

Chris Forno

Skills

Languages	Advanced:	PHP, HTML, CSS, SQL, RegExps
	Intermediate:	Haskell, Lisp, C/C++, Bash, Perl, Ruby
	Basic:	Python, Objective-C, R, JavaScript
Skills	Advanced:	Technical Documentation, Automation
	Intermediate:	Statistical Analysis, Project Management
	Basic:	Interface Design, Software Testing
Tools	Advanced:	Emacs, CVS, SVN, T _E X/L ^A T _E X, Graphviz
	Intermediate:	vi, Darcs, Make, MySQL, PostgreSQL
	Basic:	Maxima, gnuplot, OpenGL
Networking	Advanced:	Linux, iptables, Apache, encryption
	Intermediate:	BSD, OS X, DNS, qmail, SSH
	Basic:	IPSec, SELinux, PAX

Professional Experience

Aug 2007–current	Application Developer , deviantART Inc.	http://www.deviantart.com/
Nov 2006–Aug 2007	Application Developer , TG Publishing	http://www.tomshardware.com

Responsibilities: Extending the Content Management System, working with the creative team to create new advertising “products”, integrating the work of external software contractors.

Projects: The first project was to convert a new web design to HTML/CSS/Javascript for the main site. Next, I extended the Content Management System to support the new design. In April 2007 we were purchased by Bestofmedia Group. I spent 3 weeks in France working with their engineering team. I wrote an XML-RPC interface to the CMS to allow France to pull our content into their CMS. To give Bestofmedia visibility into our bandwidth usage, I installed monitoring software onto our 40 FreeBSD servers.

Initiatives: I installed and wrote articles for an internal documentation wiki (the Bestofmedia team later switched from their documentation system to ours). To improve our search engine rankings, I added a Google sitemap generator to the CMS.

Technologies: The CMS was written in Perl and XSLT and used MySQL for its database. I also worked with PHP, Ruby on Rails, Javascript, HTML, and CSS. I wrote the XML-RPC interface in Common Lisp. The documentation wiki ran on MediaWiki.

Conclusion: TG Publishing had a hacker culture (strongest in the hardware testing labs). Someone was almost always working on an interesting project, whether it was a solar-powered Nintendo Wii or testing the cooling properties of beers from around the world. But most importantly, everyone was given individual responsibility and accountability for their projects and the freedom to use their own judgement and creativity. Unfortunately, that culture began to disintegrate after the acquisition.

Jun 2006–Oct 2006 **Network/Security Engineer**, The Yucaipa Companies (contracted via Robert Half Technology)

Responsibilities: Security was my team's top priority. We maintained a nationwide network of servers, audited network security, and set up the networks and equipment for new offices. We documented all the existing systems and new systems we were putting into place and planned for eventual worldwide expansion.

Initiatives: I found a method to link multiple remote servers via IPSec tunnels (an open problem that predated my arrival). When a computer's partition table crashed, I modified a disk recovery program so that it could recover the LUKS encrypted partition. Soon after my contract began, I saw the need for, planned, and co-designed a company-wide contact and marketing database, which led to the expansion of our department.

Technologies: The firewalls used Netfilter rules (up to 5,000+ for a single network) generated by iptables and Bash scripts. For network diagnostics, I used nmap, rsync, netcat, tcpdump, IPSec and others tools. Each server ran Hardened Gentoo Linux with a customer kernel and disk encryption. The documentation wiki ran on MoinMoin. The marketing database was a mix of PostgreSQL and Microsoft Access.

Conclusion: Yucaipa had a very strong team environment. Security was our highest authority, and as such we didn't directly report to anyone. Instead we were responsible to the owner for the integrity and availability of the network. This meant that the 3 of us brainstormed together, relied on each other, and shared responsibilities based on our individual strengths. When my contract ended, most of the office expansions was complete and the team had gone into maintenance mode. We mutually agreed not to extend my contact.

Mar 2004–Apr 2006 **Chief Information Officer**, Smoke Free International

Responsibilities: I was the first employee of this startup. In the beginning, I did anything that needed to be done. As time went on, I focused on building systems to manage the business' information. During live trainings (the company's product), I ran the sound and projection systems. When the CEO was out ill for 3 months, I took over his responsibilities.

Projects: Coordinated the typesetting, printing, and mailing process for direct mail marketing (50,000–200,000 7-piece packets each run) and did so each time under budget. Worked with the sales to keep track of both our sales and customer databases. Helped write, edit, layout, record, and assemble training materials. Maintained the customer support web portal.

Initiatives: I built the company's website from scratch. Similarly, I setup email, a VoIP phone system, a shared filesystem, and automatic database integration with an off-site call center.

Technologies: Customer/Sales Management: ACT!. Training Materials: Microsoft Word, PowerPoint. Communications: Debian GNU/Linux, Apache, Plone, exim4, samba, SIP.

Conclusion: The founder of the company and the environment he created were very unique. I've never seen another company like it. Each employee had an "unlimited" personal development budget. The focus of the company was on improving the lives of the employees just as much as the customers. I left reluctantly to pursue more technical work (where I felt my calling more strongly).

Personal Projects

gressgraph <http://jekor.com/gressgraph/>

Synopsis: gressgraph produces a graph of an iptables firewall ruleset using Graphviz. I wrote it to help me reason about complex firewalls.

Technologies: I wrote the program in Haskell to test the claims of pure and lazy functional languages. I also used a literate programming style to test the benefits claimed by Donald Knuth. I was pleased with the results of both. This ended up being the shortest and clearest program I've written to date, proving easy to modify and debug.

Gt <http://jekor.com/Gt/>

Synopsis: Gt was my most ambitious project. It's a low-level graphics library written for the Ecclesia RPG (video game). It provides geometric drawing operations, text output, and window management.

Technologies: Written in C++, it's a layer on top of the SDL and FreeType libraries. Additionally, I wrote some PHP and HTML for ecclesia.sourceforge.net. Gt was my first project meant for wide distribution and included a autoconf build system, class documentation (Doxygen), used a CVS repository, and was hosted on SourceForge.

rubirc <http://jekor.com/rubirc/>

Synopsis: I was very active on IRC from 1997 until 2002. After using and scripting several IRC clients, I had a different idea about how IRC clients should work (especially for scripting). rubIRC was based on the idea that to in order to create a sufficiently powerful scripting system, you should create the scripting system first and then write the IRC client using it (rather than the other way around). (It was 4 years later when I discovered Emacs and its take on this same philosophy.)

Technologies: Ruby, GTK+, FOX, regular expressions, test-driven development.

batchid3v2 <http://jekor.com/batchid3v2/>

Synopsis: I wrote batchid3v2 to help me organize my music collection. At the time I knew of no Linux tools that supported id3v2. It was a short project but an interesting one because of the low-level bit pushing it involved.

Technologies: C++, binary formats.