The Monthly GNU Column

Brave GNU World

This column looks into projects and current affairs in the world of free software from the perspective of the GNU Project and the FSF. In this issue, we will be focusing on the Balsa email program, Spam: kill or cure, safe chatting with Silky, Agnula and the Agnula Trademark License. BY GEORG C.F. GREVE

Welcome to another issue of the Brave GNU World. This month's central topic is communication, and the successful completion of the European Commission's first free software project.

Email

Email has been a standard requirement for most computer users for many years now. Linux has a variety of email programs. Many of them are cluttered with functions that have little or nothing to do with electronic mail.

The Gnome-based Ximian Evolution [5] program is just one of these multifaceted offerings, aimed at Microsoft Outlook users, who appreciate the similarities between Evolution and their former Windows-based environment.

There is a downside to Evolution. As Evolution needs to emulate the whole range of Microsoft Outlook features, the application is quite bulky. Linux still aims to uphold the old Unix KISS paradigm, and this is one of the reasons why ASCII mailreaders such as Mutt [6] are so successful.

Balsa

When the Gnome Desktop project [7] realized that an email program was necessary back in 1997, Stuart Parmenter started working on Gnome Balsa [8].

Pawel Salek took over the coordination of the project as of version 0.7, and 2001 saw the release of Balsa 1.0. This version already offered sought after features such as multiple identity management and HTML-based mail. Support for Brian Stafford's libesmtp library was added in 1.2. Thanks to Albrecht Dress, version 1.4 was the first to support GnuPG [9]. Although there is a 2.x branch by now, Balsa 1.4.4 ist still quite widespread.

Version 2.0 saw a move to Gtk 2, and the feature list for the 2.2 release includes a new MIME interface called GMime, improved IMAP support, S/MIME and Gtk 2.4 file dialogs. Pawel Salek, Carlos Morgado, Peter Bloomfield, Emmanuel Allaud, Albrecht Dress, and a number of volunteer helpers are currently involved in this work.

The Balsa mail program is integrated with the Gnome desktop, although it can be run independently of Gnome.

The developers' aim has always been to keep as close as possible to email standards. Pre 2.0 Balsa versions used libmutt, a Mutt derivative by Alan Cox, to achieve this. Balsa version 2.0 and later uses GMime by Jeffrey Steadfast, as this library provides support for GUI-based programs with multiple threads.

Another of the programmers' major concerns is maintaining compatibility to other mail programs running either on the local machine or on other machines on a network. Finally, the developers want to restrict Balsa to basic requirements, and a few cool features, to avoid an overloaded and thus slow program.

Back-end side, Balsa supports a variety of local mailbox formats, POP3 and IMAP. According to Steffen Klemer, one of the developers, Balsa's flexible mail filters, cryptography and LDAP support, integrated spellchecking, and the variety of user-configurable preferences are Balsa's strongest points. All of these features are configurable via the GUI-based front-end. Also, Balsa supports multiple sender identities per account, mailing lists, and signatures.

As Balsa is a Gnome component, it comes as no surprise that the program is distributed as a free application under the GNU General Public License (GPL).

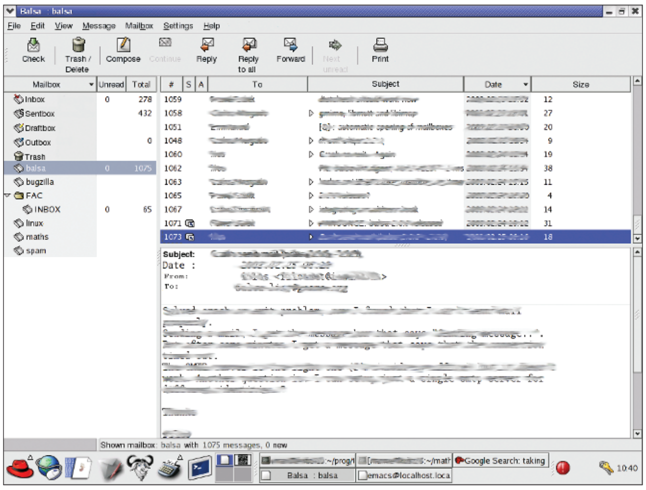


Figure 1: The Balsa program is just one of many email applications for Linux. Balsa is integrated with the Gnome desktop, although it will run in other environments.

Spam: Kill or Cure

As most people already know, spam is more than just a nuisance. It has led some users not to read their email, rather than run the daily gauntlet. Most countermeasures prove ineffective after a short while. It is an arms race that consumes resources which could be put to better use. With spam on the increase, there is a parallel surge in defensive techniques, but some are grotesquely unintelligent.

As an example, I recently received an automatic response from a person I had never heard of before, stating that he had just received a message from my address, and that I should reply by mail to confirm that my message was genuine. Failure to do so would lead to any future messages from my address being blacklisted and automatically deleted.

As the sender of the message had obviously received a message that used my email address, I had two alternatives. I could send a reply, which would mean that the victim would be subjected to even more spam with my spoofed address, or I could do nothing, which would mean that I would never again be able to communicate with this person. This is a classic example of a totally ineffective spam filter that, in the long run, will prevent any messages at all from getting through to the correspondent.

Email systems allow the recipient to send an error report to the sender, if the sender has attempted to send a message to a non-existant address. Unfortunately, spammers tend to spoof genuine addresses that they grab off the Web or Usenet in their messages. In this case, the response will only reach an innocent third party who has nothing to do with spam mail. Thus, a single spam message can impact two Internet users' mailboxes at the same time.

Many users configure their systems to delete potential spam mail without prompting them first. This prevents the spammer from learning whether the email address actually exists.

Diagnostic systems that were useful before the spam era have now become useless. Many Internet users simply delete diagnostic messages without reading them. It can be quite difficult for a user to filter out a real message from the thousands of diagnostic messages.

Spam has had a lasting effect on the Internet. Where users previously assumed “no reply = probably arrived”, the typical assumption nowadays is “no reply = probably trashed”.

Blacklists Are Bad

Blacklists are probably the most destructive and questionable form of anti-spam protection. A blacklist rejects messages from specific IPs. Many Internet users have dynamically assigned IPs, this means arbitrary discrimination against innocent users. Outlawing people from a communication channel is just like depriving them of basic human rights.

Spammers typically used cracked machines belonging to innocent victims to deliver their dubious messages. For the owners of victim machines this is tantamount to a fait accompli, as nobody can warn them about their communication channel being blacklisted.

Blacklists are an instrument of arbitrary censorship. Users get used to the fact that email may not reach its destination due to a combination of blacklists and other filter mechanisms. If political censorship were to happen, we probably wouldn't notice the difference.

Recently, Colin P. Fahey published a far more detailed discussion of this issue online [10]. He suggested introducing a code system for email authentication that could be implemented on top of existing mechanisms, thus avoiding any changes to the existing infrastructure.

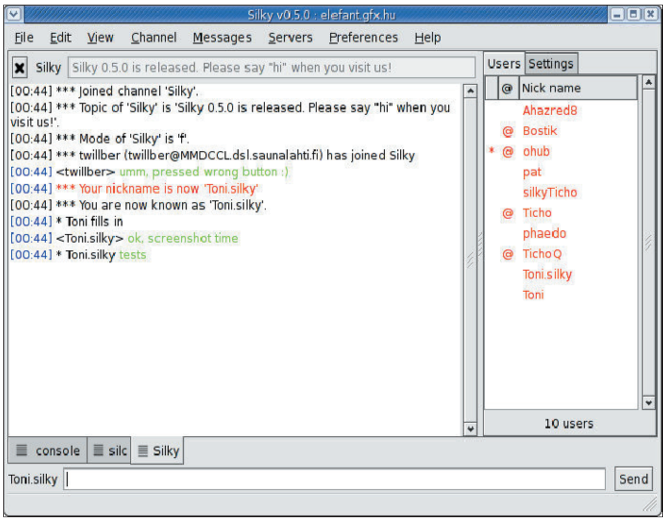
Two correspondents could use the system to agree on a shared secret, which would then be used to generate a code that no spammer could guess. The sender-side mail client would need to insert the key, allowing the client at the receiving end to validate the code, and thus check the message for spam.

This sounds quite promising at first, but it has a downside. Before exchanging messages, two correspondents would need to exchange keys, and this would mean major changes both for users and to email programs.

It makes more sense to use OpenPGP [11] to sign mail. This provides the same benefits (that is an automatic method for distinguishing beween spam and genuine messages). At the same time, the method is more secure, and allows for digital signing and encryption of mail traffic in general.

Onus on the Provider?

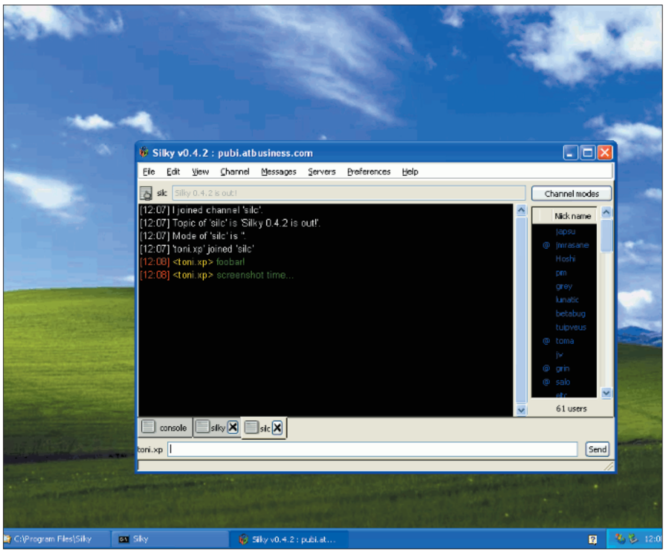
The regular readers of this column may remember me writing about a provider called UK Free Software Network (UKFSN). UKFSN requires customers to agree to a fee of £150 per recipient of spam messages from their accounts.



Silky 0.5.0 is released. Please say "hi" when you visit us!

```
[00:44] *** Joined channel 'Silky'.
[00:44] *** Topic of 'Silky' is 'Silky 0.5.0 is released. Please say "hi" when you
visit us!'.
[00:44] *** Mode of 'Silky' is 'f'.
[00:44] *** twillber (twillber@MMDCCL.dsl.saunalahti.fi) has joined Silky
[00:44] <twillber> umm, pressed wrong button :)
[00:44] *** Your nickname is now 'Toni.silky'
[00:44] *** You are now known as 'Toni.silky'.
[00:44] * Toni fills in
[00:44] <Toni.silky> ok, screenshot time
[00:44] * Toni.silky tests
```

Figure 2: The SILC realtime protocol has a similar structure to IRC, but implements strong cryptographic features. The Silky SILC client is the first GUI-based SILC program.



Silky 0.4.2 is out!

```
[12:07] I joined channel 'silc'.
[12:07] Topic of 'silc' is 'Silky 0.4.2 is out!'.
[12:07] Mode of 'silc' is ''.
[12:07] 'toni.xp' joined 'silc'
[12:08] <toni.xp> foobar!
[12:08] <toni.xp> screenshot time...
```

Figure 3: The Silky chat program supports encrypted connections on Linux, Windows and Mac OS X.

Extremely low costs that allow for profits even with a poor response ratio are the main motivation for spammers. UKFSN effectively eliminated the benefit by levying a fee, causing spammers to lose interest in UKFSN.

If more providers were to adopt this posture it would definitely help. It might also help to replace hardline black or whitelists by greylists that rate providers on their spammer-friendliness.

The use of digital signatures, based on GnuPG [9] for example, is also effective. We need to rethink our use of these technologies, as larger companies, absurdly, tend to prefer unsigned email.

It is also conceivable that the spam problem might finally be resolved by introducing a new mail transport protocol. This is what the author of the QMail mail server, D. J. Bernstein, suggested.

It might take a while to finally resolve the spam issue. Until that happens, there are at least one or two gleams of hope, such as the successful implementation of “Anti-Spam haikus” by Habeas [12].

Silky

IRC (Internet Relay Chat) is popular with many users despite the widespread availability of similar applications, such as Instant Messenger, ICQ and the like.

Many Internet protocol developers failed to put enough effort into securing their protocols, their major concerns being functionality and performance.

Even though spam in IRC is not a major issue, the danger to privacy on the Internet continues to grow. Not only from enterprises collecting profiling data, but also governments.

This is particularly critical in countries where governments do not tolerate dissidents, and an unconsidered statement can lead to reprisals.

If you simply want to avoid everything you type landing in NSA repositories, or if an unconsidered statement could endanger you, cryptography provides the answers to safeguarding your privacy.

SILC (Secure Internet Live Conferencing) [13] is a cryptographically enhanced IRC counterpart. The protocol allows programmers to develop chat systems such as IRC and Instant Messenging solutions. The GUI-based Silky [14] client implements the SILC standard and is GPL-licensed.

Toni Willberg launched the Silky project in 2003, as there was no GUI-based SILC client around at the time. Volunteers and developers have translated it into eight languages. The program is usable in parts, although some features may be missing. Toni refers to the release status as “somewhere between alpha and beta”, and anticipates version 1.0 before the end of summer 2004.

Toni is looking for developers to help out. They need experience of C programming, Glib and Gtk + skills, as Silky is based on these libraries. The program integrates with Gnome, although it will run in other environments as it does not use any higher-level Gnome libraries.

In Toni’s opinion, the project’s strong points are its cryptography and the explicit commitment to user-friendliness. Cryptography between clients allows message exchanges without the server admins being able to read them.

Linux for Audiophiles: Agnula

The Agnula project [15] aims to compile a GNU/Linux distribution for professional audio users, based either on Debian (Demudi) or Red Hat (Rehmudi).

The project was supported by the European Commission and the official closing meeting took part in Brussels, in March 2004, culminating in a demonstration of Agnula Version 1.0 by its developers. As far as the Commission is concerned, the Agnula project has been completed. Agnula has built up a Community over the years, and the project is still under active development.

Agnula Trademark License

To keep things this way, FSF Europe introduced the Agnula Trademark License [16]. Representing the former Agnula consortium, FSF registered the Agnula Trademark in the EU, and developed an appropriate license.

The idea behind the license is to allow free use of the trademark, in line with the basic premise of free software, providing that users uphold the principles of the project and the Agnula consortium, and distribute free-only software.

This provides users with the knowledge that Agnula distributions or products will be free of proprietary software, no matter what distributor they come from. Commercial distributors are welcome to publish their own Agnula distributions and product lines.

Instead of using the trademark to restricting free trading, the developers decided to use it to allow free trading, as long as all the players keep to rules. This is important as commercial free software is often placed on a par with proprietary software in all aspects, thus restricting comparisons to purely technical criteria. However, this is discriminating against commercial free software, as in contrast to proprietary software, free software guarantees the freedom and independence of customers, enterprises, and society as a whole.

The Agnula trademark looks set to remove this discrimination, at least partly, providing companies that distribute commercial free software with a fair opportunity to do business without placing them at a disadvantage.

That’s all for this month. Please send your ideas, suggestions, comments, and questions by mail to [1]. ■

INFO

[1] Send your ideas, suggestions, and comments to Brave GNU World: *column@brave-gnu-world.org*

[2] GNU project homepage: *http://www.gnu.org/*

[3] Georg’s Brave GNU World homepage: *http://brave-gnu-world.org*

[4] “We run GNU” initiative: *http://www.gnu.org/brave-gnu-world/rungnu/rungnu.en.html*

[5] Ximian Evolution: *http://www.ximian.com/products/evolution/*

[6] Mutt mail client: *http://www.mutt.org*

[7] Gnome: *http://www.gnome.org*

[8] Gnome Balsa: *http://balsa.gnome.org*

[9] GNU Privacy Guard (GnuPG): *http://www.gnupg.org*

[10] Article “Spam: The Phenomenon”: *http://www.colinfahey.com/spam_topics/spam_the_phenomenon.htm*

[11] OpenPGP: *http://www.openpgp.org*

[12] “Habeas win $100k judgement against spammer” *http://www.theregister.co.uk/2004/04/07/habeas_spam_lawsuit/*

[13] Secure Internet Live Conferencing (SILC): *http://www.silcnet.org*

[14] Silky: *http://silky.sf.net/*

[15] The Agnula project: *http://www.agnula.org*

[16] Agnula Trademark License: *http://www.fsfeurope.org/projects/agnula/license.html*