuksa; większość aplikacji działa bez problemu (np. Quake3, Opera, StarOffice, Matlab)
  - Emulacja innych Unixów, Windows przez wine
- Kompatybilność kodu (POSIX, XPG/SUS)
- Zawiera Xfree86 w wersji 6.4
- Obsługa wielu różnych systemów plików
- Obsługa wielu różnych protokołów sieciowych

Historia

- Założyciele: Chris Demetriou, Theo de Raadt, Adam Glass and Charles Hannum
- Mar. 1993 – pierwszy release (NetBSD 0.8, 1 platforma, bazuje na 386BSD)
- Paź. 1994 – pierwsza wersja multiplatformowa (NetBSD 1.0, 4 platformy)
- 1994 - Wykopanie z projektu Theo de Raadta :P
- Gru. 2004 – NetBSD 2.0, obsługuje 48 platform
- Aktualna stabilna wersja: NetBSD 3.1, działa na ~60 platformach

PKGSRC – kolekcja pakietów

- Aktualnie zawiera > 6000 pakietów
- Łatwość instalacji
- Dwa sposoby instalacji pakietów:
  - z binarek
  - ze źródeł

Emulacja linuxa

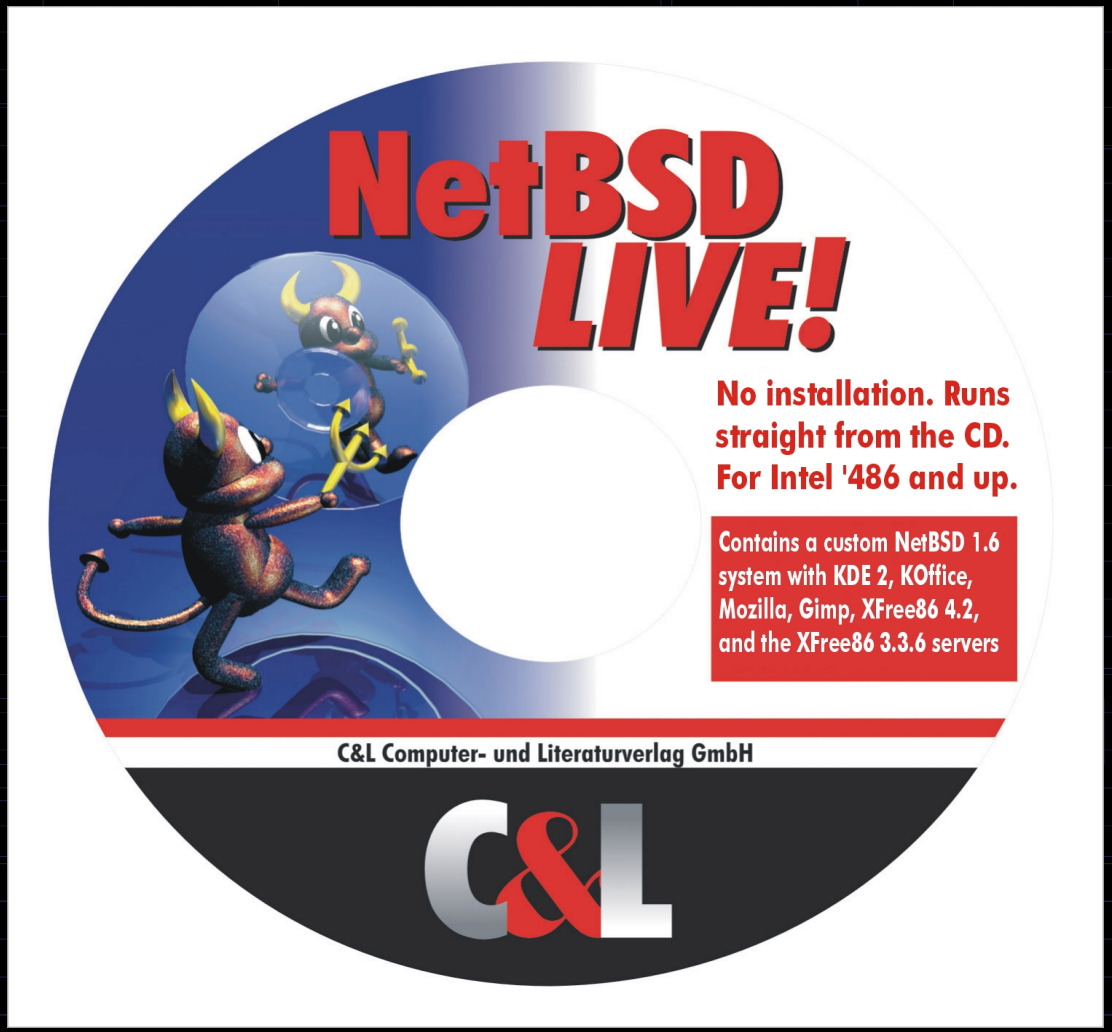
- Uruchamianie aplikacji linuxowych na kernelu NetBSD
  - Obsługa plików ELF
  - System plików /proc
  - Tłumaczenie wywołań systemowych
- Praktycznie brak strat na wydajności (1-2%)
- Ograniczenia

Zabawa z kernelem

- Łatwość kompilacji i konfiguracji
- Narzędzie do minimalizacji: adjustkernel
- Rozbudowany debugger, automatycznie uruchamiany w przypadku krytycznego błędu

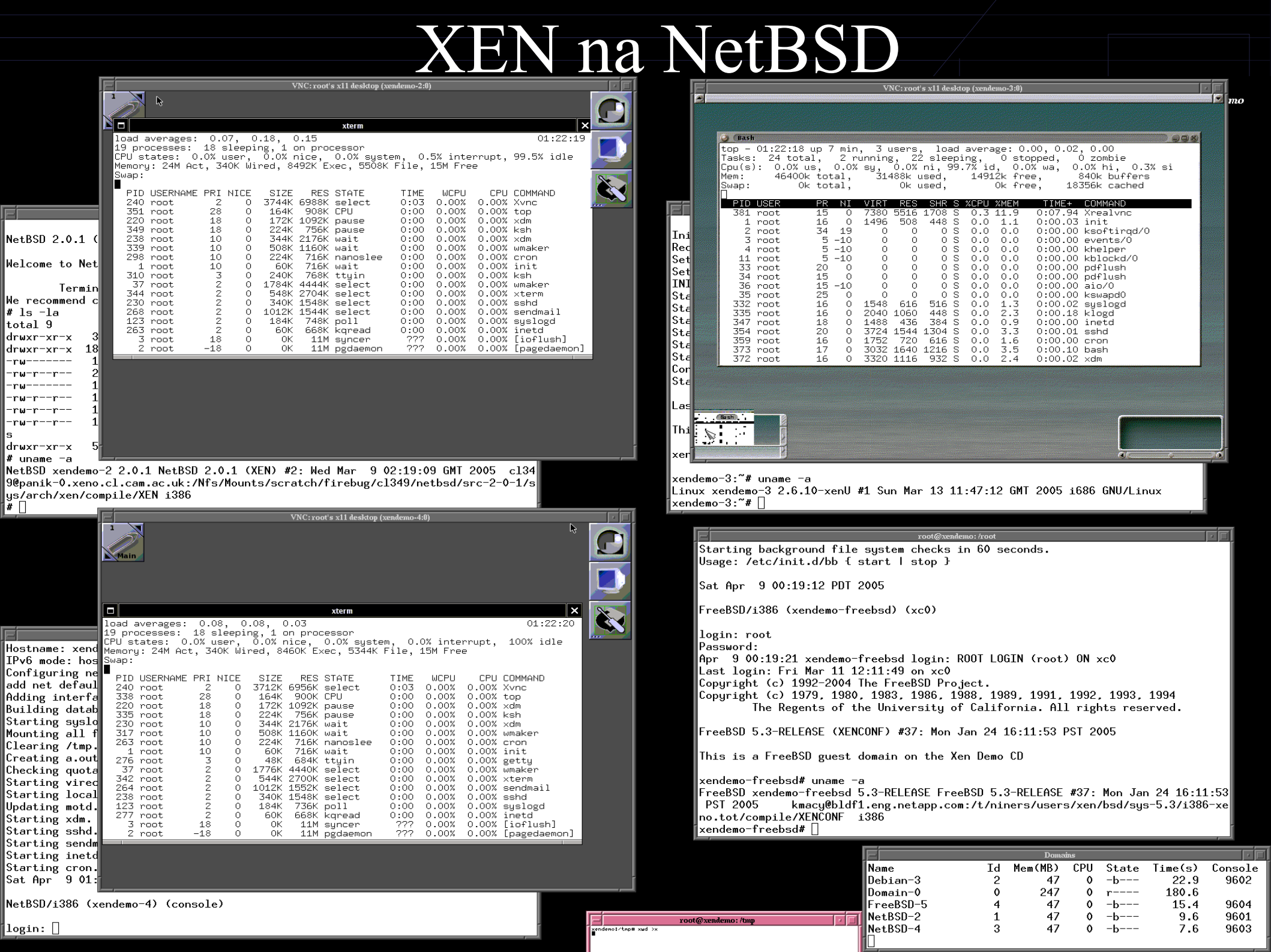
Dostępność

- Co kwartał nowe wydanie stabilne
- Minimalny system z podstawowymi pakietami dostępny w postaci obrazu iso
- Dostepne LiveCd (NetBSD Live! CD 2007)

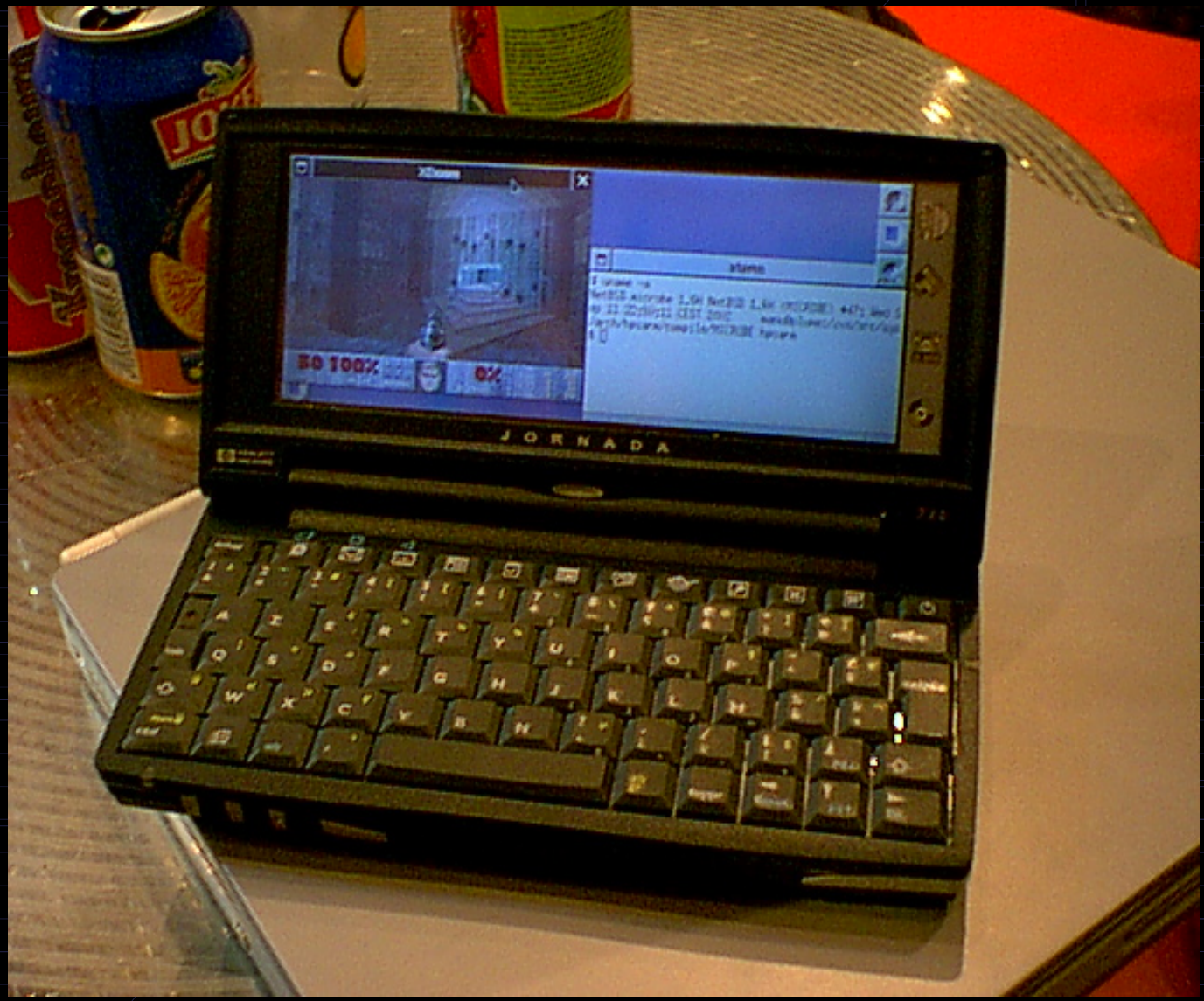


NetBSD działający na tosterze przerobionym na odtwarzacz mp3

XEN na NetBSD



Doom działający na NetBSD na palmtopie HP Jornada 720 PDA



Baldur's Gate na NetBSD przez wine

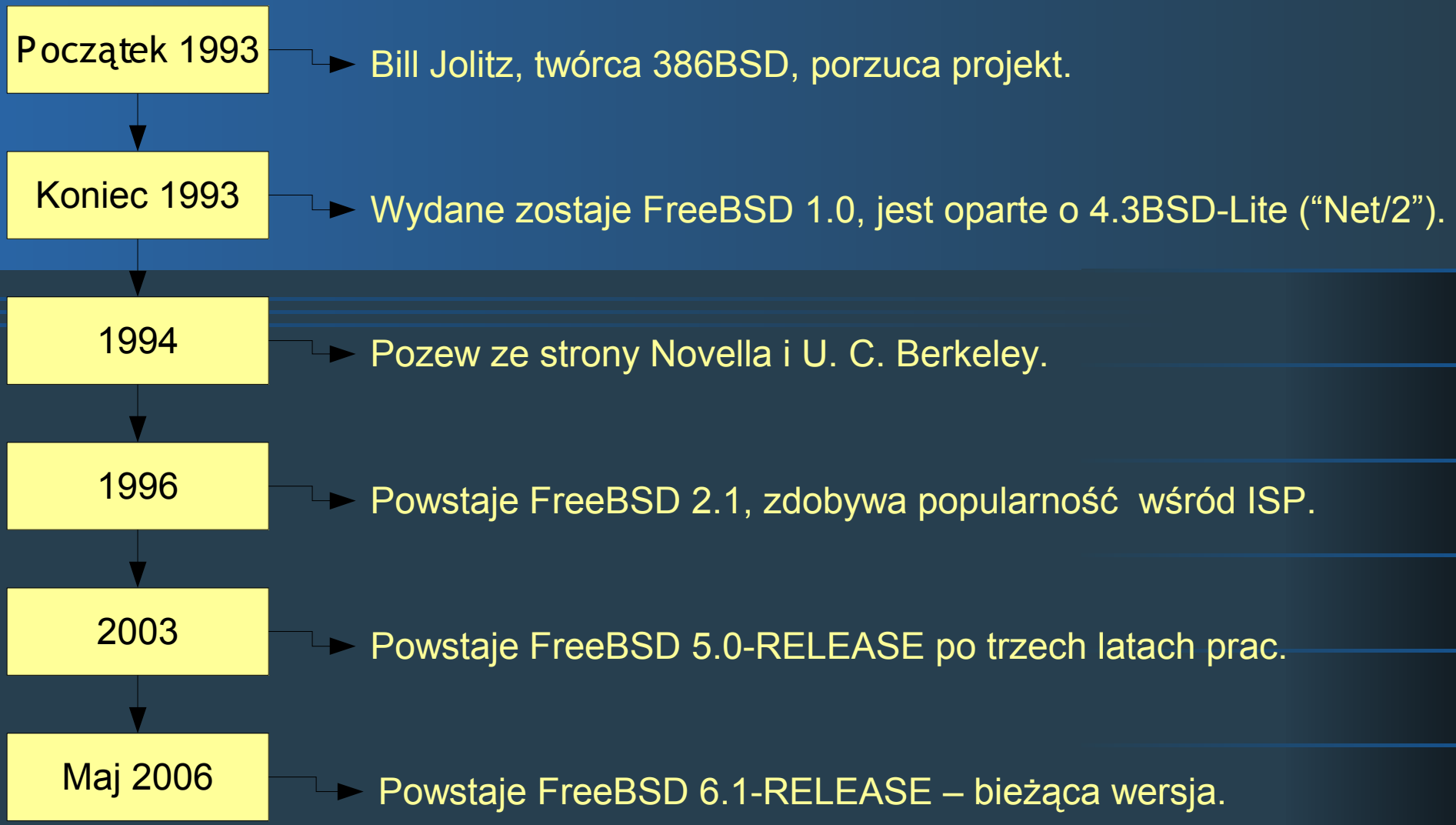
xterm

```
euler$ uname -a
NetBSD euler.wickles.net 1.6.1 NetBSD 1.6.1 (GENERIC) #0: Tue Apr  8 12:05:52 UT
C 2003     autobuild@tgm.daemon.org:/autobuild/netbsd-1-6/i386/OBJ/autobuild/net
bsd-1-6/src/sys/arch/i386/compile/GENERIC i386
euler$ import -window root -delay 10 bg.jpg
```

FreeBSD

FreeBSD®

Historia


Początek 1993
Bill Jolitz, twórca 386BSD, porzuca projekt.
Koniec 1993
Wydane zostaje FreeBSD 1.0, jest oparte o 4.3BSD-Lite (“Net/2”).
1994
Pozew ze strony Novella i U. C. Berkeley.
1996
Powstaje FreeBSD 2.1, zdobywa popularność wśród ISP.
2003
Powstaje FreeBSD 5.0-RELEASE po trzech latach prac.
Maj 2006
Powstaje FreeBSD 6.1-RELEASE – bieżąca wersja.

Idee projektu

- FreeBSD jest dla każdego.
- FreeBSD jest do wszystkiego.
- Dzięki temu jest z niego największy pożytek.
- Nie ma żadnych kruczków prawnych.

FreeBSD 6.1-RELEASE

Moduły jądra

- Jako jedyny wykorzystuje moduły.
- Można podmieniać syscalle.
- Można dodawać systemy plików albo podmieniać funkcje w istniejących.
- Można tworzyć nowe urządzenia.
- Można modyfikować dane jądra.

Moduły - użytkownik

- kldload – ładuje moduł do jądra
- kldunload – usuwa moduł z jądra
- kldstat – wypisuje załadowane moduły

Moduły - programista cz.1

```
static d_open_t      echo_open;
static d_close_t     echo_close;
static d_read_t      echo_read;
static d_write_t     echo_write;

static struct cdevsw echo_cdevsw = {
    .d_open = echo_open,
    .d_close = echo_close,
    .d_name = "echo",
    .d_read = echo_read,
    .d_write = echo_write,
    .d_version = D_VERSION
};
```

Moduły - programista cz.2

```
static int echo_loader(struct module *m, int what, void *arg)
{
    int err = 0;

    switch (what) {
    case MOD_LOAD:
        echo_dev = make_dev(&echo_cdevsw,
            0,
            UID_ROOT,
            GID_WHEEL,
            0600,
            "echo");
        break;
    case MOD_UNLOAD:
        destroy_dev(echo_dev);
        break;
    default:
        err = EINVAL;
        break;
    }
    return(err);
}
```

Moduły - programista cz.3

```
DEV_MODULE(echo,echo_loader,NULL);
```

```
SRCS=echodev.c
KMOD=echodev

.include <bsd.kmod.mk>
```

Jail

- Metoda wirtualizacji
- Zwiększa bezpieczeństwo i ułatwia administrację
- Wsparcie dla wirtualnych sieci
- Przeznaczone do uruchamiana serwerów

Jail - zabezpieczenia

- chroot do wybranego katalogu - muszą tam być wszystkie binarki
- Ograniczenie widzenia innych procesów i interakcji z nimi (np. IPC)
- Ograniczenie dostępu do urządzeń
- Ograniczenie dostępu do wywołań systemowych i sysctl
- Dostęp do tylko jednego numeru IP

Jail - uruchamianie

- Przygotowanie struktury katalogów (najlepiej zainstalować całe FreeBSD)
- Przygotowanie połączeń sieciowych - IP oraz hostname
- Wywołanie polecenia jail
- Jail jako usługa systemowa
- Wiele więzień? Użyj ezjail!

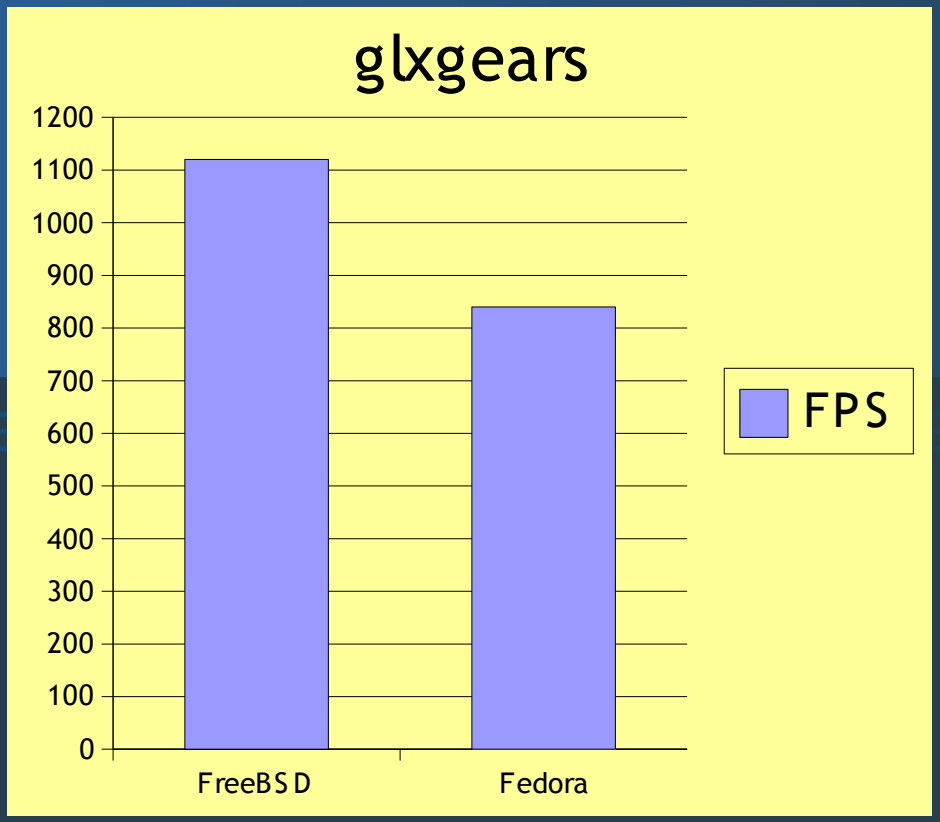
Wydajność

vs

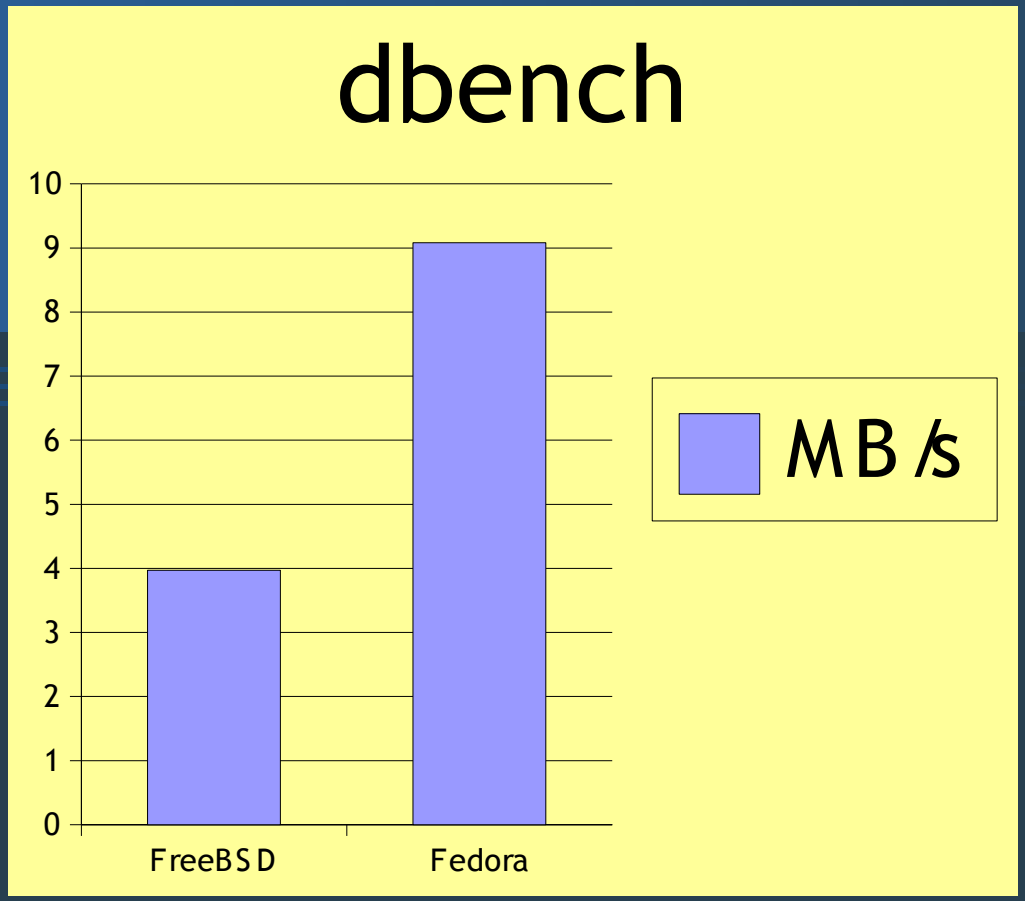
Środowisko testowe

- Fujitsu-Siemens Lifebook S7020
- Procesor: Intel Centrino 1,8GHz
- Pamięć: 512MB RAM
- Grafika: Intel 915GM
- Dysk: SATA100 80GB

Test 1 - glxgears



Test 2 - dbench



Test 3 - libMicro

OpenBSD

Jaki to system?

- Darmowy
- Wieloplatformowy
- Bazowany na 4.4BSD
- Bezpieczny
- Przenośny
- Zgodny ze standardami
- Dobrze udokumentowany

Only one remote hole in the default install, in more than 10 years!

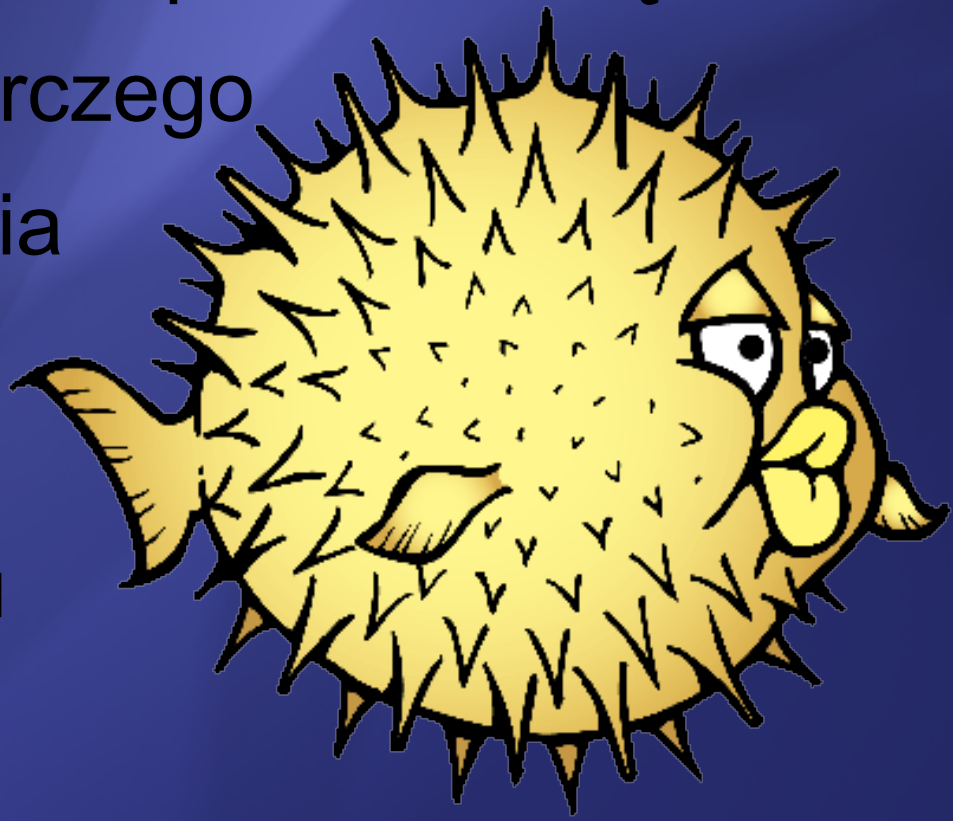
www.openbsd.org

Historia

- Grudzień 1994 – **Theo de Raadt**, współzałożyciel NetBSD traci dostęp do repozytorium
- Październik 1995 – początek projektu (na bazie NetBSD 1.0)
- Lipiec 1996 – OpenBSD 1.2
- Październik 1996 – OpenBSD 2.0 od tej pory regularne wydania
- Listopad 2006 – OpenBSD 4.0

Cele projektu

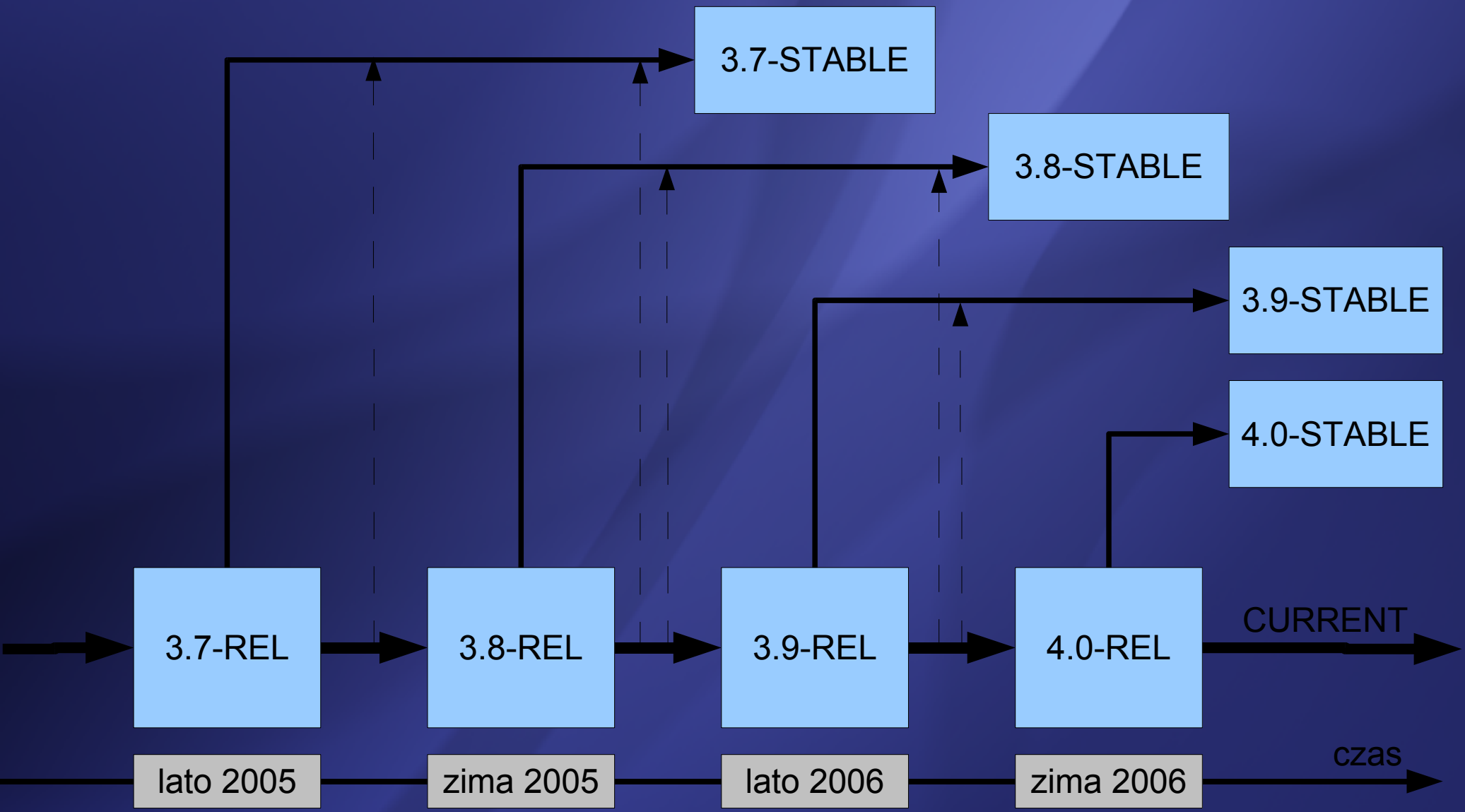
- Bezpieczeństwo i szybkie naprawianie błędów
- Jawność procesu wytwórczego
- Zintegrowana kryptografia
- Zgodność z normami (ANSI, POSIX)
- Niezależność od sprzętu
- Wydanie co 6 miesięcy
- Tylko naprawdę wolne oprogramowanie!



Życie projektu

- Szefem jest **Theo de Raadt**
- Rozwijany przez ochotników
- Zarabia na sprzedawaniu płyt, koszulek oraz dotacjach

Cykl wytwórczy



Instalacja systemu

- Wyłącznie w trybie tekstowym
- Ładowanie instalatora z dyskietki, płytki CD-ROM, dysku twardego (/bsd.rd), sieci (PXE)
- Ściąganie dystrybucji z płytki CD-ROM, dysku twardego, FTP, HTTP, NFS
- Jądra: bsd bsd.rd bsd.mp
- Paczki: base, etc, comp, man, misc, game, x*
- Możliwość dostosowania systemu (site40.tgz)

Ładowanie systemu 1

- Pierwszy sektor dysku twardego - MBR
- Pierwszy sektor partycji OpenBSD (0xA6) – PBR w PBR jest stage1-boot (biosboot)
- Ładuje stage2-boot (/boot) (wie z jakich sektorów) /boot to interaktywny program
- Domyślnie po 5 sek. wykonuje: boot /bsd
- Wczytuje obraz jądra z systemu plików do pamięci
- Oddaje sterowanie do jądra

Ładowanie systemu 2

- Wykrywanie urządzeń (UKC)
- Montowanie systemu plików /
- Odpalenie /sbin/init (inny niż w System V)
- W trybie single-user uruchomienie shella
- Wykonanie /etc/rc
  rc.conf[.local] rc.securelevel rc.local
- Obsługa terminali (/etc/ttys)
  czyli odpalenie /usr/libexec/getty na aktywnych
- Czekanie na sygnały, pilnowanie terminali

Programowanie w C

Linux

```
#include <stdio.h>

int main()
{
        printf("hello, world!\n");
        return 0;
}
```

OpenBSD

```
#include <stdio.h>

int main()
{
        printf("hello, world!\n");
        return 0;
}
```

Progr. niskopodłogowe

Linux

```
SYS_exit = 1
SYS_write = 4
STDOUT = 1

        .data
buf:
        .ascii "yo man!\n"
        buflen = . - buf

        .text
.global _start



_start:
        movl    $buflen, %edx
        movl    $buf, %ecx
        movl    $STDOUT, %ebx
        movl    $SYS_write, %eax
        int     $0x80


        movl    $0, %ebx
        movl    $SYS_exit, %eax
        int     $0x80
```

OpenBSD

```
SYS_exit = 1
SYS_write = 4
STDOUT = 1

        .data
buf:
        .ascii "yo man!\n"
        buflen = . - buf

        .text
.global _start

_syscall:
        int     $0x80
        ret

_start:
        pushl   $buflen
        pushl   $buf
        pushl   $STDOUT
        movl    $SYS_write, %eax
        call    _syscall
        addl    $12, %esp

        pushl   $0
        movl    $SYS_exit, %eax
        call    _syscall
```

Progr. niskopodłogowe

Linux

```
SYS_exit = 1
SYS_write = 4
STDOUT = 1

        .data
buf:
        .ascii "yo man!\n"
        buflen = . - buf

        .text
.global _start




_start:
        movl    $buflen, %edx
        movl    $buf, %ecx
        movl    $STDOUT, %ebx
        movl    $SYS_write, %eax
        int     $0x80


        movl    $0, %ebx
        movl    $SYS_exit, %eax
        int     $0x80
```

OpenBSD

```
SYS_exit = 1
SYS_write = 4
STDOUT = 1

        .data
buf:
        .ascii "yo man!\n"
        buflen = . - buf

        .text
.global _start

_syscall:
        int     $0x80
        ret

_start:
        pushl   $buflen
        pushl   $buf
        pushl   $STDOUT
        movl    $SYS_write, %eax
        call    _syscall
        addl    $12, %esp

        pushl   $0
        movl    $SYS_exit, %eax
        call    _syscall
```

Ale to nie wszystko!

openbsd$ as -o hello.o hello.s

openbsd$ ld -o hello hello.o

openbsd$ ./hello

./hello: Operation not permitted.

openbsd$??? _

Ale to nie wszystko!

openbsd$ as -o hello.o hello.s

openbsd$ ld -o hello hello.o

openbsd$ ./hello

./hello: Operation not permitted.

openbsd$???

sh: ???: command not found

openbsd$ _

Czego (jeszcze) nie wiemy?

OpenBSD dodatkowo taguje binarki

openbsd$ file hello

hello: ELF 32-bit LSB executable, Intel 80386, version 1, statically linked, not stripped

openbsd$ file hello-tagged

hello-tagged: ELF 32-bit LSB executable, Intel 80386, version 1, for OpenBSD, statically linked, not stripped

openbsd$ ./hello-tagged

yo man!

openbsd$ _

```
.section ".note.openbsd.ident", "a"
.p2align 2
        .long 8
        .long 4
        .long 1
        .ascii "OpenBSD\0"
        .long 0
```

Inne ciekawe niespodzianki

TIME(2) Linux Programmer's Manual

NAME

time - get time in seconds

SYNOPSIS

#include <time.h>

time_t time(time_t *t);

DESCRIPTION

time returns the time since the Epoch (00:00:00 UTC, January 1, 1970), measured in seconds.

If t is non-NULL, the return value is also stored in the memory pointed to by t.

TIME(3) OpenBSD Programmer's Manual

NAME

time - get time of day

SYNOPSIS

#include <time.h>

time_t
time(time_t *tloc);

DESCRIPTION

The time() function returns the value of time in seconds since 0 hours, 0 minutes, 0 seconds, January 1, 1970, Coordinated Universal Time (UTC).

A copy of the time value may be saved to the area indicated by the pointer tloc. If tloc is a null pointer, no value is stored.

strace

```
linux$ strace ./hello
execve("./hello", ["./hello"], [/* 30 vars */]) = 0
write(1, "yo man!\n", 8yo man!
)               = 8
_exit(0)                        = ?
```

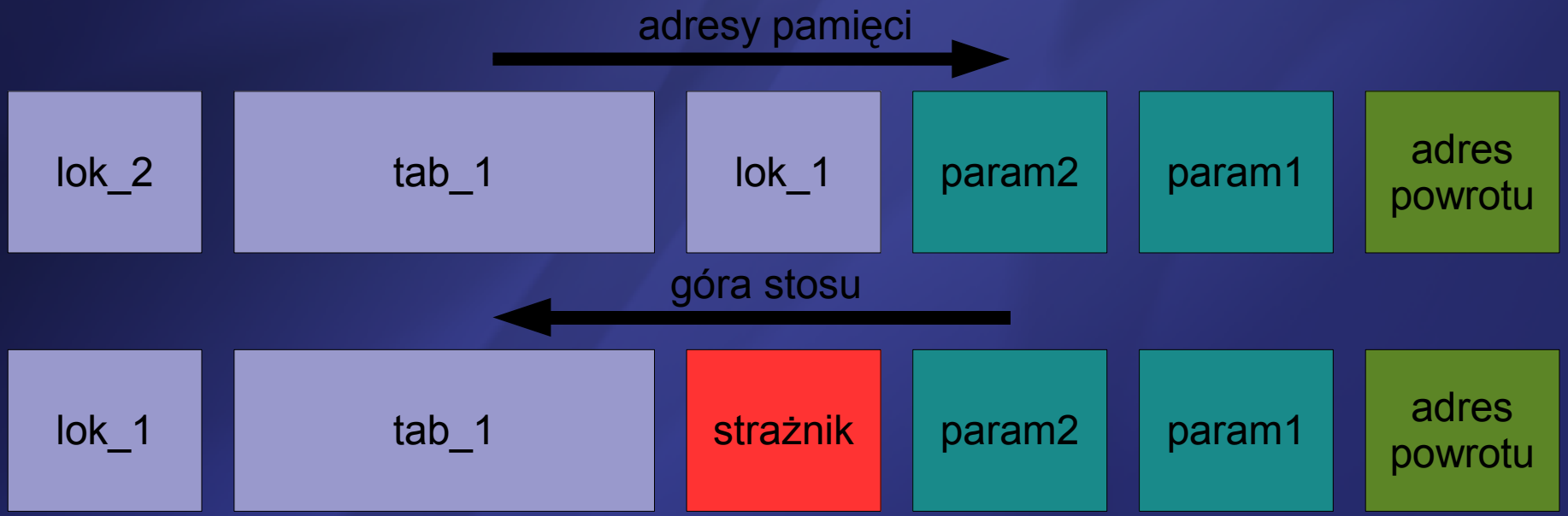
ktrace

```
openbsd$ ktrace ./hello
yo man!
openbsd$ kdump -f ktrace.out
 17218 ktrace   RET   ktrace 0
 17218 ktrace   CALL
   execve(0xcf7e7eef,0xcf7e7d88,0xcf7e7d90)
 17218 ktrace   NAMI  "./hello"
 17218 hello EMUL  "native"
 17218 hello RET   execve 0
 17218 hello CALL  write(0x1,0x3c000000,0x8)
 17218 hello GIO   fd 1 wrote 8 bytes
       "yo man!
       "
 17218 hello RET   write 8
 17218 hello CALL  exit(0)
```

?trace

Propolice

- Domyślnie włączone
- Zabezpiecza przed zniszczeniem stosu
- Umieszczenie strażnika: terminator / losowy
- Niezgodność kończy proces



Więcej bezpieczeństwa

- strlcpy(), strlcat()
- losowe PIDy
- separacja uprawnień: chroot, więcej UIDów
- dodatkowa ochrona pamięci
- W ^ X
  - strona procesu tylko do pisania / wykonywania
  - na i386 emulacja, natywnie na amd64, sparc
  - od OpenBSD 3.3 – pierwszy system obsługujący
  - inne wersje: PaX, Exec Shield, DEP

Bezpieczeństwo - inne

- ld ostrzega przed niebezpiecznymi funkcjami
- ograniczony dostęp do /dev/kmem
- ograniczony dostęp do pamięci wideo – xf86(4)
- flagowanie plików – chflags(8)
- domyślnie nie działa *ctrl-alt-del*
- securelevel – poziom bezpieczeństwa systemu:
  - -1 – bez ograniczeń (permanentny)
  - 0 – bez ograniczeń (awansuje do 1)
  - 1 – standardowy (nie można z niego obniżyć)
  - 2 – przegięty (np. nie można cofać czasu)

Narzędzia w base (w portach)

- (k)sh, csh (bash, tcsh)
- nvi (vim, elvis)
- ftp (wget)
- tar, gzip, pax (bzip2)
- nawk (gawk)
- lynx (links)
- less
- perl
- cvs
- make (gmake)
- gcc, g++
- flex, yacc
- gdb

Coś dla superużytkownika

- quota
- pkg_add, pkg_info, ... package(5)
- sudo
- useradd, ...
- wsmoused
- netstat ,tcpdump, nc, route, ifconfig, ping(6), traceroute(6)
- httpd (Apache 1.3.29+)
- ftpd (własna wersja)
- sshd
- sendmail, popa3d
- dhclient, dhcpd
- bind
- lpd
- NFS

Urządzenia

- Klasyczny /dev (standardowy MAKEDEV)
- Dyski twarde: wd0, wd1, ... (kolejność wykrywania)
- Dyski SCSI: sd0, ... (też pamięci USB)
- Napędy optyczne: cd0, ...
- Stacje dyskietek: fd0, ...
- Mysz: wsmouse (PS/2 i USB)
- Wirtualne terminale: ttyC0, ...
- Dźwięk: audio (interfejs Sun Audio, obsługa OSS)
- Porty szeregowe: tty00, cua00, ...
- Loopback device: vnd0, ... (w Linuxie /dev/loop0)

Urządzenia blokowe

- Dostęp wyłącznie poprzez **disklabel**
- Możliwe 16 partycji a-p, z wyjątkami:
  - a – partycja /
  - b – swap
  - c – całe urządzenie
- Partycja ma: offset (bezwzględny), rozmiar, typ
- W przypadku braku struktury jądro generuje domyślną instancję
- Każde urządzenie blokowe BLKx ma interfejs znakowy rBLKx

Obsługa napędów CD-ROM

openbsd# dmesg | grep ...

atapiscsi0 at pciide0 channel 0 drive 1

scsibus0 at atapiscsi0: 2 targets

cd0 at scsibus0 targ 0 lun 0: <TSSTcorp, CD/DVDW TS-L532A, TI51>
SCSI0 5/cdrom removable

cd0(pciide0:0:1): using PIO mode 4, Ultra-DMA mode 2

openbsd# mount [-t cd9660/udf] /dev/cd0c /mnt/cdrom

openbsd# umount /mnt/cdrom && eject /dev/rcd0c

openbsd# pkg_add cdrtools-2.01.tgz

openbsd# cdrecord dev=/dev/rcd0c -checkdrive

...

Crypto

- OpenSSH
- Kerberos, IPsec
- Hash: MD5, SHA1, RIPEMD-160
- Cipher: DES, 3DES, Blowfish
- PRNG: prandom, urandom, srandom, arandom
- Wsparcie dla sprzętu (i8x0, VIA C3)
- Integracja z systemem (/etc/master.passwd)

Cool solutions

- OpenSSH
- OpenNTPD
- OpenBGPD
- OpenCVS
- pf
- CARP
- spamd

Sieci

- IPv6
- IPsec (VPN)
- tunelling: GIF, GRE
- OpenBGPD (BGP, OSPF)
- PPP(oE,oA)
- CARP
- pf
- spamd

Packet Filter

- Filtrowanie pakietów
- Translacja pakietów (NAT, BINAT, RDR)
- Normalizacja pakietów (składanie fragmentów)
- Pasywne wykrywanie systemu operacyjnego
- Kontrola pasma (ALTQ)
- Ograniczanie obciążenia (zapobiega DOS)
- Autoryzacja authpf
- Integracja ze spamd
- Redundancja: CARP i/lub pfsync

Trochę multimediów

- Obsługa ISO-9660 (cd9660), UDF
- Odtwarzanie MP3 (mpg123, xmms)
- Nagrywanie CD (cdrecord), DVD (dvd+rw-tools)
- Filmy, strumienie (mplayer)
- OpenGL (ale bez akceleracji sprzętowej)
- Emulacja binarna Linuxa (też SVR4, FreeBSD, BSD/OS, SunOS, HP-UX)
- Java, Flash, Opera, OpenOffice

Zalety i wady systemu

+ Bezpieczeństwo
+ Chlujność
+ Dokumentacja
+ Przewidywalność
+ Sieciowość
+ Obsługa SMP
+ Emulacja Linuxa

- Nowy sprzęt
- Multimedialność
- Nowe technologie (ACPI, RAID)
- Nowe systemy plików (ext3, NTFS)
- i18n
- Wydajność

Podsumowanie

Fajny (bezpieczny) serwer...

...ale nie za ciekawa stacja robocza...

...a o laptopie lepiej wogóle zapomnieć

Systemagic (song31.mp3)

BSD fight buffer reign
Flowing blood in circuit vein
Quagmire, Hellfire,
    RAMhead Count
Puffy rip attacker out

Crackin' ze bathroom,
    Crackin' ze vault
Tale of the script, HEY!
    Secure by default

Can't fight the Systemagic
Über tragic
Can't fight the Systemagic

Sexty second, black cat struck
Breeding worm of crypto-suck
Hot rod box unt hunting wake
Vampire omellete, kitten cake

http://www.openbsd.org/lyrics.html

Crackin' ze boardroom,
    Crackin' ze vault
Rippin' ze bat, HEY!
    Secure by default

Chorus

Cybersluts vit undead guts
Transyl-viral coffin muck
Penguin lurking under bed
Puffy hoompa on your head

Crackin' ze bedroom,
    Crackin' ze vault
Crackin' ze whip, HEY!
    Secure by default
Crackin' ze bedroom,
    Crackin' ze vault
Crackin' ze whip, HEY!
    Secure by default

Chorus

Inne *BSD

- DragonFlyBSD
  - Wywodzi się od FreeBSD 4.8
  - Maskotka: ważka o imieniu Fred
  - Łatwy w użytkowaniu system dla systemów wieloprocesorowych
- PC-BSD, DesktopBSD
  - Pochodzą od FreeBSD
  - Łatwe w użytkowaniu

MacOSX (MacOS 10)

- Bazuje na Darwinie (który bazuje na FreeBSD)
- Całkowicie odmienna budowa niż poprzednie wersje MacOS
- Łatwość adaptacji oprogramowania dla systemów BSD
- Struktura systemu MacOSX:

Aqua		
Carbon	Cocoa	Java
Quartz	OpenGL	QuickTime
Darwin		

Konkluzja

To BSD or not to BSD?

Dziękujemy za uwagę !!!