The NetBSD Project

"Of course it runs NetBSD"

OSC 2012 北海道

たまには謎マシン抜きの話をしてみよう

NetBSDのクロスビルドのしくみとインストール済みLive Imageの作成

Izumi Tsutsui

tsutsui@NetBSD.org

NetBSDの特徴

- **特定の機種に依存しない設計**
  各種ソースが Android や MINIX に流用されたり
- **ものすごい数の機種への移植**
  55〜58機種くらい(?)を一応いまだにサポート
- **クロスコンパイル開発環境**
  NetBSDを使っていなくてもNetBSD開発が可能

クロスコンパイルとは

■「セルフコンパイル」

バイナリを作るマシン＝バイナリを実行するマシン

⇒バイナリを作ったマシンでそのまま実行

■「クロスコンパイル」

バイナリを作るマシン≠バイナリを実行するマシン

⇒作ったバイナリを実行するマシンに転送して実行

クロスコンパイル用語

「ターゲット」

⇒バイナリを実行するマシン

「ホスト」

⇒バイナリを作成するマシン

「ツールチェイン」

⇒コンパイラ、アセンブラなどの
クロスビルドバイナリ作成用プログラム一式

NetBSDとクロス開発環境

- NetBSD 1.6 (2002年) から公式にサポート

  OS標準の配布物だけで全部をクロスで作成可能

  - カーネル
  - ユーザーランドバイナリ
  - man ページ
  - リリース用配布バイナリー式 tar.gz
  - インストール用フロッピーイメージ
  - インストール用CD ISOイメージ　　(3.0以降)
  - インストール済み Live イメージ　　(6.0以降)

クロスビルド実行コマンド

- “build.sh” スクリプト

  面倒な設定不要なクロスビルド実行スクリプト

  “build.sh -U -m i386 tools”

  →i386用ツールチェインバイナリー式作成

  “build.sh -U -m i386 release”

  →i386 リリース用バイナリー式作成

  “build.sh -U -m i386 live-image”

  →i386 インストール済み Live イメージ作成

なぜクロスビルドなのか(1)

■ にわとり卵問題

「バイナリを実行するマシンでバイナリを作る」には
「バイナリを実行できる環境」が必要

⇒まだOSすら動いてないマシンはどうするの？

OS以前にハードから自作した場合は？？

当然 ターゲット≠ホスト のクロス開発が必須

なぜクロスビルドなのか(2)

■ 遅マシン対策

- **移植バブル期のターゲット**⇒現役引退ハード多数
  - お世辞にも速いとは言えない
  - PCのほうがよっぽど速い
- **NetBSD/news68k カーネルコンパイル時間**(当時)
  - NWS-1750D(MC68030/25MHz): **3時間20分**
  - 自作PC (AMD K6-2/350MHz): **5分**

……苦労してでも5分で作れる環境用意しますよね

なぜクロスビルドなのか(3)

■ 全自動ビルド化

・セルフビルド時代でもリリース作成は自動化済み

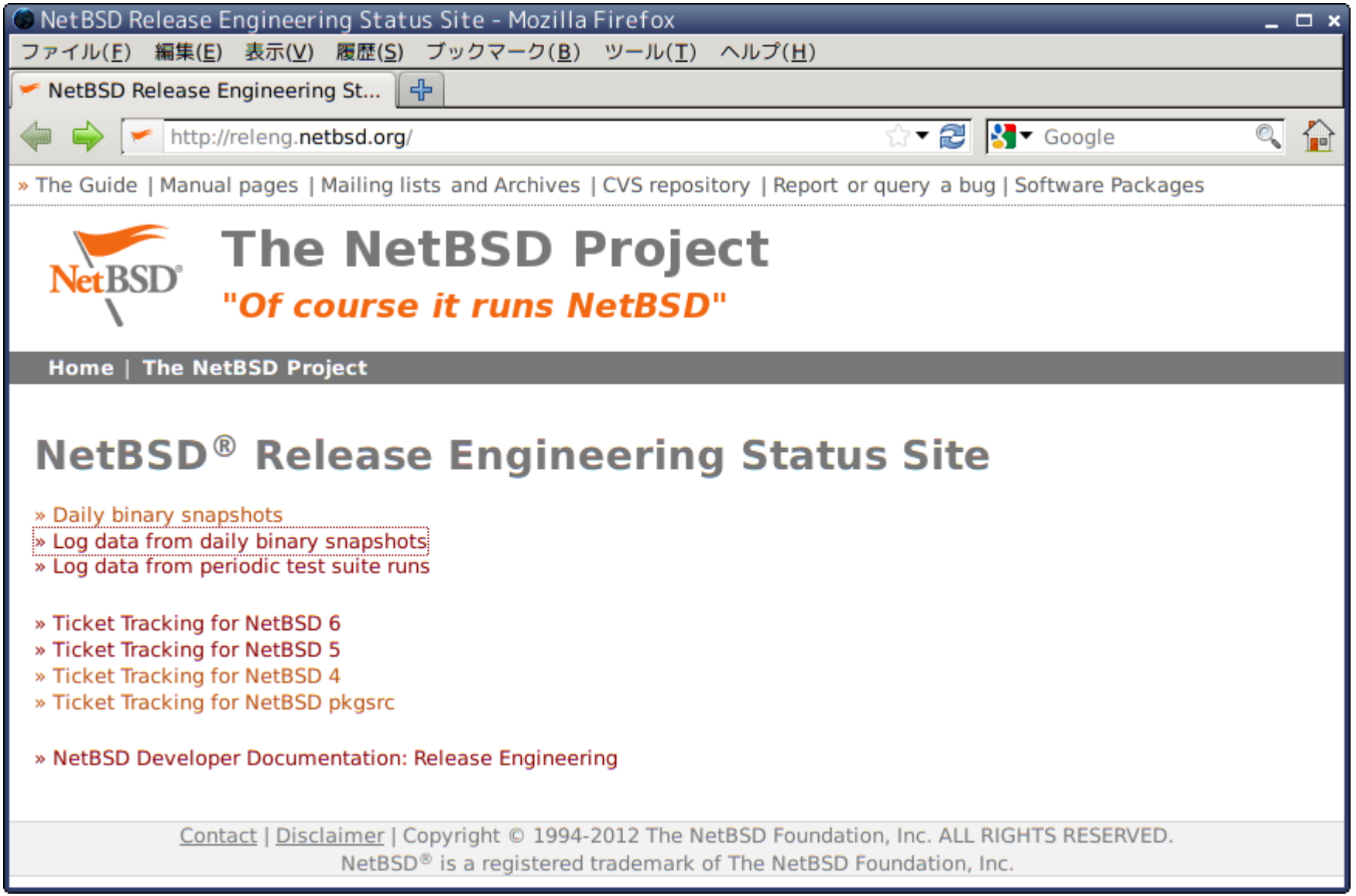
・NetBSD/news68k のリリース作成に1週間とか

⇒速いマシンでやれば全機種毎日テスト可能?

・ツールチェイン作成～ビルド～リリース一式作成を全自動で実行して結果チェック

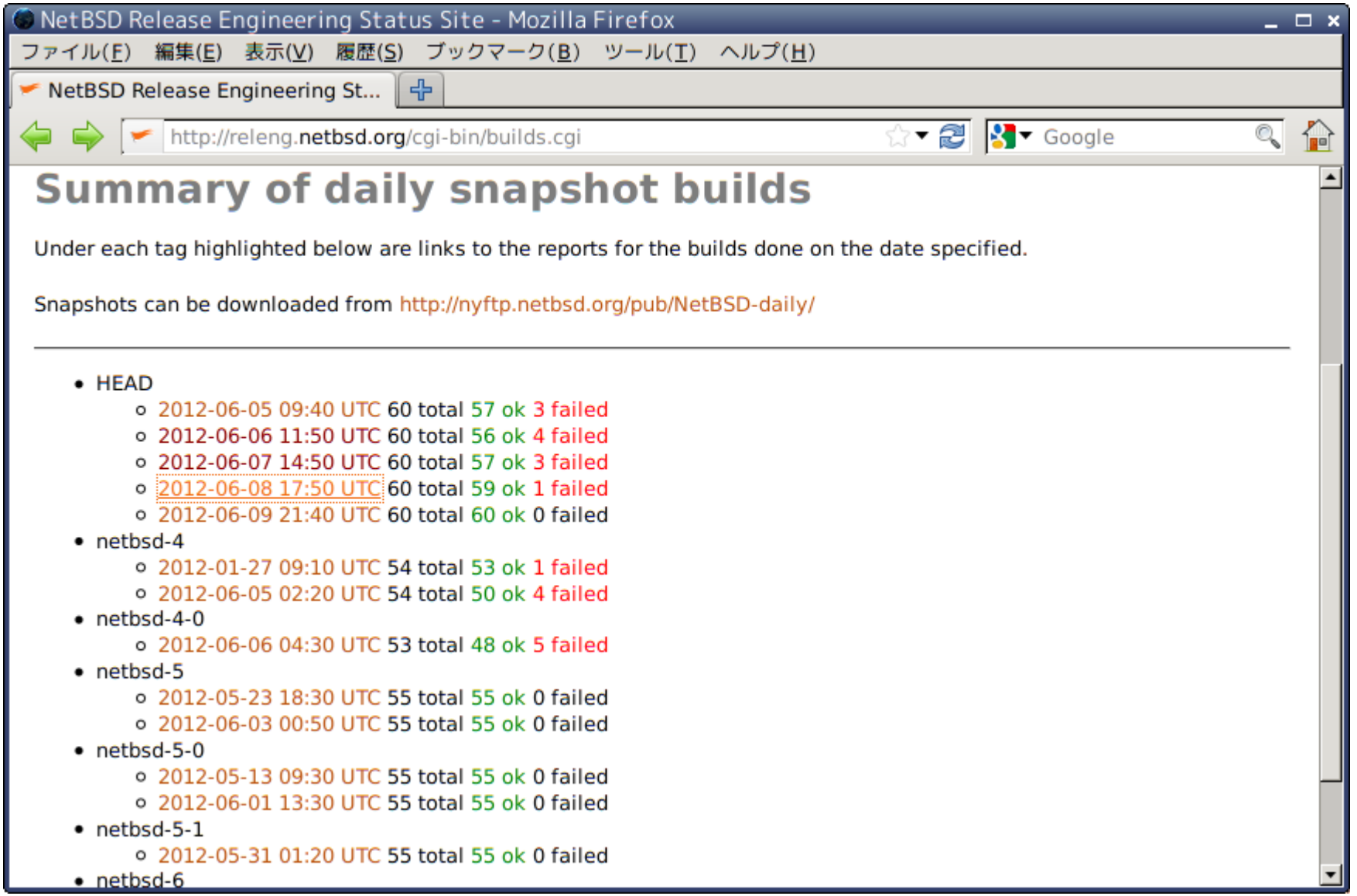
releng.NetBSD.org

http://releng.NetBSD.org/



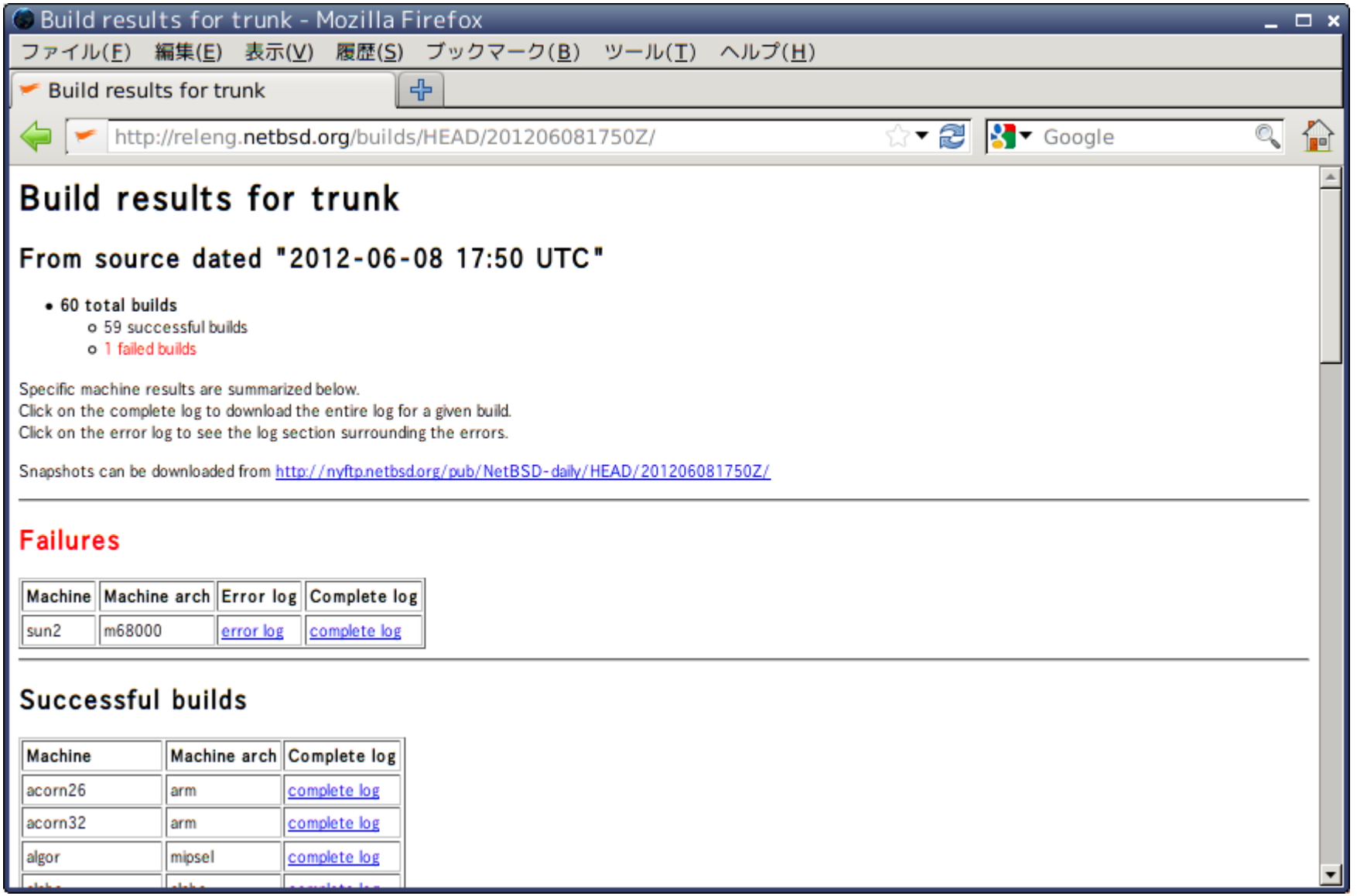
デイリービルド結果サマリー

Summary of daily snapshot builds



ある日のNetBSD -current

59 successful builds, 1 failed builds



さらなる自動化

■ デイリービルドの弊害

・「ビルド至上主義」問題

・とりあえずエラーを消すやっつけ修正する人が発生

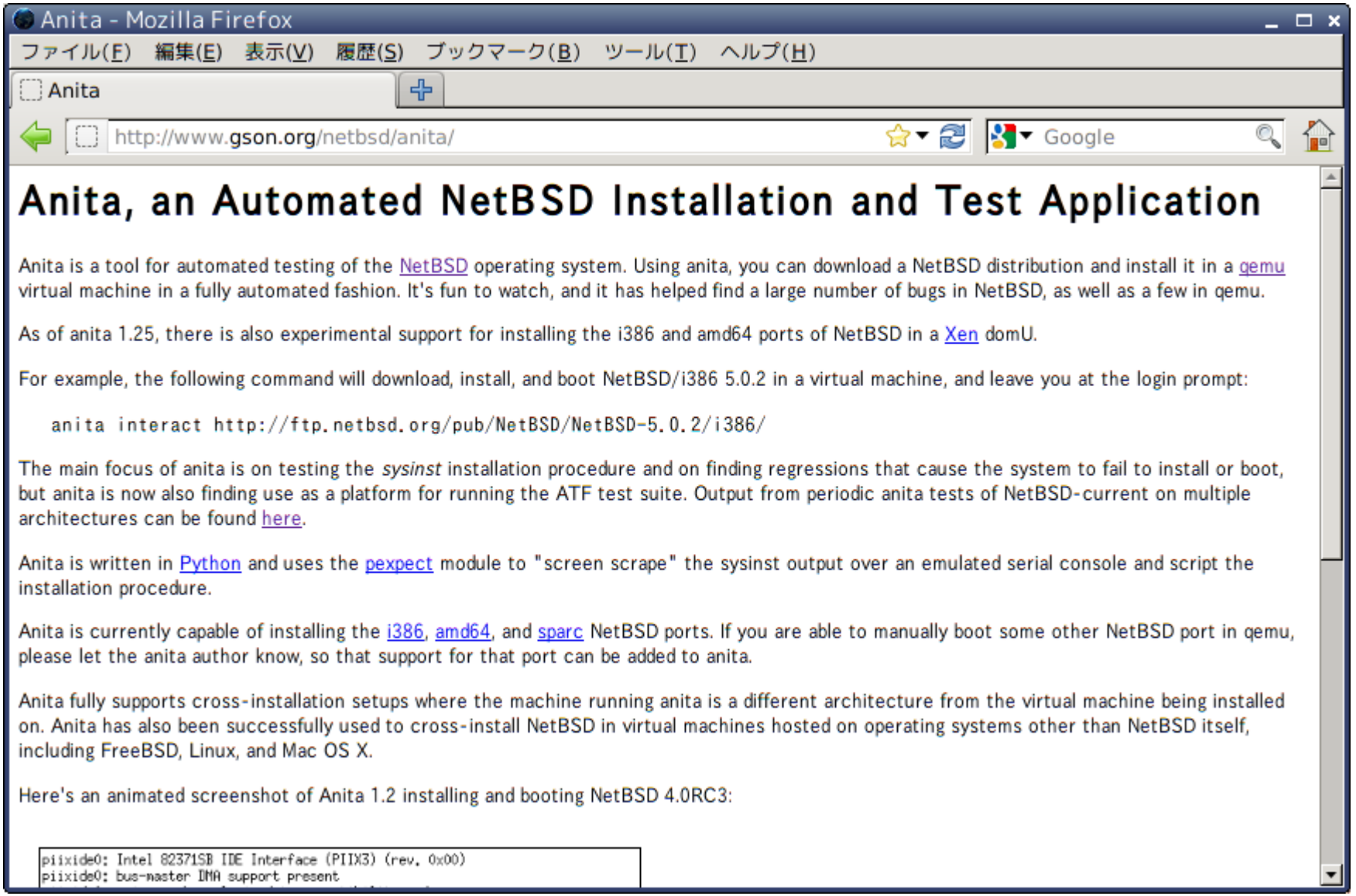
・「ビルドできたけど壊れてて動かない」という問題

⇒その先の動作確認まで自動化する人が発生

・バイナリ作成後、エミュレータと専用ツールを使い
インストール ～ 起動 ～ テスト実行 まで自動化

Anita: 全自動NetBSDテスト

http://www.gson.org/netbsd/anita/



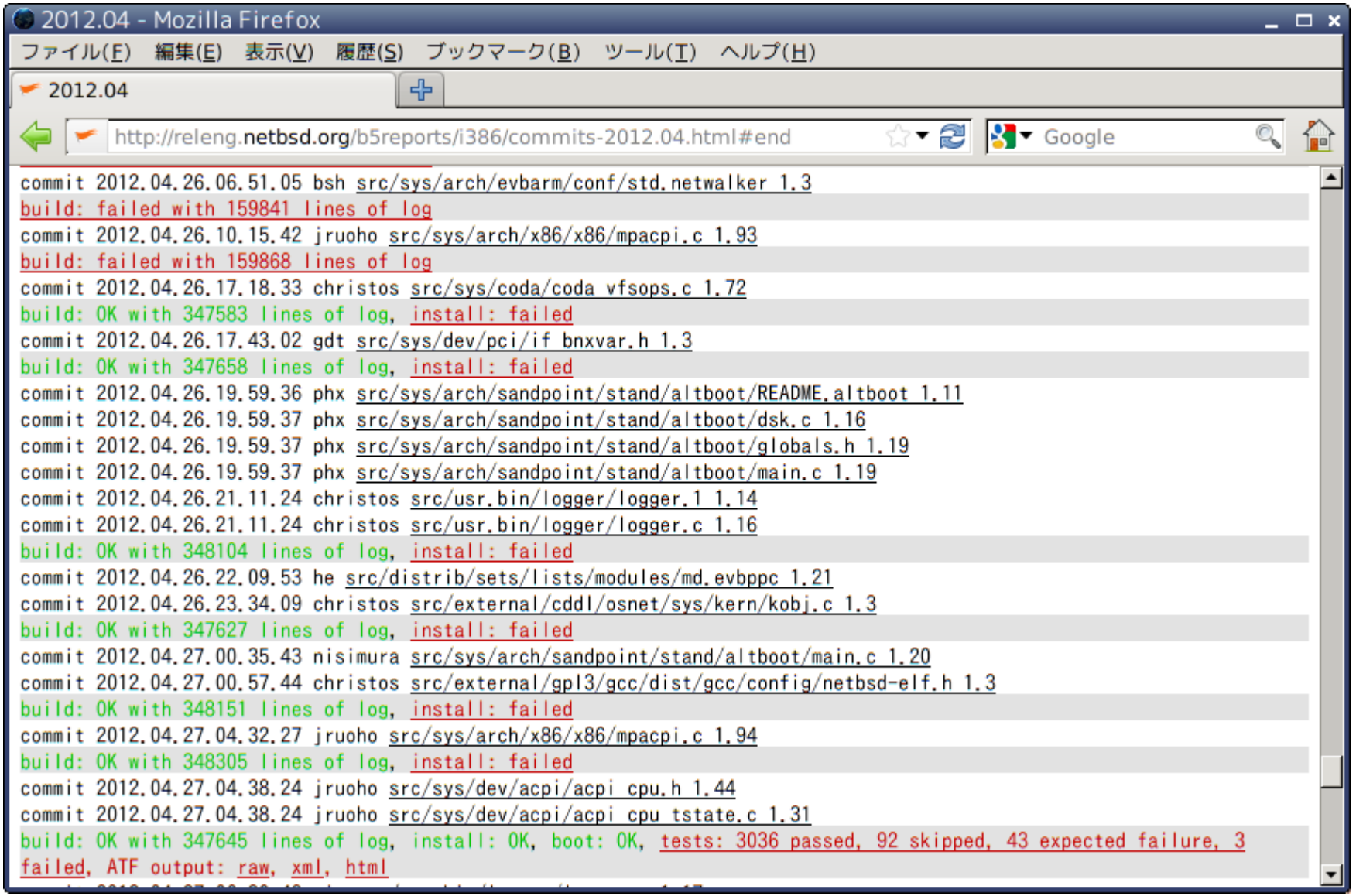
Anita の動作

- エミュレータ（QEMU）を起動
- シリアルコンソール設定でインストールCDを起動
- スクリプトで sysinst を使ってフルインストール
- その後マルチユーザーで起動
- ATF（Automated Testing Framework）を実行
  ※ ATF: NetBSD標準のテスト機構

- 現状 amd64, i386, sparc のみ対応

Anita実行結果サマリー

誰がいつ壊した/直したが一目瞭然という恐怖



クロスビルド用ツールチェイン

■ 肝は src/tools の下

ざっと以下のディレクトリが存在 ……要はホストで実行するもの全部

amiga-elf2bb amiga-txlt asn1_compile autoconf awk binstall binutils cap_mkdb cat cksum compat compile_et config crunchgen ctags ctfconvert ctfmerge db dbsym disklabel fdisk fgen file gcc gdb genassym gencat gettext gmake gmp grep groff headerlist hexdump host-mkdep hp300-mkboot hp700-mkboot ibmnws-ncdcs installboot join lex libctf libdwarf libelf lint lint1 lint2 llvm llvm-clang llvm-clang-tblgen llvm-include llvm-lib llvm-tblgen lorder m4 m68k-elf2aout macppc-fixcoff make makefs makewhatis mandoc mdsetimage menuc mips-elf2ecoff mkcsmapper mkdep mkesdb mkheaderlist.sh mklocale mknod mktemp mkubootimage mpc mpfr msgc mtree nbperf pax paxctl pcc pigz pkg_install powerpc-mkbootimage pwd_mkdb rpcgen sed sgivol slc sparkcrc stat strfile sunlabel texinfo tic tsort uudecode veriexecgen yacc zic

ツールチェイン概要(1)

- コンパイラ関連
  - gcc, binutils
  - llvm, pcc （オプション）

    ⇒コンパイラの性格上、元からクロス対応してるので configure の設定さえすれば構築可能

    - gcc は GNU make 他のツールやライブラリも要求するので、それらもあわせてビルド

ツールチェイン概要(2)

- コンパイラ以外のビルド用ツール
  - config, lint, groff, mandoc, pax 他
    - サードパーティのツールはそのまま configure で
    - NetBSD専用のツールは libnbcompat という互換ライブラリを用意して最小工数で対応
  - awk, grep, sed も

    ⇒OS別の方言にハマるのが面倒くさいのでこれらの一般ツールも作ってしまいます

ツールチェイン概要(3)

■ インストールイメージ作成用ツール

・ブートローダー、パーティション設定
（installboot, disklabel, fdisk など）

・従来は vnd(4) を使いカーネルサービスで書き込み

⇒ツールプログラム上で通常ファイルに対して
イメージ内のデータを操作する機能を追加

・ファイルシステム作成 （makefs）

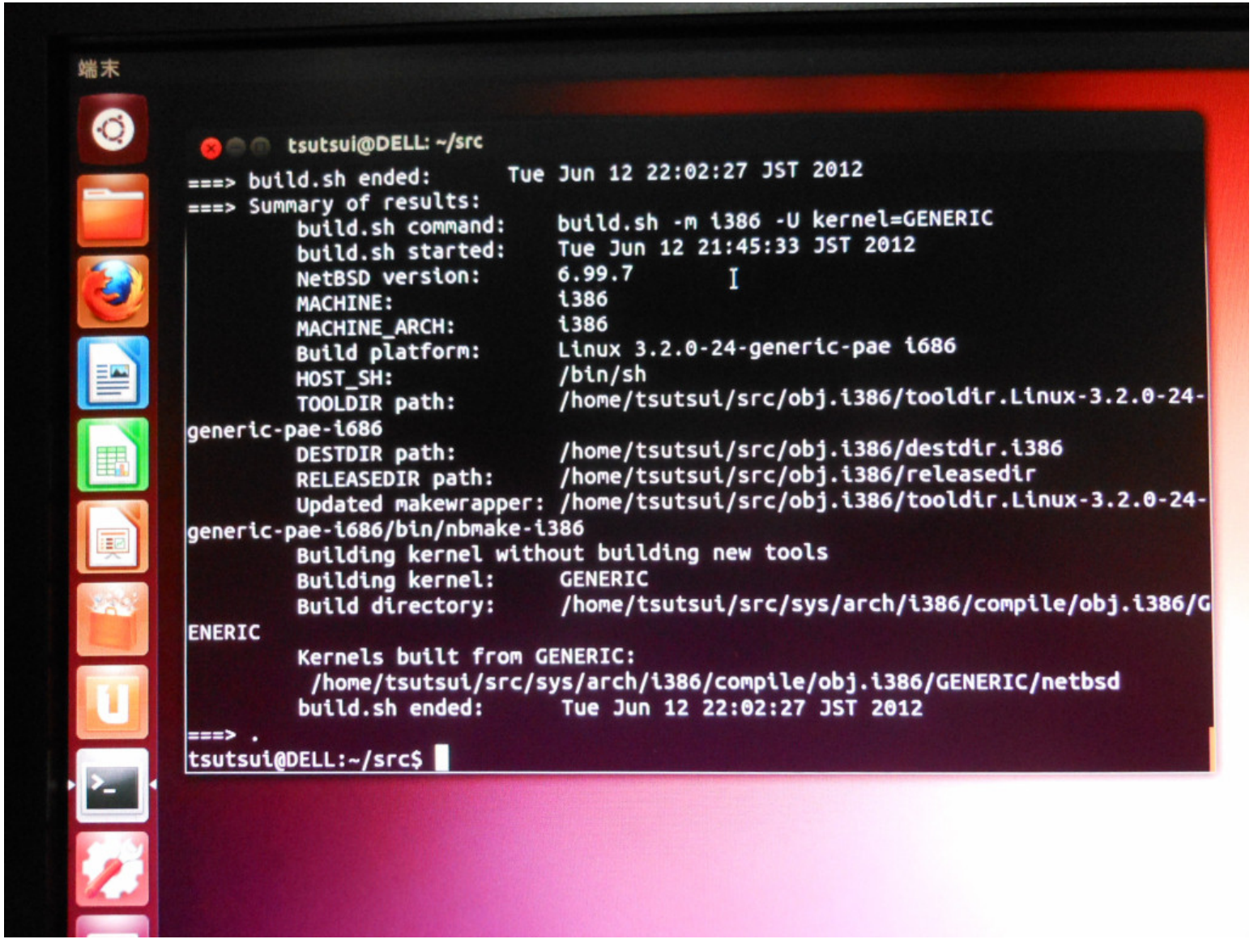
・従来は vnd(4) + newfs(8) + mount(8) で作成

⇒これもユーザーランド上で作成するツールを用意

※ vnd(4): 通常ファイルをディスクデバイスとして扱う擬似デバイスドライバ

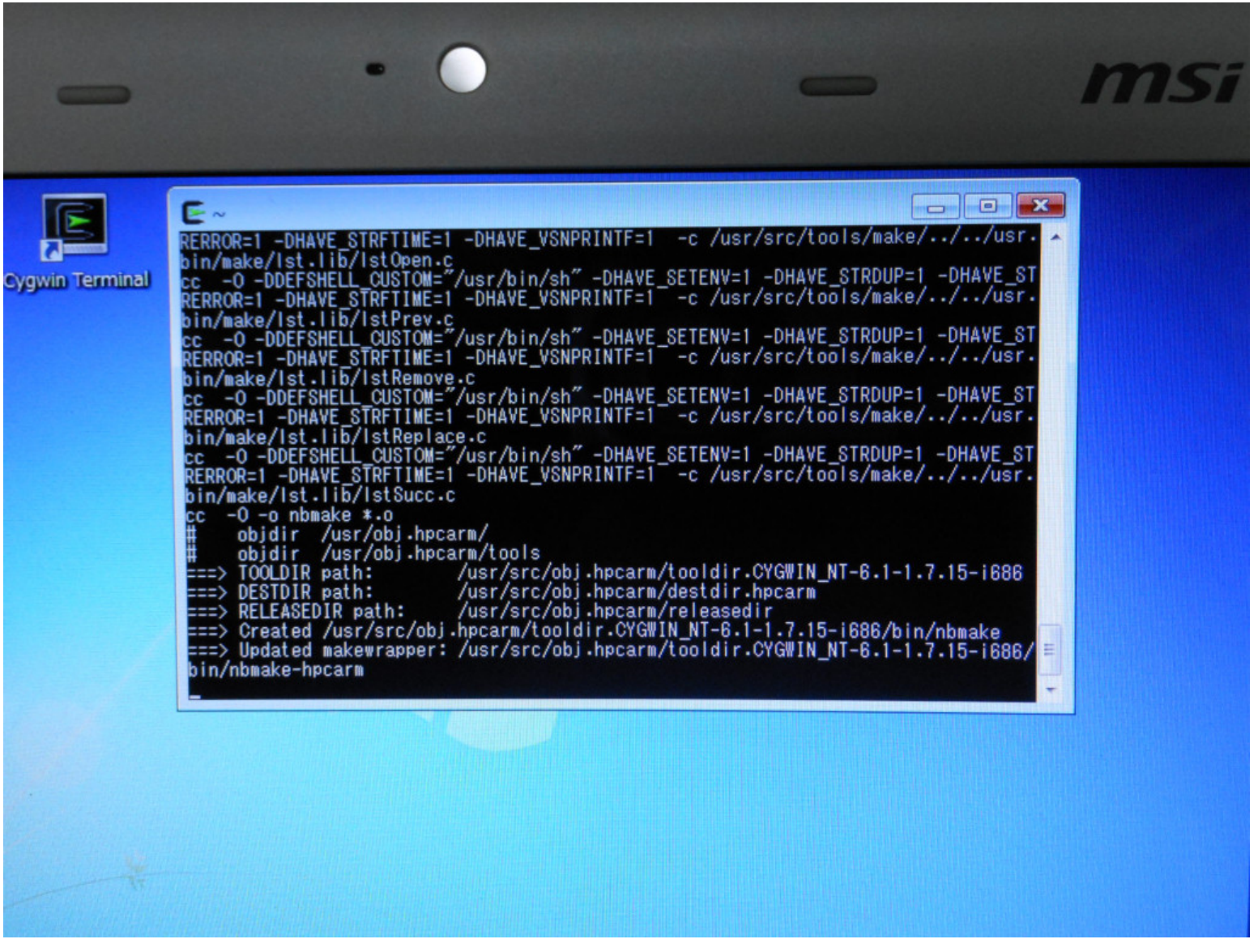
ubuntu上でのクロスビルド

"Of course it builds NetBSD on ubuntu"



Windows上でのクロスビルド

"Of course it builds NetBSD on cygwin"



インストール済みイメージ作成需要

- NetBSDが動くエミュレータの登場が契機
  - TME (The Machine Emulator)
    NetBSD/sun2, sun3, sparc, sparc64
  - GXemul
    NetBSD/pmax, hpcmips, arc, ……

  ⇒エミュレータ上でカーネルのテストする場合、
  インストールが実機以上に遅かったり

インストーラ用イメージを作るツールがあるんだから
インストール済みのイメージも全自動で作れるはず？

実機でのインストール手順

- パーティション設定
- newfs(8) でファイルシステム作成
- バイナリー式を tar で展開
- installboot(8) でブートローダーインストール
- /dev 以下のデバイスノードを作成
- password, timezone その他の各種初期設定

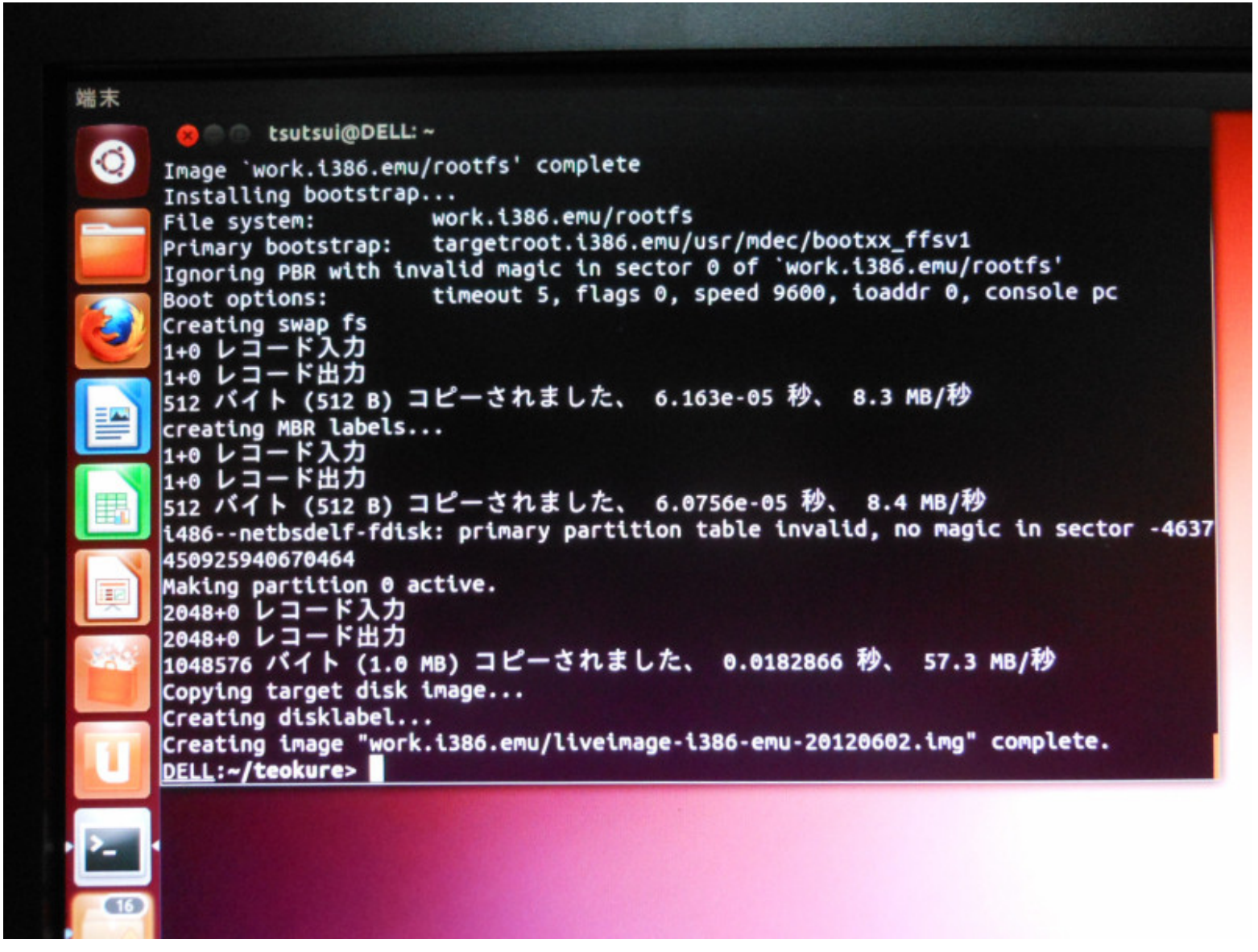
ツールでの起動イメージ作成

- **パーティション設定**
  ⇒ disklabel(8) と fdisk(8) でコマンドライン指定
- **ファイルシステム作成、バイナリ展開、/dev作成**
  ⇒ makefs(8) で作成可能
  …ただしオフセット指定やパーティション分割は無理なので小細工要
- **ブートローダーインストール**
  ⇒ installboot(8) でOK　　…インストールイメージと同じ
- **各種初期設定**
  ⇒ makefs前に sed や cat で設定ファイルに書き込み

……という感じでわりと簡単に作れたので2009年末から個人的に使用

ubuntu上での起動イメージ作成

dd や cat で切り貼りしてるのがバレバレ ^^;



インストールした後どうするか問題

- あまりにも~~不親切~~ストイックなデフォルト
  - **ログインユーザー、ログインシェル**
    sysinst にはログインユーザー作成設定無し
    シェルも /bin/sh, /bin/csh, /bin/ksh だけ
  - **X環境**
    デフォルトはtwmで デスクトップ何それ状態
  - **アプリケーション**
    OfficeどころかWebブラウザすら入ってない
  - **日本語入力環境**
    何をどうすればいいのかすらわからない

ストイック要因

■ 開発者視点に偏りすぎ？

・ **カーネル開発ゴール**
カーネル起動してマルチユーザーまで行けば完了？
そもそもXが動かない（画面が無い）機種もある

・ **技術的興味対象**
「どう実現するか」と「どう設定するか」の違い
……個人的嗜好部分で対立すると不毛という問題

・ **毎年平均年齢が一つずつ**(以下略
大昔から直しながら使っててあるべき設定を知らない
そもそも最新のおすすめアプリを知らなかったり

Live Image 企画

■ ユーザーの選択もいいけど、お試し用も必要でしょう

- Live CD
  - OSのインストール無しに、CD-ROMから起動するとそれなりの環境で起動するCD-ROMイメージ
    - ……NetBSDにも非公式版Live CDが存在
- Live Image とは
  - USBメモリなどに書き込んでそこから起動、またはエミュレータ用ディスクイメージとして指定すればそれなりの環境で起動するディスクイメージ

CDだとRead Onlyで面倒だし、今時はUSBのほうが標準ですよね

Live Image コンセプト(1)

- NetBSDの中の人として
  - とある公式方面からの声
    - Live CDのようにUSBメモリやエミュレータで
      すぐにNetBSDを試せるLive Imageが必要だ
    - Live Image は CD ISO イメージと同様に
      build.sh のしくみの上で作れるようにすべき

えーと、イメージ作成スクリプトなら使ってますけど……

Live Image コンセプト(2)

- NetBSDユーザーとして
  - インストールまでは終わったけど
    その後どうしたらいいのかわかんないよ
  - Firefox は pkgsrc のバイナリがあるっぽいけど
    Flash とか日本語入力とかどう設定すればいいの？

……FlashはLinuxエミュ設定大変だしライセンスも面倒

……日本語入力もいまどき Wnn+kinput2 とかないわー

Live Image コンセプト(3)

- pkgsrcユーザーとして
  - bash, tcsh, emacs, Firefox, OpenOffice は pkgsrc のコンパイル済みバイナリがあるのに、解説記事だとなぜか pkgsrc.tar.gz 取ってきて自分でコンパイルする話ばっかり

……詳しい人はみんな -current を使ってて
バイナリは自分で作る前提になってる？

公式pkgsrcバイナリの存在が不確かという問題もありますが……

Live Image コンセプト(4)

- Twitter/mikutterユーザーとして
  - 定番アプリと言えば Webブラウザだけど、
    最近 pkgsrc に入った Twitter クライアントの
    mikutter もわりといい感じ
  - pkgsrc だと一瞬でインストールできるけど、
    Linux だとどのディストリでも結構大変？

……mikutter人気に乗じて、
mikutterが即使える NetBSD起動イメージ作れば
NetBSDの宣伝になるんじゃね？

Live Image 作成方針

- **設定手順のテンプレ化**
  ⇒ 通常インストールでも使える設定手順
    - 普通の人は toolchain なんて作らないので、
      イメージ作成までとそれ以降の設定は別手順に

- **自動化**
  ⇒ 極力自動で作れるように
    - 手間がかかるとそのうちメンテしなくなる罠
    - 手動インストールだと余分な情報がイメージに残る

インストールアプリ(1)

- すべて pkgsrc バイナリからインストール
  - ログインシェル
    - とりあえず bash と tcsh
      デフォルトは独断で tcsh（変更は可）
  - ウインドウマネージャー
    - 軽さと見た目のバランスで jwm を選定
      ……デスクトップは選択多すぎて検討の余地あり
  - ウェブブラウザ
    - Firefox + 言語パック + gnash
      - Firefox 自体は異論ないと思いますが更新頻度は問題かも
      - 再配布の問題があるので Flash は gnash で対応

インストールアプリ(2)

- 日本語環境
  - IMEは ibus-anthy
    当初は SCIM でしたが有識者の意見や ubuntu などを参考に
  - フォントは VLゴシック と IPAフォント
- その他アプリ
  あとから追加は可能なので、すぐ使えるものを適当に
  - mikutter (ruby, alsa)
  - emacs (anthy-elisp)
  - mlterm, kterm

各種設定(1)

■ わかっていればルーチンワーク?

- **ログインユーザー作成とログイン設定**
  - adduser(8) コマンドでパスワード含め一括作成
    dotファイル内の設定は経験則で独断込み
- **X環境**
  - xorg.conf はナシ (デフォルトの自動判定で xdm起動)
  - 環境変数 XAPPLRESDIR
    kterm や emiclock のような太古のアプリで必要
  - フォントパス: /usr/pkg/lib/X11/fonts/ 関連
  - キーマップ関連: setxkbmap(1) 設定例を記載
  - jwmrc は適当に日本語フォント等の設定を修正

各種設定(2)

- 日本語環境
  - 環境変数 LANG は ja_JP.UTF-8
    - 過去の遺産テキストが無ければあまり困らないでしょう……
    - mlterm では UTF-8 日本語表示にいろいろと設定が必要
  - ibus関連設定
    - 環境変数 XMODIFIERS, GTK_IM_MODULE
    - anthyをデフォルトにするおまじない (thanks to @obacheさん)
- Firefox
  - デフォルトで日本語メニューで起動する設定
    - 環境変数 LANG で UI locale を自動選択する設定
    - システムのアドオンの自動実行を許可する設定

※ 詳細はLive Imageのページを参照
http://www.ceres.dti.ne.jp/tsutsui/netbsd/liveimage/

設定自動化手法

(1)手で打つ内容をシェルスクリプトに

- pkg_add(1) でパッケージインストール
- cat や sed 他で各種設定ファイルを書き込み
- adduser(8) でユーザー作成

(2)必要ファイルを入れたディスクイメージを作成

- 全パッケージバイナリ （依存関係込み）
- 上記設定ファイルと設定スクリプト

  自動化のためディスクイメージ作成スクリプトも作成

あとは イメージをマウント→スクリプト実行 でOK

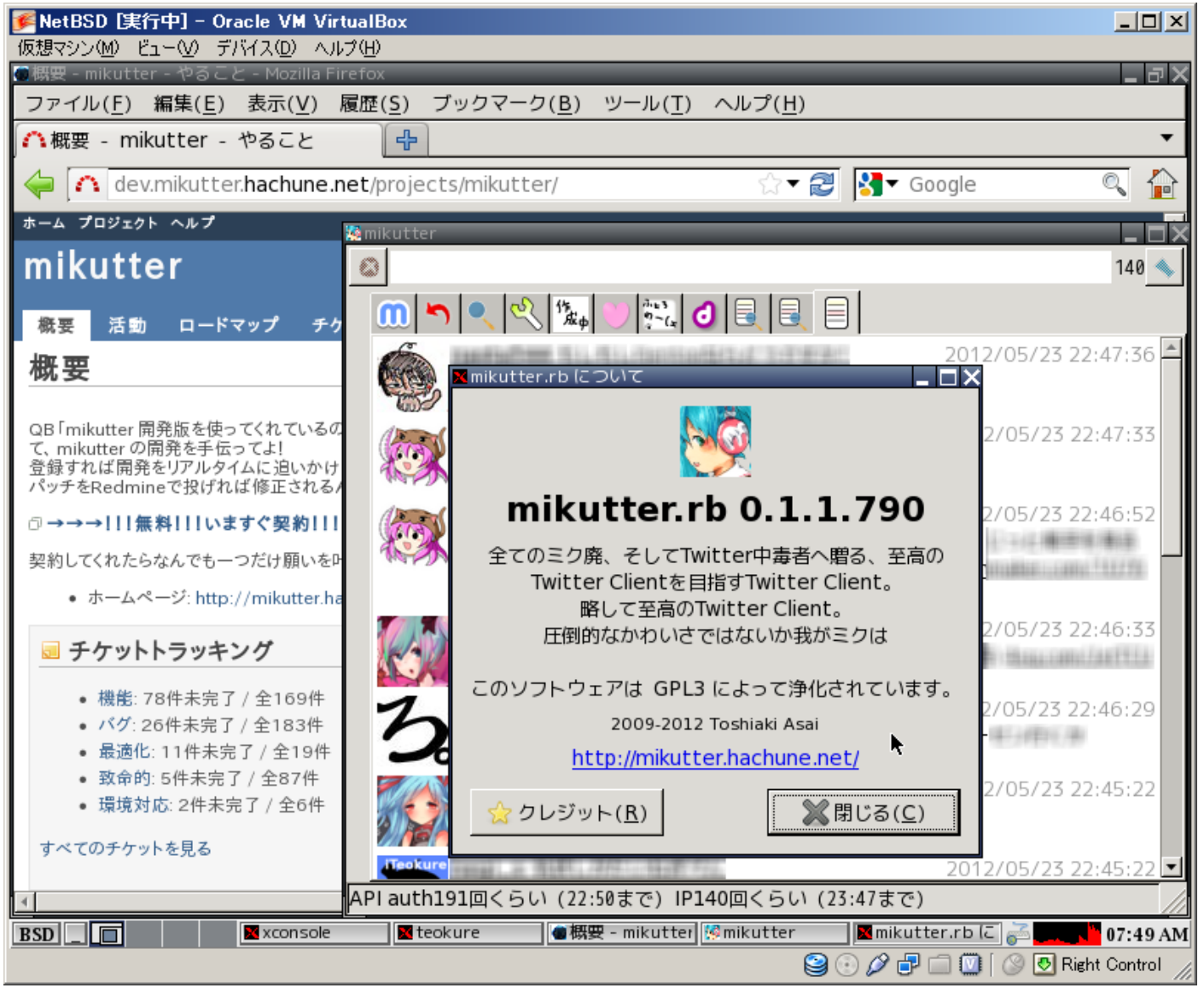
インストール実演デモ

- デモマシン: MSI U135DX
  - 2010年7月発売
  - Atom N455 なネットブック
    デモ用に先週ハードオフで買いました

  - 6.0_BETA2 だと X起動しなくてアセりましたが、
    パッチ当てたら動いたので 6.0 では動くでしょう…
  - 次の世代のN2x00 Atom なネットブックはXは無理かも

Live Image スクリーンショット

Windows上の VirtualBoxで mikutterも使えます



課題

■公式バイナリパッケージ

- 最新版(5.1.2, 6.0_BETA2)の最新pkgsrcバイナリが無い
- 置いてあっても実は中身がおかしかったり
- 自分で必要なパッケージを作ると3時間くらい必要

→思想やマシンリソースの問題もあるのですが、
公式以外も含めバイナリを用意する手段を考え中

■デスクトップ環境選択

- 各人の好みによるので、お勧めがあったら
設定を含めて教えてください

まとめ

■ 昔からNetBSDを使っている人へ

- 大昔の設定に満足せず、イマドキのアプリも重要
- たまにはイチからインストール＋設定を試しましょう

■ NetBSDをこれから使う人へ

- いろいろ不親切なんですが、わかると結構面白いです
- カーネルとかCとかわからなくても、
  ビルド関連は Makefile とシェルスクリプトだけで
  結構遊べます

「楽をするための手段を全力で考えよう」