Reproducible builds

Czyli ostateczne zwycięstwo FOSS.

O Autorze

- Siemanko jestem Alex!
  Studiuje na UJ
  Pracuje w firmie EuroLinux.
- Mail: ab@euro-linux.com
  Prywatny: aleksander.baranowski@yahoo.pl

Czym jest powtarzalne / reprodukowalne budowanie?

- Proces w którym tworzymy weryfikowalny flow, od kodu źródłowego do kodu binarnego.
- Krócej – zgodność bit w bit.

Zalety:

- Użytkownicy mają pewność, że binaria są tym co mieli dostać.
- Developerzy muszą tworzyć oprogramowanie, którego setup i w następstwie kompilacja jest procesem deterministycznym.
- Możliwość wprowadzania zmian na etapie redystrybuowania pakietów (binarek) jest zdecydowanie ograniczona. To jest realne zagrożenie – atak nie na soft jako taki, ale narzędzia budujące go.

Wady:

- Część softu musi zostać załatana, np. testy jednostkowe, które losowo nie przechodzą.
- Co jeśli chcemy celowo mieć losowość w naszym programie?
- Generowanie sekretów, itp. wyklucza się z powtarzalnością. Jak to obejść?

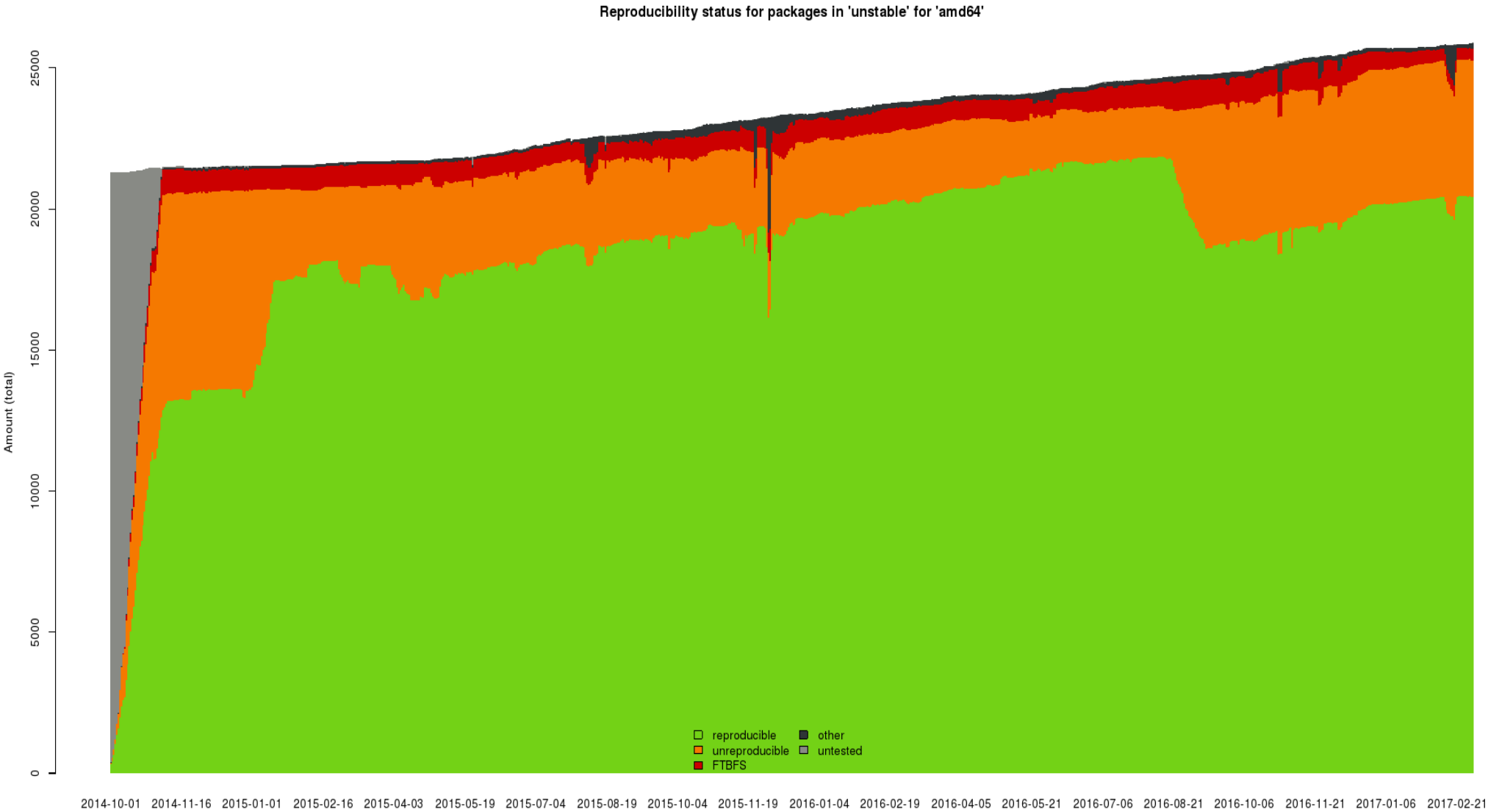
Kilka obserwacji o samym temacie.

- Temat jest dość nowy – a przynajmniej wydaje się być ;).
- Niejednoznaczności już w samej nazwie
  - Nazewnictwo: ReproducibleBuilds, Reproducible Builds, Deterministic Builds, Idempotent Builds.
- Niejednoznaczności w definicji
  - Deterministyczne == Reprodukowalne???
  - Reprodukowalne co do czego? wersji oprogramowania, środowiska?

Komu się już udało?

- Developerzy Bitcoina
- Projekt TOR
- NetBSD 100 %
- FreeBSD > 99%
- Debian > 80%

Debian


Reproducibility status for packages in 'unstable' for 'amd64'
Amount (total)
0
5000
10000
15000
20000
25000
2014-10-01
2014-11-16
2015-01-01
2015-02-16
2015-04-03
2015-05-19
2015-07-04
2015-08-19
2015-10-04
2015-11-19
2016-01-04
2016-02-19
2016-04-05
2016-05-21
2016-07-06
2016-08-21
2016-10-06
2016-11-21
2017-01-06
2017-02-21
reproducible
unreproducible
FTBFS
other
untested

W jaki sposób osiągnąć cel?

- Trzymać metadane odnośnie budownania.
- Gitian - budowanie na maszynie wirtualnej. (TOR i Bitcoin)
- Diffoscope https://diffoscope.org/
- Jenkins
- Skrypty budujące.

Inżynieria PDF czyli przychodzi Alex do przełożonego.

- Tutaj krótka historia czemu nie ma tutaj praktycznej prezentacji.
- Kłopotem jest udowodnienie wartości biznesowej dla przedsiębiorstwa nie budującego wielkich binarek.
- Google oszczędza poprzez wprowadzenie reprodukowalnych buildów i dokompilowywanie tylko zmian – coś jak cacheowanie wyników kompilatora.

Pytanie o Testowanie.

- SHA1sum – bo jest w coreutils, niestety doszło ostatnio do pierwszej kolizji po 9 trylionach generowania przy pomocy specjalnego algorytmu.
- Rozwiązaniem jest uznanie, że sha1sum + md5sum ma część wspólną tak małą, że wystarczająco.
- Pytanie o testowanie dalsze,
  czy funkcje w postaci (int) fun((int) a); da się przetestować gruntownie?

Pytanie o Testowanie cd.

- Czy da się w ogóle mówić o powtarzalnym budowaniu na innych środowiskach? - NIE
- Tutaj krótkie czemu nie.
- Ciekawostka czy Docker zawsze działa tak samo?

A może Harry Potter?

- Wyobraźmy sobie kompilacje warunkowe.
- #ifdef MACRO_VER
- Sialalala
- endif

W jaki sposób próbuje to osiągnąć w EuroLinuxie.

- Kolejka zadań.
- Vagrant + Ansible – Powoływanie maszyn + odpalanie zadań.
- Stress Testing – Różne zmienne środowiskowe, różnice fizyczne między hostami.
- Freeze repozytorium – ma jednak dużą wadę – np. przy przechodzeniu z wersji 6.8 – 6.9 będzie okres gdzie znaczna cześć 6.9 nie będzie się budować aż do następnego zamrożenia.

Istotne zmienne podczas budowania.

- Hostname
- Domainname
- env TZ, LANG, LC_ALL, PATH, USER
- Uid, guid, umask
- Kernel ver
- CPU Type/Model
- DATE!
- Filesystem

Dlaczego jest to ostateczne zwycięstwo FOSS?

- Wolność do uruchamiania programu jak chcecie, w dowolnym celu (wolność 0).
- Wolność do analizowania, jak działa program, i zmieniania go aby robił co i jak potrzebujecie (wolność 1). Warunkiem koniecznym jest dostęp do kodu źródłowego.
- Wolność do rozpowszechniania kopii, byście mogli pomóc innym ludziom (wolność 2).
- Wolność do udoskonalania programu i publicznego rozpowszechniania własnych ulepszeń, dzięki czemu może z nich skorzystać cała społeczność (wolność 3). Warunkiem koniecznym jest tu dostęp do kodu źródłowego.
- **Wolność do uzyskania identycznych wyników kompilacji?**

Web of Trust dla softu.

- W chwili obecnej pobierając soft z wiodących dystrybucji ufamy, iż podpisanie kluczem gpg wystarcza do instalacji paczki w systemie.
- Podobnie jak z importowaniem kluczy zaufanych w gpg możemy w przyszłości stworzyć Web of Trust dla softu.

Koniec.

- Już, wiem z czego będę pisał magisterkę. Potwornie ciekawy temat, w który można wpleść następujące zagadnienia:
  Teorie kompilatorów, Testowanie Oprogramowania, Inżynierie Oprogramowania, Teorie Języków i Automatów, Matematykę Dyskretną, Statystykę.

Pytania?

- Pytania?
  - Pytania?
    - Pytania?
      - Pytania?
        - Pytania?
          - Pytania?
            - Pytania?
              - Pytania?
                - Pytania?
                  - Pytania?