

Da hobbisti a mainstream

Challenges e opportunità del FOSS nel prossimo decennio(?)

Francesco P. Lovergine
<frankie@fsfe.org>
@gisgeek@floss.social

Free Software Foundation Europe
Fellowship

http://creativecommons.org/licenses/by-nc-nd/4.0/legalcode



Foundation



http://www.fsfe.org/

https://lists.fsfe.org/mailman/listinfo/bari

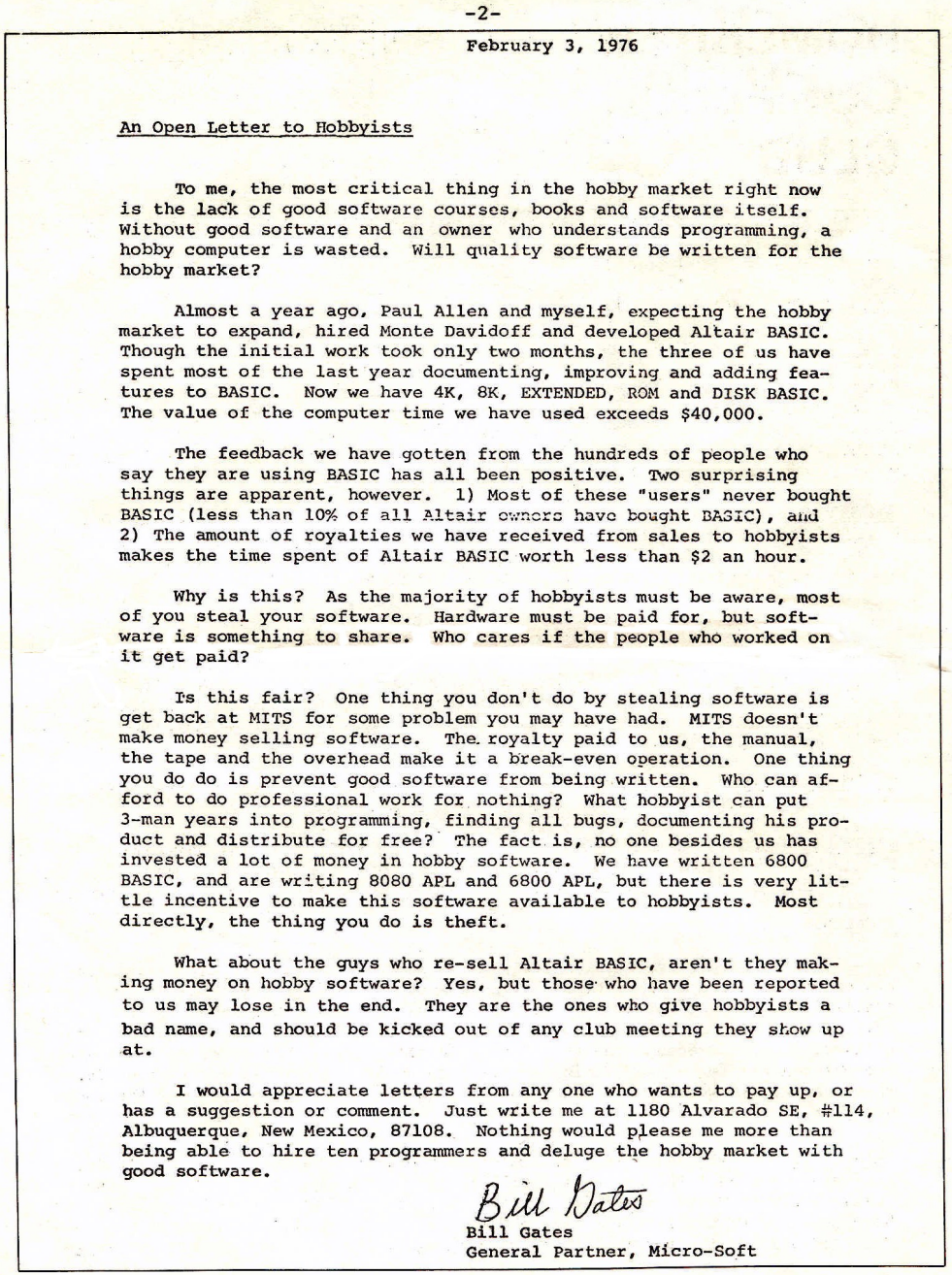


Who am I ?

- Classe 1967
- Laurea 1991
- OSMapper dal 2008
- Ricercatore CNR dal 1997
- Debian Developer dal 2000 →
- FOSS activist da sempre
- Presidente LugBARI 2002–2009
- Geek per nascita :-)

Outline

- Da dove siamo partiti
- Dove siamo arrivati
- Dove andremo (probabilmente) a finire
- *… e forse qualche deep thought in corso d’opera*



-2-

February 3, 1976

An Open Letter to Hobbyists

To me, the most critical thing in the hobby market right now is the lack of good software courses, books and software itself. Without good software and an owner who understands programming, a hobby computer is wasted. Will quality software be written for the hobby market?

Almost a year ago, Paul Allen and myself, expecting the hobby market to expand, hired Monte Davidoff and developed Altair BASIC. Though the initial work took only two months, the three of us have spent most of the last year documenting, improving and adding features to BASIC. Now we have 4K, 8K, EXTENDED, ROM and DISK BASIC. The value of the computer time we have used exceeds $40,000.

The feedback we have gotten from the hundreds of people who say they are using BASIC has all been positive. Two surprising things are apparent, however. 1) Most of these "users" never bought BASIC (less than 10% of all Altair owners have bought BASIC), and 2) The amount of royalties we have received from sales to hobbyists makes the time spent of Altair BASIC worth less than $2 an hour.

Why is this? As the majority of hobbyists must be aware, most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid?

Is this fair? One thing you don't do by stealing software is get back at MITS for some problem you may have had. MITS doesn't make money selling software. The royalty paid to us, the manual, the tape and the overhead make it a break-even operation. One thing you do do is prevent good software from being written. Who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free? The fact is, no one besides us has invested a lot of money in hobby software. We have written 6800 BASIC, and are writing 8080 APL and 6800 APL, but there is very little incentive to make this software available to hobbyists. Most directly, the thing you do is theft.

What about the guys who re-sell Altair BASIC, aren't they making money on hobby software? Yes, but those who have been reported to us may lose in the end. They are the ones who give hobbyists a bad name, and should be kicked out of any club meeting they show up at.

I would appreciate letters from any one who wants to pay up, or has a suggestion or comment. Just write me at 1180 Alvarado SE, #114, Albuquerque, New Mexico, 87108. Nothing would please me more than being able to hire ten programmers and deluge the hobby market with good software.

Bill Gates

Bill Gates
General Partner, Micro-Soft

- Gli hobbisti e la *condivisione* del software …
- La famosa ‘lettera di Zio Bill’ del ‘76 è in parte condivisibile, anche se dare del ladro al lettore non è una grande idea…
- Fino agli anni ‘70 chi c’era (pochi 😊) ricorda che il software era considerato una commodity da fornire insieme all’hardware per usare i prodotti, **tipicamente includendo i sorgenti**.
- Il software come ‘prodotto da vendere’ nasce con i microcomputer e da lì cominciano le licenze d’uso.
- Una parte degli utenti-sviluppatori (tipicamente high-end) va per la propria strada e non ci sta… → Richard Martin Stallman
- Un’altra parte degli utenti (tipicamente low-end) va anche per la propria strada e non ci sta lo stesso → pirateria
- Due facce di una stessa medaglia: la percezione del valore

Progetto GNU @ 1984 & Linux kernel @ 1991





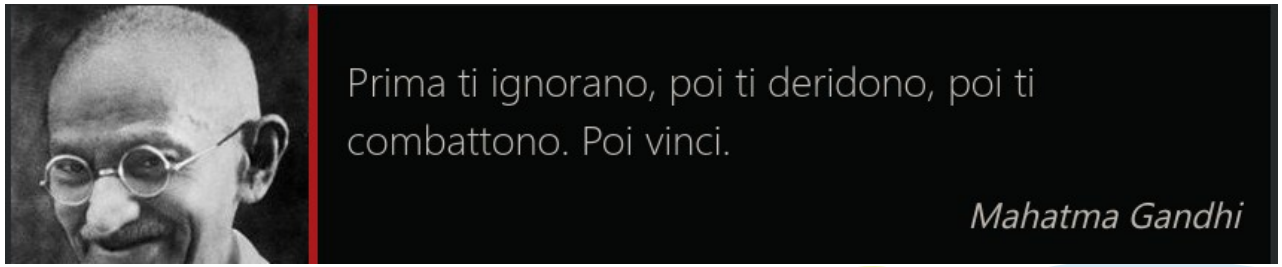
- Copyright e licenze d'uso permissive e virali (GPL)
- Uso – Modifica – Condivisione – Distribuzione liberi
- Ecosistemi interoperanti
- Sviluppo collaborativo e alla luce del sole (open repositories, issue tracker, forum di sviluppo, ...)
- Utenti e sviluppatori sullo stesso piano
- Salire sulle spalle dei giganti → on boarding

Le 4 libertà del free software

Libertà 0: Libertà di eseguire il programma per qualsiasi scopo. La libertà di usare un programma significa libertà per qualsiasi tipo di persona od organizzazione di utilizzarlo su qualsiasi tipo di sistema informatico, per qualsiasi tipo di attività e senza dover successivamente comunicare con lo sviluppatore o con qualche altra entità specifica. Quello che conta per questa libertà è lo scopo dell'utente, non dello sviluppatore; come utenti potete eseguire il programma per i vostri scopi; se lo ridistribuite a qualcun altro, egli è libero di eseguirlo per i propri scopi, ma non potete imporgli i vostri scopi.
Libertà 1: Libertà di studiare il programma e modificarlo. L'accessibilità al codice sorgente è una condizione necessaria per il software libero, altrimenti non avrebbero senso neanche la libertà 0 e la 2.
Libertà 2: Libertà di ridistribuire copie del programma in modo da aiutare il prossimo.
Libertà 3: Libertà di migliorare il programma e di distribuirne pubblicamente i miglioramenti, in modo tale che tutta la comunità ne tragga beneficio. Questa libertà comprende quella di usare e rilasciare le versioni modificate come software libero. Una licenza libera può anche permettere altri modi di distribuzione; insomma, non c'è l'obbligo che si tratti di una licenza con copyleft. Tuttavia, una licenza che imponesse che le versioni modificate non siano libere non si può categorizzare come licenza libera.

Fast forward … 2023

- Linux è il più diffuso s.o. su server
- Linux (o quasi 😕) è il più diffuso s.o. su mobile e *dispositivi-che-non-sono-computer* → IoT
- Il web è il più usato paradigma di sviluppo software (cit. Electron) e di digital enhancement
- Cloud pervasivo
- Convergenza mobile web
- Software sempre più complessi rendono impossibile l'esistenza di una software company che possa sviluppare prodotti **completamente in house** in modo sostenibile

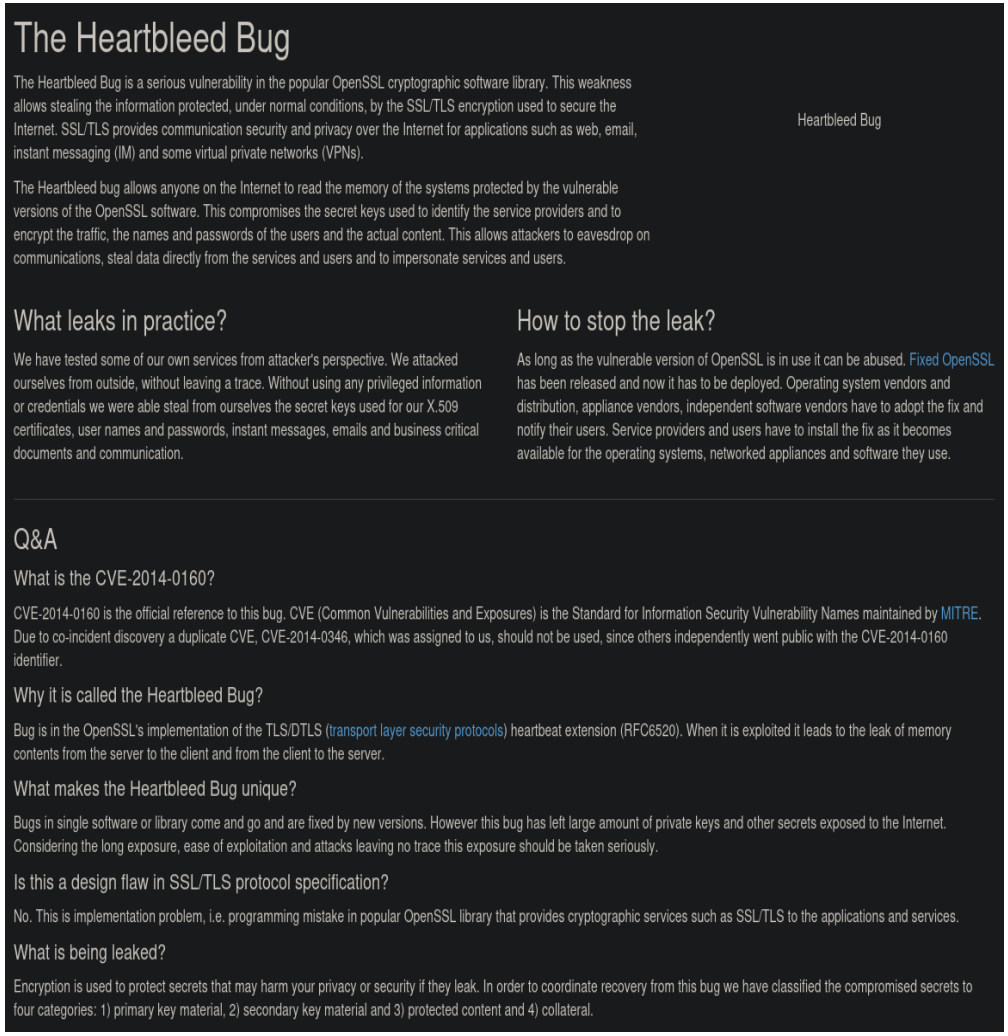




“Potente tu sei diventato…

Il Lato Oscuro percepisco in te.”

S come Sostenibilità



The Heartbleed Bug

The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. This weakness allows stealing the information protected, under normal conditions, by the SSL/TLS encryption used to secure the Internet. SSL/TLS provides communication security and privacy over the Internet for applications such as web, email, instant messaging (IM) and some virtual private networks (VPNs).

Heartbleed Bug

The Heartbleed bug allows anyone on the Internet to read the memory of the systems protected by the vulnerable versions of the OpenSSL software. This compromises the secret keys used to identify the service providers and to encrypt the traffic, the names and passwords of the users and the actual content. This allows attackers to eavesdrop on communications, steal data directly from the services and users and to impersonate services and users.

What leaks in practice?

We have tested some of our own services from attacker's perspective. We attacked ourselves from outside, without leaving a trace. Without using any privileged information or credentials we were able steal from ourselves the secret keys used for our X.509 certificates, user names and passwords, instant messages, emails and business critical documents and communication.

How to stop the leak?

As long as the vulnerable version of OpenSSL is in use it can be abused. Fixed OpenSSL has been released and now it has to be deployed. Operating system vendors and distribution, appliance vendors, independent software vendors have to adopt the fix and notify their users. Service providers and users have to install the fix as it becomes available for the operating systems, networked appliances and software they use.

Q&A

What is the CVE-2014-0160?

CVE-2014-0160 is the official reference to this bug. CVE (Common Vulnerabilities and Exposures) is the Standard for Information Security Vulnerability Names maintained by MITRE. Due to co-incident discovery a duplicate CVE, CVE-2014-0346, which was assigned to us, should not be used, since others independently went public with the CVE-2014-0160 identifier.

Why it is called the Heartbleed Bug?

Bug is in the OpenSSL's implementation of the TLS/DTLS (transport layer security protocols) heartbeat extension (RFC6520). When it is exploited it leads to the leak of memory contents from the server to the client and from the client to the server.

What makes the Heartbleed Bug unique?

Bugs in single software or library come and go and are fixed by new versions. However this bug has left large amount of private keys and other secrets exposed to the Internet. Considering the long exposure, ease of exploitation and attacks leaving no trace this exposure should be taken seriously.

Is this a design flaw in SSL/TLS protocol specification?

No. This is implementation problem, i.e. programming mistake in popular OpenSSL library that provides cryptographic services such as SSL/TLS to the applications and services.

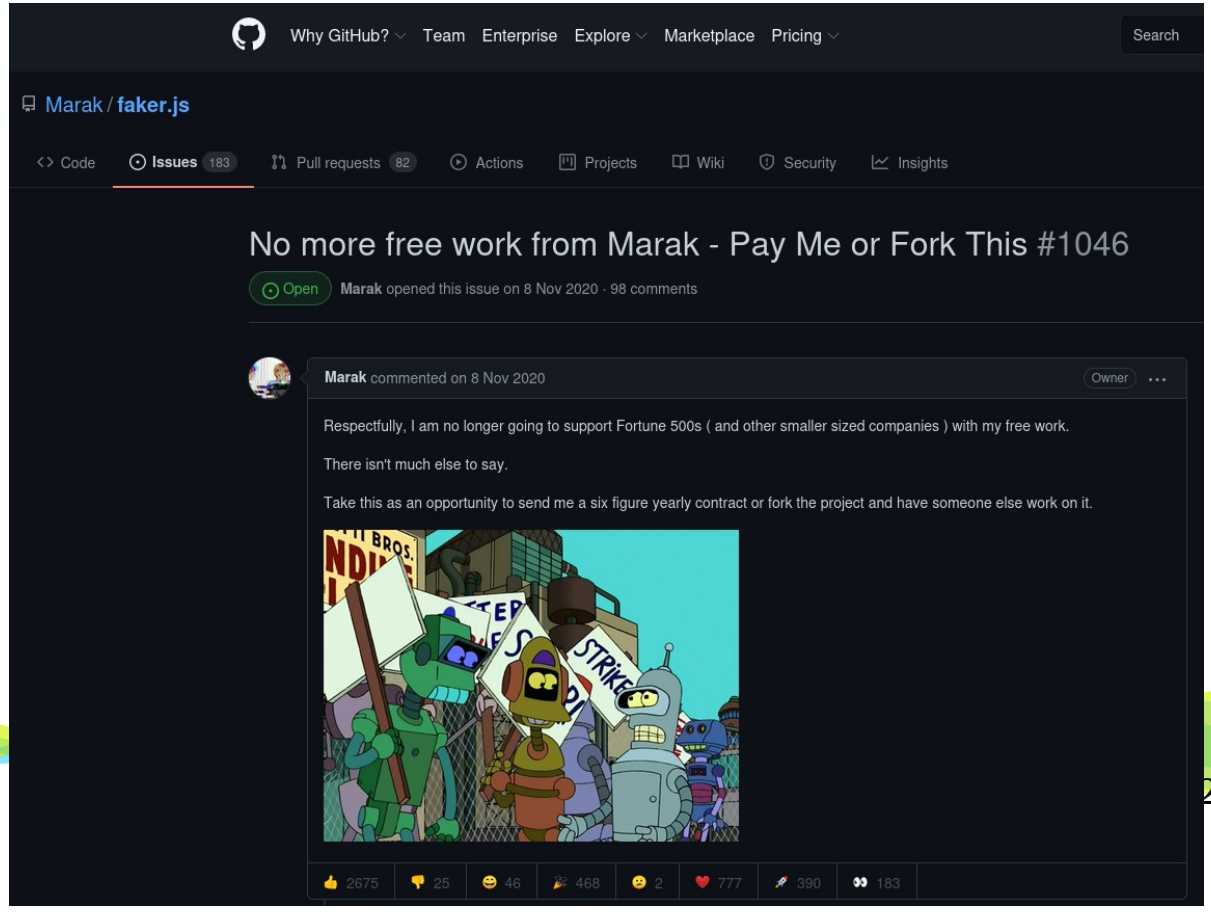
What is being leaked?

Encryption is used to protect secrets that may harm your privacy or security if they leak. In order to coordinate recovery from this bug we have classified the compromised secrets to four categories: 1) primary key material, 2) secondary key material and 3) protected content and 4) collateral.

- Uno di innumerevoli esempi nel tempo: **OpenSSL**
- Insospettabili giganti (per esempio l'intera infrastruttura di Internet) si poggiano sulle spalle di progetti formica, spesso inconsapevolmente o per sottovalutazione.
- Nel 2014 si corse ai ripari di urgenza, ma…
- Quanti progetti sono ancora nelle medesime "condizioni" ?
- TROPPI
- Nel 2023 molti linguaggi di sviluppo si basano su package hub con lughe catene di dipendenze → Python, Nodejs, Rust, Go, …

Cosa fa la società $X per sostenere il progetto $Y ?

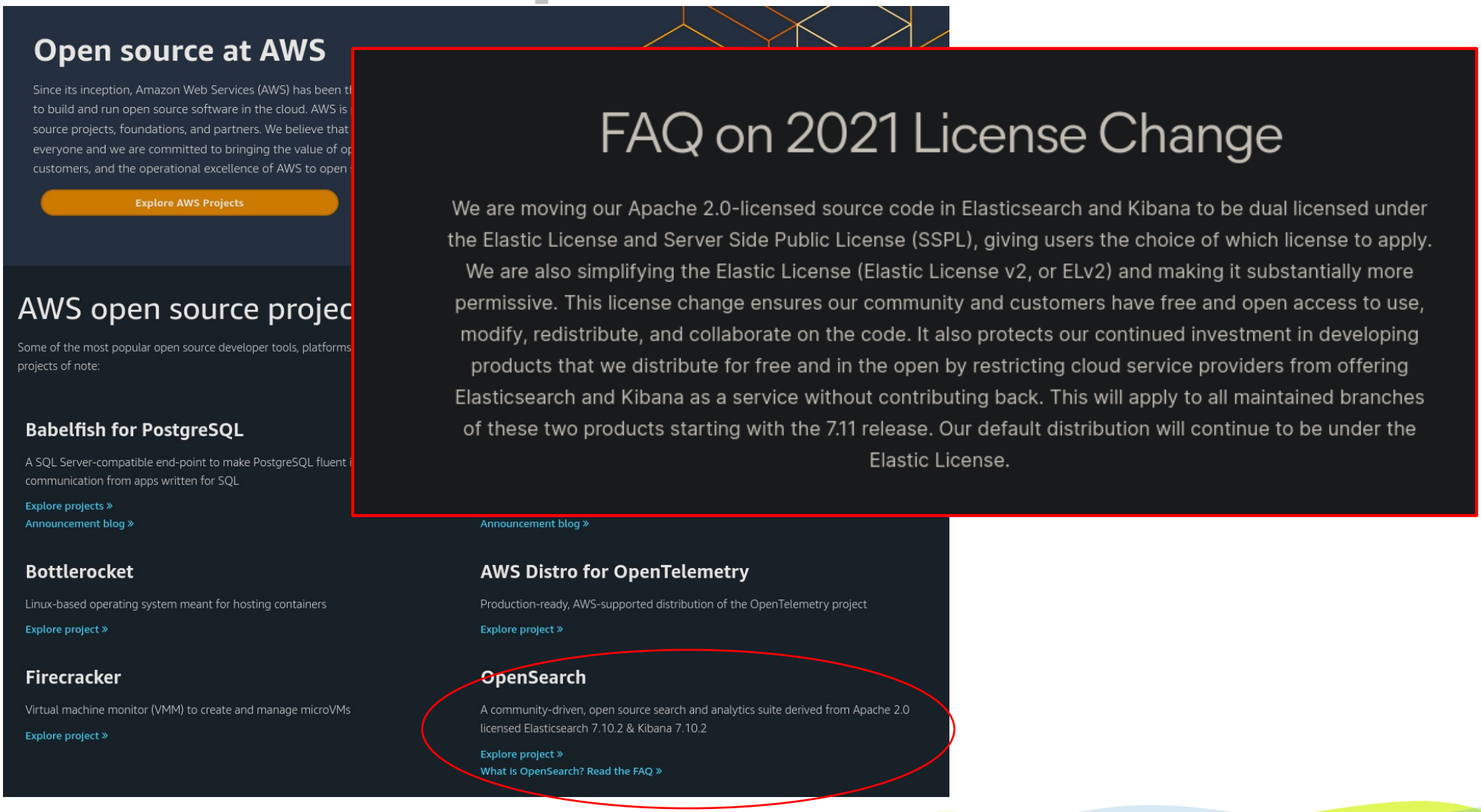
- Moltissime software (e non) company hanno approcci puramente predatori nei riguardi del FOSS.
- I contributi possibili sono innumerevoli: di codice e patch, economici, di risorse umane, di tempo di sviluppo, di risorse di calcolo, di banda e storage, sponsorship di eventi, di sviluppatori.
- Anche insospettabili big companies che non hanno certamente problemi di badget...



Il famosissimo caso delle librerie color e faker di Nodejs nel 2020.



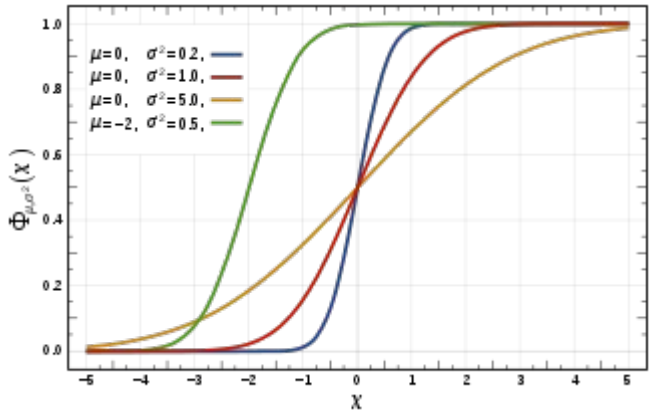
Gli insospettabili … diciamo



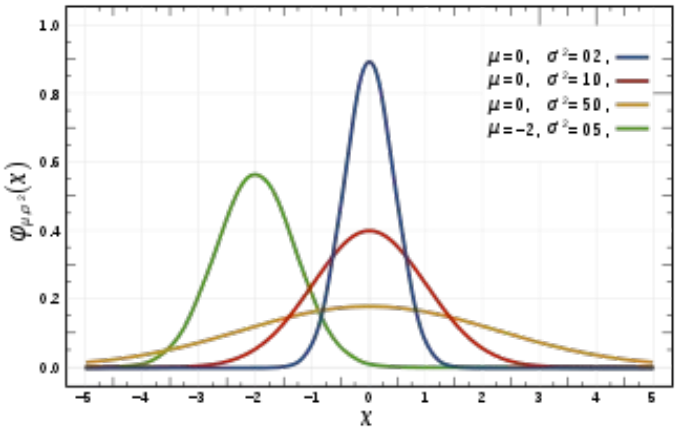
G come Governance

- Progetti monocratici e mono-aziendali → rischio di controparte
- Trasferimenti di copyright → rischio di controparte (cit. MySQL)
- Regole di contribuzione/partecipazione opache o non ben definite
- Rapporti tra maintainer → rischio di relazione
- Insufficienti fonti di entrate per attività con scarsa *attrattività* → rischio di degrado qualità

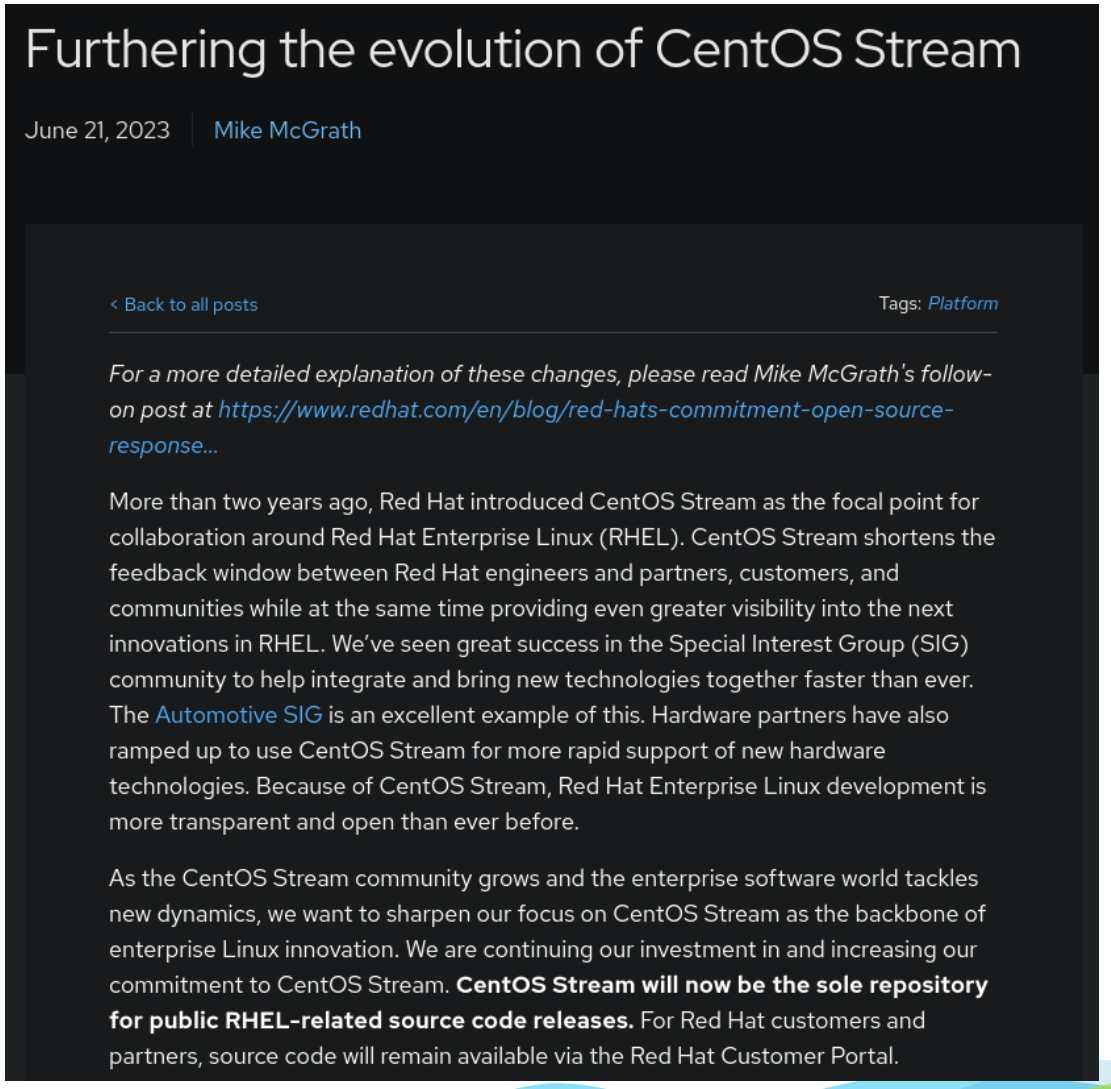
Progetti a sigmoide



Progetti a gaussiana



Un paio di esempi



Furthering the evolution of CentOS Stream

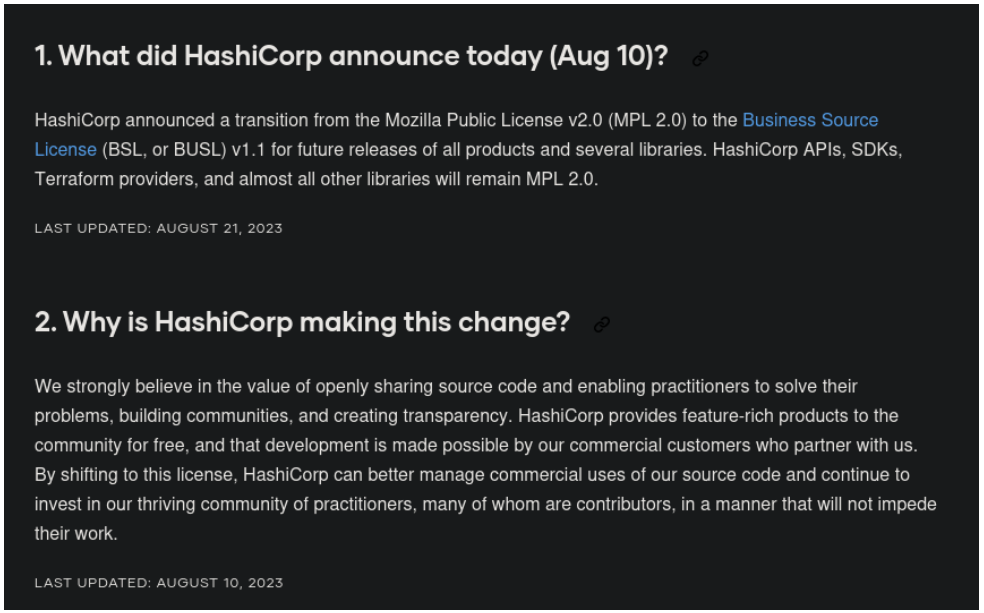
June 21, 2023 | Mike McGrath

< Back to all posts Tags: *Platform*

For a more detailed explanation of these changes, please read Mike McGrath's follow-on post at https://www.redhat.com/en/blog/red-hats-commitment-open-source-response...

More than two years ago, Red Hat introduced CentOS Stream as the focal point for collaboration around Red Hat Enterprise Linux (RHEL). CentOS Stream shortens the feedback window between Red Hat engineers and partners, customers, and communities while at the same time providing even greater visibility into the next innovations in RHEL. We've seen great success in the Special Interest Group (SIG) community to help integrate and bring new technologies together faster than ever. The Automotive SIG is an excellent example of this. Hardware partners have also ramped up to use CentOS Stream for more rapid support of new hardware technologies. Because of CentOS Stream, Red Hat Enterprise Linux development is more transparent and open than ever before.

As the CentOS Stream community grows and the enterprise software world tackles new dynamics, we want to sharpen our focus on CentOS Stream as the backbone of enterprise Linux innovation. We are continuing our investment in and increasing our commitment to CentOS Stream. **CentOS Stream will now be the sole repository for public RHEL-related source code releases.** For Red Hat customers and partners, source code will remain available via the Red Hat Customer Portal.



1. What did HashiCorp announce today (Aug 10)?

HashiCorp announced a transition from the Mozilla Public License v2.0 (MPL 2.0) to the Business Source License (BSL, or BUSL) v1.1 for future releases of all products and several libraries. HashiCorp APIs, SDKs, Terraform providers, and almost all other libraries will remain MPL 2.0.

LAST UPDATED: AUGUST 21, 2023

2. Why is HashiCorp making this change?

We strongly believe in the value of openly sharing source code and enabling practitioners to solve their problems, building communities, and creating transparency. HashiCorp provides feature-rich products to the community for free, and that development is made possible by our commercial customers who partner with us. By shifting to this license, HashiCorp can better manage commercial uses of our source code and continue to invest in our thriving community of practitioners, many of whom are contributors, in a manner that will not impede their work.

LAST UPDATED: AUGUST 10, 2023

- Due diverse reazioni che mostrano il '*malessere*' rispetto alle *predator* companies …
- Le licenze non sono scolpite nella pietra. Si cambiano e non sempre in modo 'amichevole'.

A come Abandonware

- Github e gli altri hub contengono moltissimi progetti che sono solo proof-of-concept, progetti abortiti, progetti che non hanno più maintainance, codice scritto male e manutenuto peggio, con troppo pochi dev → **puntare sul cavallo bolso** (da utenti o da sviluppatori)
- Peggio, ci sono anche progetti che lo diventeranno, magari fra 1-2 anni, e ve ne accorgerete dopo …
- Il maintainer responsabile, quando perde interesse, cerca altri contributori cui passare il testimone, ma non viviamo in un mondo perfetto. E non sempre si trova la soluzione.
- Ma non è che nel mondo proprietario si viva meglio 😆 (cit. Actionscript) …

M come Monopoli

- Github è un monopolio (→ Codeberg, Gitlab, ...)
- Stackoverflow è un monopolio
- Il cloud è un oligopolio
- Alcuni prodotti con una governance *problematica* sono pericolosamente vicini allo stato di monopolio di fatto (es. Chrome, VSCode, ...)
- Nel **2017** ne parlammo già a lungo (cfr. *Privacy e sicurezza ai tempi del cloud*). La situazione non è migliore, anzi...
- *Tempi maturi per diversificare e abbracciare il fediverse*

Soluzioni?

- Le soluzioni sono individuali, come utenti semplici o come developer. Non si può salvare il mondo tutti i giorni ☺
- Adottare prodotti, tool, linguaggi e framework FOSS è possibile e conveniente, ma sempre in modo *fair* e *cum grano salis*. Un occhio alla governance e alla sostenibilità è oggi fondamentale molto più di ieri.
- Per i geek: lasciate perdere gli *hype*, siate razionali e non fan boy.
- Se è gratis, il costo semplicemente/probabilmente non si vede. Come e chi lo paga?
- Contribuire è alla portata di tanti, a vari livelli (documentazione, speech, test, bug report, ecc.). E permette di migliorare globalmente a lungo termine la qualità.

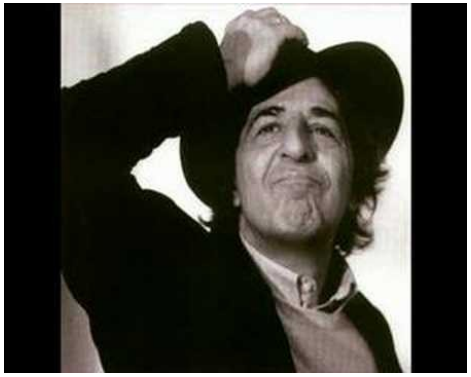
La libertà

“La libertà non è star sopra un albero

Non è neanche avere un'opinione gratis

La libertà non è uno spazio ~~libero~~

Libertà è ***partecipazione***”



Giorgio Gaber

Discussione

