

Six Seven ways to differentiate your resume from those of other recent graduates.

I tried for seven as a Covey thing... wound up with ~~six~~ seven...finally removed the ~~must have put~~ the sand ~~in first~~.

My opinions only, your mileage may vary. —J.D. McCown 3/00, updated 3/01, 3/02, 2/04, 2/06, 2/07

This document is intended to point out paths to core skills in a few areas which may be “unusual or interesting” to hiring managers—and could well be stimulating and beneficial to the student. There is a distinct (and significance) difference between gun-decking a resume, and strategic preparedness. Working through the materials and processes listed below should result in a functional familiarity with the subject area, and quite possibly could serve as a springboard into “more interesting” areas of specialization.

Representing personal knowledge and background to a prospective employer is an area requiring tact and truthfulness. A candidate who completed (I) below “Oracle familiarization” probably would have little basis in fact for representing themselves as an “Oracle DBA”, but could in good conscience represent that they had worked with Oracle version X.Y on platform Z (possibly “at home”), had experience installing, configuring and using the tools which came with it.

It is important (in my view) to adequately distinguish between:

- a reading familiarity with the subject area
- a brief interaction with the subject area (tutorial, demo)
- training/instruction/coursework
- in-depth wrangling with the subject
- original research/published work

There is no harm at all in declaring a ‘reading familiarity’ with a technology area, particularly if it comes up in an interview—and if you are prepared to disclose what is known (don’t fabricate). There is a difference between “three years experience” and “three semesters of class” and “three projects over three semesters.” All of them do satisfy the binary “has experience with subject”, but to varying degrees.

Playing “buzzword bingo” is dangerous, unless the interviewer knows less about the subject than you do—whereas being forthright about your degree of knowledge and experience can avert a wide range of potential disasters.

- I. Consider adding the magical **Oracle** item to your resume
 - Buy *Oracle for Dummies* (It isn’t Knuth, but it isn’t a bad book... like Cliff Notes)
You don’t need to tell anyone what text you used, particularly during an interview.
 - Download a trial of Oracle 10g (was 8i) from <http://technet.oracle.com/>
 - **Oracle will give you the software to play with – they want you to do so**
 - Basics/familiarization
 - Install the product (defaults are ok)
 - Do the exercises in the book (are easy)
 - Intro DBA
 - Acquire: *Learn Oracle in 21 Days* (published by SAMS Technical Publications)
 - Work through the DBA exercises
 - Check out the application server and other bundled software (Java friendly)

If you require additional challenges and/or resume bulk IBM’s DB2 for Linux is available free for personal use. <http://www14.software.ibm.com/webapp/download/category.jsp?s=c&cat=data>
IBM has made **huge** resources available to “developers” (you) for free. Dig in. Their *Red Book* series in particular contains fast technical reads in many interesting areas. Check out the SEEK (Software free eval kit CD: Rational (!), Tivoli, Lotus Notes, DB2)

MySQL and *PostgreSQL* DBMS systems are very common (and free), although both have

complex features, they are “at the core” functional SQL DBMSes. **Recommendation : Explore both, be able to say “We could use either one, they both work.”**

- II. If your school participates in MSDN Academic Alliance be sure to get set up on developer software while you are still a student. Buying VS.NET, Visio or MS-Project retail is harsh. [Dr. Chase here: **Messiah does, and I used to think that you had to delete it when you graduated. You don't. As long as you acquire it as a student, you may continue to use it forever for non-commercial purposes. Skill development to get a job is non-commercial.**]
- III. If you don't currently have familiarity with a **Unix-family operating system**, add at least one and two if you have time (in order of my preference):
- A Linux:
 - <http://mirrors.kernel.org/> : Fedora Core, SuSE, OpenSUSE, Red Hat, Ubuntu
 - White Box Enterprise Linux (WBEL) is a FREE report of Red Hat Enterprise Linux <http://www.whiteboxlinux.org/>
 - Knoppix is fun, boots from CD does not require an installation, great way to get your feet wet without repartitioning your hard drive.
 - Solaris (<http://www.sun.com/solaris/freesolaris.html>) Free Download for x86
 - Additional essential software (compilers etc.) available for free download at: <http://www.thewrittenword.com/>
 - FreeBSD 5. Docs & Howtos: <http://www.freebsd.org/> (are also on the CD)
 - Acquire *Unix in a Nutshell* from O'Reilly and Associates <http://www.ora.com/> ORA publishes a wonderful variety of “get clueful quickly” books. Spend the money.

Tip: Both Linux and FreeBSD are workable on ‘less than current’ hardware, your parents’ old clunker in the attic might be just fine for this kind of thing.

Things to do:

- VMware Server is **FREE** – virtual machines are great for learning - http://www.vmware.com/products/free_virtualization.html
- Install the operating system, preferably several times until you are comfortable with what the various options are available during the process. (Solaris in particular has endless option paths)
- Configure **IP networking**.
 - Learn how to change network addresses, routes, DNS etc. after the system is initially configured.
 - Learn how to do so with or without the GUI (oooooh!)
 - ping, netstat, ifconfig, arp.... these are your friends.
 - Time invested learning diagnostic procedures is invaluable
- Become at least minimally functional with **vi**. It's ugly, but it is on every unix box in the world.
- Understand how **man pages** work (there are people who never get this)
- Understand how to compile a ‘hello.c’ type program on the platform
- Become familiar with the ‘stock’ software installation procedures for each platform:
 - Red Hat - “RPM” Redhat Package Manager
 - FreeBSD - `pkg_add`
 - Debian – ack! `apt` **still** confuses me, Synaptic on Ubuntu is training wheels version
 - Solaris – `installpkg`
- Install APACHE <http://www.apache.org/> **the** webserver
- Install SAMBA <http://www.samba.org/> which allows unix boxes to serve files in a Microsoft networking environment.
 - Configure file shares, experiment with user rights
 - If you have access to an existing MS environment, experiment with authenticating through the MS domain
 - Think about the licensing implications of how this works

- Become familiar with the **tar** command... it's old and cruffy, but it is the least common denominator for file archiving and transfer.

IP Networking (General):

If you have the money and are into "authoritative" works on things, W. Richard Stevens *TCP/IP Illustrated (series)* is **the** work on TCP/IP networking. Volumes one and two in particular are an excellent "reading education" and a shelf reference as well.

VOIP

- Check out Asterisk*/Trixbox - A real PBX with all the features <http://www.asterisk.org/> <http://www.trixbox.org/>
- Asterisk@home – "click to install" Asterisk ISO distribution with autoattendant music-on-hold and web-UI too (Caution: the installer wipes the drive – use an old/disposable machine)
- Check out VOCAL - <http://www.vovida.org/> - a Cisco VOIP implementation - Very good documentation and tutorials
- You thought Vonage was cheap? <http://www.freeworlddialup.com/>
- SJ Labs Softphone: <http://www.sjlabs.com/sjp.html>

IV. Intrusion Detection/Forensics [2006: tools still there – Snort, commercialized by Sourcefire]

Intrusion detection is a major buzz in the network security space. Even though most administrators do not bother to read the logs *already* generated by servers, firewalls, OSes they currently use, there seems to be major drive for corporates to install IDS.

Be advised: Using these tools in "someone else's network" is a problem—get permission first.

Become familiar with "network monitors" (commonly called sniffers, which is trademarked by NAI, and thus not what I'll call them that here)

- *tcpdump* <http://www.tcpdump.org/> available for most unixes and ships with both Linux and FreeBSD. With some effort is available on NT known as *windump*.
- *ethereal* <http://www.ethereal.org/> available for **both** unixes and XP, is **free** and actually quite pleasant to use on XP (if you have the horsepower). Wins the 'nice GUI' award in the free sniffer category
- *sniffit* (check a favorite search engine) Was once the favorite monitor of many, has an interesting ability to watch text traffic in a selected session.
- *Dsniff* <http://www.monkey.org/~dugsong> This (along with most of Dug's tools) demonstrates the insecurity of lots of things at once. DSNIFF grabs passwords out of just about any unencrypted session, it makes a nice demo. Other tools on his site are worth a look, but probably do not have "white hat" applications.

Projects:

- Set up a network (if you can't borrow one).
- Demonstrate how to display network traffic at the various decode levels supported by the product of choice
- Establish how to 'record' session traffic for later review, especially ability to track only particular protocols or hosts.
- Demonstrate effectiveness of DSNIFF (with the permission of the network owner)

Download a copy of SNORT (<http://www.snort.org/>) a free IDS (runs best on BSD unixes, almost ok on Linux). Be sure to download the latest signatures **and** the whitepaper on how to write detection signatures.

Project:

- Set up a snort system on an 'in use' network
- Evaluate reporting over a week long period
- Correct/modify rules to reflect anomalies in reporting

- Develop/test your own rules to detect anomalous events

V. Cryptography

Acquire *Applied Cryptography*, by Bruce Schneier (looks good on a shelf, is large and red)

- Section one *Foundations* is probably review, be sure that it is.
- Carefully study Section I chapter 2 (Protocol Building Blocks) and chapter 3 (Basic Protocols) and the part of chapter 4 dealing with Key Escrow.
- The remainder of Section I is interesting, but can be skipped for now and reviewed later.
- Section II, the core “background knowledge” of the applied cryptography, much of it review.
- Section III, Chapter 11 is the mathematical background for the problems and technologies of current cryptography (interesting).
- Consider the rest of the book to be reference material.

Things to do:

- Understand the applications and limitations of hashing functions
 - MD5, SHA-1, RIPE-MD
 - How are they different from simple checksums or CRCs?
 - Understand how an HMAC application of a one-way-hash differs from the basic application.
- Gain a “reading” feel for the similarities and differences between: DES, IDEA, Blowfish, CAST, RC4.
- Read <http://www.cs.auckland.ac.nz/~pgut001/tutorial/index.html> then read the rest of Peter Guttman’s site (a concise crypto education for free)

How does key-length relate to security?

How does one know whether an algorithm is trustworthy?

Why do people trust SSL? What doesn’t it do....

What is a Feistel network ? (almost trivia, but good stuff)

[2006: How broken is SHA-1 ? Can you explain the risk?]

Practical application in PKI:

- Download OpenSSL <http://www.openssl.org/> <http://www.openca.org/>
 - Become familiar with the command line tool options to encrypt/decrypt and generate hashes in all of the supported algorithms (which is very useful on its own....)
 - Using the supplied CA programs, set up your own Certificate Authority and issue some certificates. (not rocket science... may take 15 minutes to learn oddities of the program)
 - Obtain and load a (real + free) certificate into your favorite S/MIME email client. http://www.trustcenter.de/en/produkte/my_certificate_express.htm
 - Issues a certificate from OpenCA and load it into an Apache (or IIS) web server. Makes you feel really secure doesn’t it?
 - Check out Microsoft Code Signing (not just for web apps) and Software Restriction Policies - Interesting security implications.
 - <http://www.microsoft.com/technet/prodtechnol/winxpro/maintain/rstrplcy.mspx>
 - <http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/ServerHelp/9c25d487-eb0b-4e2d-a5f7-89b2686d6a69.mspx>
 - <http://support.microsoft.com/default.aspx?scid=kb;en-us;324036>
- Who signs the root? Do you trust **them**? Why?

VI. Internet Security (General) – I’ve worked with these guys, it helps to know smart people.

Avolio Consulting reading list: <http://www.avolio.com/resources.html#links> (see “papers”)

Marcus Ranum http://www.ranum.com/security/computer_security/index.html (cluefulness on caffeine)

Mich Kabay <http://www2.norwich.edu/mkabay/> (Check the security lectures, papers, ppts ...)

Familiarize yourself with the “vocabulary of compliance” (ie: standards with which business must comply) You don’t want (necessarily) to *be* the auditor, but you will do well to know what they’re talking about:

PCI – Payment Card Industry security standard:

<https://www.pcisecuritystandards.org/>

SOX – Sarbanes-Oxley – Security requirements for business

<http://www.developer.com/security/article.php/3320861>

HIPAA – Healthcare (bla bla) security requirements

<http://www.developer.com/java/ent/article.php/3301011>

GLB – Graham Leach Bliley – Financial (bla bla) security requirements

http://www.securitymanagement.com/library/gramm_tech0902.pdf

VII. Reverse Engineering

Make a firm decision on your ethical stance hacking, malware, reverse-engineering, responsible disclosure and where your limits might lie if someone threw “crazy money” at you to do something unethical or illegal. Think about this before digging in.

Resources

1. Intel processor manuals – Intel will happily send these to you *free* since they like people to use their products
http://www.intel.com/design/pentium4/manuals/index_new.htm
2. “Under the Hood” columns from the former Microsoft Systems Journal – now located in MSDN <http://msdn.microsoft.com/> Search for “Pietrek” (author Matt Pietrek) and start reading his articles from 1997 forward.
3. Sysinternals.com (Mark Russinovich + Bryan Cogswell)
 - a. <http://www.sysinternals.com/> Download the toolset
 - b. Buy the book <http://www.sysinternals.com/WindowsInternals.html>
 - c. Watch this webcast (is Mark Russinovich presenting on rootkits)
<http://www.microsoft.com/events/EventDetails.aspx?CMTYSvcSource=MSCOMMEdia&Params=%7ECMTYDataSvcParams%5E%7Earg+Name=%22ID%22+Value=%221032274950%22/%5E%7Earg+Name=%22ProviderID%22+Value=%22A6B43178-497C-4225-BA42-DF595171F04C%22/%5E%7Earg+Name=%22lang%22+Value=%22en%22/%5E%7Earg+Name=%22cr%22+Value=%22US%22/%5E%7EParams%5E%7E/sParams%5E%7E/CMTYDataSvcParams%5E>
 - d. Explore this list of tools <http://www.solsem.com/lab-setup.txt> - Amazing what you can get legitimately AND for free.
 - e. If you have extra cash: <http://www.sysinternals.com/troubleshooting.html>
4. Acquire Greg Hogg’s book on *Rootkits*
<http://www.amazon.com/gp/product/0321294319/102-2232589-4484965?v=glance&n=283155>
Note: While this book is an essential on getting started quickly in understanding rootkit technology, it also (by its publication, and by Greg’s other activities) served to make rootkits more of a problem. I regret the implied endorsement of the book by listing it, however ‘the bad guys’ already know this material—and you should too.
5. Acquire Peter Szor’s book *The Art of Computer Virus Research and Defense*
Note: the whole book is worth reading, but the last five chapters are the most useful. Some of the history and “I was there” information in the front half can become a bit tedious—but “he was there” so it is legitimate.
<http://www.peterszor.com/>
6. Read Joanna Rutkowska’s papers <http://invisiblethings.org/> Be *very* afraid.
7. Google for ORC+ – Old Red Cracker -- have your AV up to date, be careful – Interesting files on interpreting tricky i386 asm.