BASIC PC SECURITY

A *Ciphers By Ritter* Page

Terry Ritter

A = ritter B = ciphersby
A@BA.com

2009 October 30

> **MY CONCLUSIONS:** *Microsoft has bumbled into an epic fail: The Windows environment is dangerous for on-line purchasing and banking, even when patched and with reasonable user attention. To reject malware, do not browse the Web in Windows; use some other operating system (OS). To prevent infection, boot your browsing OS from CD or DVD.*

There is a Web war on, and we are losing. Distant criminals can and do subvert the computers in our homes for their own profit. When criminals get control, they can distribute advertising, expose our passwords and data, find our personal information, and do whatever they want on our computers. When a Web computer is infected, anything on it can be exposed, changed or deleted remotely, so even "removing" the malware may not put things right.

We do not have to give up the Web. We do have to give it some attention.

MY CONCLUSIONS

Wikipedia reports that about 93 percent of browsing occurs from a Windows operating system (OS). Even though Microsoft effectively owns the browsing environment, Panda Security has estimated that 58 percent of US PC's are infected with malware. If that is true, then Microsoft is responsible for an environment that has failed to protect *most* users. Panda also reports,"The huge amount of Trojans in circulation is due to the spectacular increase in the number of banker Trojans aimed at stealing user data." Malware makes on-line banking risky for most of us.

I tried to find easy ways to protect PC's, but instead found that the usual approaches were not nearly enough. To get real security, we have to make some fairly substantial changes in the way we do things. For those willing to put in a little thought and effort, using the Web can be made far more secure.

Since I came at this as a Windows guy, my first recommendation comes with some sorrow:

1. To reject malware, use an operating system (OS) other than Windows for browsing.

I suggest Puppy Linux, because:

- Puppy Linux is not Windows.
- Puppy Linux fully supports the Firefox browser.
- Puppy Linux is free.

2. To prevent infection, boot your browsing OS from CD or DVD.

I suggest Puppy Linux, because:

- Puppy Linux is small and loads quickly.
- Puppy Linux supports writing browser and add-on updates back to the boot DVD, so they are present on the next boot.

3. To avoid browser weakness, use a secure browsing system.

I suggest Firefox with security add-ons, because:

- Firefox is available for both Windows and Linux.
- Firefox security faults are patched quickly.
- Firefox has an effortless update system.
- New browser security issues can be and are quickly addressed by 3rd party add-ons.
- Firefox is free.

Important Firefox security add-on's include:

- **Adblock Plus** -- hide ads to improve speed
- **BetterPrivacy** -- manage Flash cookies and DOM storage
- **Facebook Secure** -- use SSL for Facebook

- **Force-TLS** -- remembers to use SSL on some sites
- **LastPass** -- encrypted passwords in the cloud
- **Long URL Please** -- exposes target of short URL's
- **NoScript** -- scripting control and other issues
- **Permit Cookies** -- allow specific sites to save cookies
- **SSLPasswdWarning** -- warns when sending password w/o SSL
- **Tab Mix Plus** -- tab recovery after crash (also use Bookmark All Tabs)
- **WOT (Web Of Trust)** -- danger colors on search result links

4. To prevent net snooping, get an SSL connection, especially before entering a password.

The user needs to be aware:

- The "**https://**" in the browser address bar means SSL snooping protection.
- SSL protection must have been established *before* entering a password.
- SSL is included in every browser.

5. To secure your accounts, give each a different long random password, and use a password manager.

I suggest LastPass.com with the Firefox add-on, because:

- LastPass appears to be a secure solution for both Windows and Linux.
- LastPass.com can be used without an add-on, from anywhere.
- The LastPass Firefox add-on can function without a network connection.
- LastPass portable can function outside a browser.
- The encrypted password database can be backed up and archived.
- LastPass is free.

6. To protect against email malware, use web email.

I suggest Google Gmail, because:

- Gmail scans attachments for malware, probably better than we could.
- Gmail has a setting to force a secure SSL connection all the time, thus protecting email contents from network snooping.
- Gmail with SSL supports fairly secure storage of small files by sending them to yourself as attachments.
- Gmail allows on-line viewing of attachments, which should be more secure than downloading to a local file for viewing.
- Gmail HTML viewing is good for exported browser bookmark attachments, since the links can be used directly.
- Gmail is free.

7. Away from home, always boot your own OS from your own DVD.

I suggest:

- Get an old laptop and carry it with you.
- Leave the hard drive at home in an external drive box.
- Take a Puppy boot DVD and use that when away from home.
- Just say no to hotel, bar, bank, conference, customer or good friend computers.

For more lists, see:

- Basic Web Protection
- Detailed Security List

CONTENTS

INTRODUCTION

This project started as a recipe of fixes (free, where possible) to prevent malware problems on a Windows PC. To really fix something we first have to understand it. We can see malware as a wide range of criminal softwares, each exploiting on-line computing weaknesses. No single response will solve all malware problems. Some issues require users to know more, and to operate with purpose. But even with the best user actions, the Windows operating system (OS) includes a range of weaknesses which must be addressed.

Many things can be done, and those are included here, but in my opinion, Windows really cannot be patched into security. Nor can we expect Windows 7 to be more than marginally better, since it is following the same failed strategy. Malware is much worse than commonly thought, both more powerful and more common, and this is happening in a Microsoft planned environment. Current tools are fundamentally unable to solve these problems.

At the risk of claiming "The sky is falling," things are much worse than we think:

- Wikipedia reports that about 93 percent of all on-line browsing occurs on Windows computers.
- cnet news security reports that almost 60 percent of our computers are infected.
- That same article reports a "spectacular increase in the number of banker Trojans aimed at stealing user data."
- eWeek Security Watch reports that infected computers have an average of 13 infected files, from 3 different malware families.
- zdnet reports that the probability of an antivirus finding a recent banking trojan was only 23 percent.
- Computerworld reports that criminal botnet rental fees are dropping due to increased competition.

There will always be a few dummies that get into trouble. But when *most* normal users are infected, something is fundamentally wrong in the product design. If cars were built like this, the government would require their recall.

Our best-known anti-malware tools, firewalls and malware scanners, each have serious practical limitations:

- Firewalls block probes coming from the outside, but cannot help at all when malware is requested from inside. Malware can pretend to be a good process and no firewall will know any different.
- Scanners work from a signature list of malware already found and analyzed. Malware which hides well, or is narrowly distributed, is not going to be on that list.

One of the worst Microsoft failings is the lack of a tool to tell us what we need to know. We should not be working in an infected system, but there is no tool to certify that our Windows system is uninfected. No single malware scanner, nor any set of scanners, can hope to find every possible malware. Yet malware must *change* files in order to infect, and surely Microsoft can tell when one of their OS files has been changed. Hopefully Microsoft can similarly examine or eliminate every possible place for malware in their registry. An "uninfected Windows" certification tool could deliver renewed trust just when that is needed.

When infection is found, the situation changes. After a botnet has been in place for even a few seconds, *nobody* can know what it has done, not even Microsoft. The bot could have downloaded all its friends. It could have made fundamental changes to Windows. The bot could have changed entries in the hosts file, or downloaded false security certificates, and we would not know. After infection, absent some sort of Microsoft deep cleansing which does not now exist, the only real responses are to recover a saved uninfected OS image, or to re-install Windows, which is not easy.

Even if the antivirus says the virus has been removed, a recovery or re-install is necessary *anyway*, because the virus could have installed something that would not be identified as a threat. The absolute requirement to recover or re-install Windows after infection has not been made clear, and has yet to become the conventional wisdom.

Currently, the best protection I know is to use a "live" Linux DVD (like Puppy Linux) for browsing.

BASIC MALWARE RECOVERY

We can accept that a few dummies among us get into trouble. We should not accept that ordinary computer use causes infection more often than not. Something is fundamentally wrong in the product design when PC's cannot stand up to their normal usage environment.

What We Can Do

As individuals, we are not going to be re-designing our computers. But we can use available things and behave in ways that improve our security.

For better or worse, currently the best security advice is to not use Windows on the Web. One alternative is Puppy Linux, because it is small and easy, and supports the Firefox browser. Naturally, Windows is still desirable for the many standardized, professional and even free programs available for it. But on the Web, we get into a browser, and then we pretty much do not care what operating system is being used.

Unfortunately, the majority of us may already have malware, so now what?

Things Should Be Simple

Things *would* be simple if we could just "Restart" the operating system (OS) to get rid of malware. That does not work because malware changes files on the hard drive to get itself re-installed each time. Windows cannot prevent those changes when it has been subverted by the malware. In contrast, booting from a live CD or DVD does restart without malware, every time. ("Boot" apparently is a simplification of "bootstrap loader," which refers to disk-based computers starting with a tiny loading program, which brings in a better loader, which then brings in the OS. The systems thus bring themselves up "by their bootstraps.")

The Old Days Versus Now

In the old days one might find "the" malware file and then examine it to know what it did, and thus know what was needed to unwind the changes. Sadly, the old ways no longer work. Modern malware tends to set up "bot" or robot programs which are criminally remote-controlled from the net. Once the bot has phoned home, anything can be downloaded, including completely different kinds of bot on each computer. Examining the original malware provides no clue about what happened to OS files after malware contacted the mother ship. There can be no guaranteed way to unwind malware changes.

Modern malware can set up a bot and join a herd almost instantaneously on broadband. Malware can hide in a rootkit that Windows cannot see. The bot can pretend to be a valid process. The bot can download a range of low-tech malware so the owner has something to find and remove. Then the scanner says "nothing found," and the hidden bot remains.

Currently, most PC's have malware, and most of those have multiple malware files and families. Just because a scanner finds no malware does not mean there is no malware. Scanners are inherently imperfect, even though many forms of malware can be detected and removed. The problem is that malware which cannot be detected probably is the most dangerous and we do not want to keep that either. Using scanners to certify that a machine is "clean" is asking for more than scanners can do.

Recover an Earlier System

The easiest way to deal with malware is to step back in time before the malware arrived. We can do that by installing an OS image we cleverly made preparing for the future. Before recovering an old image, **first make an image of the infected system** (if possible), so the latest data files might be recovered later. Next, install the most recent image and try that. We may need several tries to get back far enough, yet save as much of the recent environment as possible.

Without saved OS images, we have a problem. Unfortunately, we cannot go back in time to make the images we need. Weekly or monthly imaging to an external hard drive is a very good idea. For noncommercial Windows imaging, I recommend Macrium Reflect Free from download.cnet.com. Businesses might try free PING, or the commercial Macrium Reflect or Acronis True Image.

Reinstall Windows

The ultimate way to deal with malware is to install some OS from scratch. This can take considerable time, in fact days of work to re-configure user programs and the environment. Periodically making OS images is a really good idea. Before reinstalling Windows, **first make an image of the infected system** (if possible), so we later can recover the latest versions of any data files we need.

A helpful strategy is just to not customize the Windows installation very much. Then Windows can be reinstalled in something like 4 hours. As a partial alternative to installing and configuring programs in Windows, one might use a free portable program suite (e.g., LiberKey). When most of your programs are portable on a removable flash drive, re-installing the OS becomes a whole lot easier. The advantage of LiberKey over other portable packages is an extremely effective upgrade process (much like Firefox) covering the hundreds of LiberKey programs. On the other hand, sometimes the programs in Lupo PenSuite are better.

Quick Alternative Access

When a computer is infected, the OS files on the hard drive have been changed so they will start malware on each new boot. Normally, it is not the computer box, or the power supply, or the keyboard, or the display, or the CPU, or the RAM, or even BIOS flash memory

that has been infected, but just the hard drive. In most cases we can safely use even an infected computer just by booting an uninfected and different OS from CD or DVD. A good example would be Puppy Linux.

Many people find it scary to think about actually using a different operating system, because nobody wants to learn another OS. But many of us live in the browser anyway, so using a new OS is not the big deal it used to be. Browsing is the on-line activity at risk, and for browsing the experience is pretty much the same either under Windows or Linux.

"Remove" Malware?

The classic approach to a malware infection has been to scan for a virus and delete it. Removal may still be appropriate for files that have not been executed. However, execution of modern malware generally means the creation of a "bot" or robot program under external criminal control. Even if we find and remove the bot loader, that does nothing about whatever the bot downloaded after calling home.

Scanning a drive for malware may not even be worth doing much longer. Scanning is limited to finding what somebody else has already found, analyzed, and introduced into the signature files. Scanning thus has little hope of finding "zero day" attacks, directed and limited attacks, or modern polymorphic or self-encrypting malware. (See, for example, the SANS risks for September 2009: The Top Cyber Security Risks: "World-wide there has been a significant increase over the past three years in the number of people discovering zero-day vulnerabilities...." "Some vulnerabilities have remained unpatched for as long as two years.")

Let us be clear about this: We can spend whatever time we want scanning and removing, but nobody (and I mean *nobody*) can guarantee a virus-free result by scanning, because antivirus scanning is inherently imperfect. **Scanning cannot certify that a machine is "clean."** If we want a clean machine, there is no choice but to load the OS from a saved clean image, or an original CD or DVD. Malware removal is just no longer appropriate. Others say the same things:

- From Roger Grimes at Infoworld: Starting from scratch is the only malware cure:

  > "If you discover malware on your system, don't mess around. Back up your data, format your hard drive, and begin again."

- From Ask Leo: How do I remove a virus?:

  > "The only way to be absolutely positive that you've removed any and all viruses is:
  > - Backup
  > - Reformat
  > - Reinstall
  > - Update
  > - Restore
  > - Learn."

Rootkit Effects

Modern "rootkit" technologies change an operating system (OS) to hide malware files so they are not even reported by the file system. (See, for example, comments from researcher Joanna Rutkowska in an article from March 19, 2009: Researchers Warn on Security Flaw in x86 Chips: "Today, we cannot effectively detect even the 'traditional' ring 0 (kernel) rootkits....") Rootkit technologies also hide the changed contents of files and report back the original contents so that scans see nothing wrong. Using Windows to scan for rootkit-protected malware probably is a waste of time. When an OS is infected, we can no longer trust the results it delivers, including the list of files it has, and their contents. To expose rootkit-protected files we generally need a "live" CD which always loads a clean OS to report files as they really are.

HOW SHALL WE BLAME MICROSOFT?

It is easy to blame Microsoft for poor security, but hacking competitions have shown that no operating system (OS) can withstand expert attack. Large, complex systems like operating systems will always have serious errors. The criminal goal is to exploit users, and since most users run Windows, most criminal attacks target Windows.

Microsoft has put substantial resources into improving security, including free monthly patches and updates, even for Windows XP products that a customer last paid for 8 years before. Windows users who do not apply free updates are welcome to use a system with known problems, but then surely have no room to blame Microsoft for negative consequences. For example, millions of Windows PC's were infected by the Conficker worm of Nov. 2008 through Mar. 2009. Conficker A apparently started to spread around Nov. 20, 2008, even though Microsoft had issued a free patch for that vulnerability a month before, on Oct. 23, 2008.

As the Windows OS strengthens, attackers have moved toward applications like browsers, players (e.g., Flash), and also file formats (e.g., PDF). Nevertheless, the problems are getting worse, not better.

We Need a Different Philosophy

It is unreasonable to expect complex software to have no errors at all. However, Microsoft *does* seem to have a design philosophy of putting execution ability in everything. We have had issue after issue with ActiveX, Word document malware, Office file format hacks, media format malware, font engine issues, Publisher files issues, video ActiveX issues, and on and on and on. Allowing unauthenticated code to execute on user computers is a design, not a mistake, and it is wrong.

We Need Better Tools

Microsoft seems strangely remiss in not providing tools to check the authenticity of their own OS installation after it has been in place for a while. Surely Microsoft can tell when one of their files has been changed--*and then they can change it back!* Correcting every changed file should end a whole class of malware operation, even if some inactive malware files remain on disk.

The smart part of this is a possibility of absolutely knowing whether or not the system has been infected, knowledge not available from scanning. The dumb part of this is to just re-install every important file, and it is possible that something very much like the current Recovery Console could do that. (See: Langa Letter: XP's No-Reformat, Nondestructive Total-Rebuild Option, Description of the Windows XP Recovery Console for advanced users, and How to install and use the Recovery Console in Windows XP.) Microsoft does not support the Recovery Console as a malware recovery tool for users. Instead, Microsoft prefers to offer users nothing easier than a full Windows re-install.

Providing an equivalent fix for malware in the registry may be more difficult, but Microsoft created that problem. The logic of building a fix may illuminate why the registry is so dangerous that it may have to be restricted (like Autorun).

Microsoft should issue an OS file-and-registry check-and-correct live DVD to verify the existence and contents of each and every Windows file, and correct those which are missing or invalid. A "live" DVD which needs a reboot would prevent hidden rootkits from correcting changed files when scanned. Even with the huge number of files in Windows, validating specific known files should be *vastly* more efficient than scanning all those same files *plus* every other file in the computer for every virus signature.

We Need Hardware Protection

Our PC's are vulnerable largely because they have a tasty hard-drive that is easily changed by malware. Windows cannot protect the drive when Windows has been subverted by malware, so only hardware protection counts, and there is none. There needs to be a way to provide a hardware lock for OS files. OS file changes should not be allowed without prior authentication. The motherboard BIOS, video card BIOSes, and all other flashable BIOSes also need hardware write-protection and periodic verification. (See, for example, the ZDNet Zero Day column for March 23, 2009: Researchers demo BIOS attack that survives hard-disk wipe.)

In practice, the owner would create a local boot lock key at install time. Part of the OS install would set up OS areas to be protected. After install, the drive would lock those areas. Updates would require the owner to enter the local key. The drive itself would have to control write-protect and write-access protection, since we cannot trust either the OS or the BIOS after malware attack. Note that drives currently may know about partitions, but not files.

The new requirement for OS files to not be written in operation probably would require Windows changes. Every other feature of Windows which allows malware to hide on the drive and be started on reboot must be controlled similarly, which also would require Windows changes. Most applications would have a similar installation process and would need similar protection and similar changes.

We Need "Trust" With Consequences

Microsoft has embraced "trust" as a basis for user security. Unfortunately, trust fails to support the user, because users *cannot know*:

- who *really* made or sent a document or programmed a web page,
- whether those they "trust" were careful, and
- whether a "trusted" web page has been hacked.

Users often do not have the information that even experts would need to make a good "trust" decision.

A better approach is to first authenticate items to a particular owner, and then hold that owner responsible. In general, trust without consequences is delusion.

We Need a Better Web

The whole idea of downloading pages that execute code on our machines was fine when everybody could be trusted--which was *never!*

The time has come to design and implement a scheme to provide cryptographic authentication of each item or page downloaded, *before* any code in that item or on that page is executed. Not only should we know to whom we are connected using SSL everywhere and all the time, but also that each item we get from that page was approved by the page owner.

BASIC COMPUTER SECURITY

In computer security, the first issue to think about is data protection. Every hard drive eventually will fail, typically without warning. We cannot stop failures, but we can duplicate our data so that any single failure will not cause loss. Ideally, we will have any important file saved in at least 3 separate and independent drives, devices, and/or services.

Backups

A backup is a copy saved where it cannot be damaged with the original. Users need to get personally involved in assuring their data are backed up. Users can copy their files to USB flash drives, or email-to-self, or perhaps save to a Web backup service. Since it is only necessary to back up new or changed files, backup programs "synchronize" the contents of disk directories. New files are copied, and unchanged files are ignored.

For backing up files to an external drive, I recommend "SyncBack Free".

Operating System Images

An "image" backup is a copy of the installed operating system as it exists on the hard drive, with installed programs and custom configuration settings, and usually the "boot track" and startup code as well. The point of an image backup is to capture the system as it was when the copy was saved. That image can later be used to reproduce the earlier system exactly, thus eliminating any malware which arrived later.

The better imaging programs allow an image to be "mounted" as a drive, making individual files accessible from Windows Explorer. Individual files can be recovered from the image even though they had not been identified as being important enough to have been backed up as files.

For imaging, I recommend "Macrium Reflect Free".

Passwords

A password is a sequence of characters used to identify a particular user. Passwords work by making it unlikely that someone who does not know the password could present it correctly. But computers are able to try one password after another, tens of millions of times, in a "brute force" attack. The typical attack will try "more probable" characters and words before nonsense random values. Any word in a dictionary, or multiple such words, or a sentence of words, may be tried before scanning through random values of much shorter length.

Password protection fundamentally implies that short passwords of any kind can be hacked, as can much longer human language passwords. Having a password hacked by brute force is a **user** failure.

Password security requires **the user** to take responsibility for creating different **long, random passwords** for each site or account or piece of hardware. Good passwords should be machine-generated random sequences of at least 15 characters. Since few if any of us can make or remember long random passwords, we need a password manager to generate and keep them for us.

Our goal is to use passwords that an opponent cannot predict and cannot afford to search. Each password character should be an arbitrary selection from among upper-case alphabetic, lower-case, and numeric characters. With 62 possible choices, each character would represent somewhat less than 6 bits of strength. A good 15-character password would represent a random selection from among 62^{15} possibilities, which is more than 10^{26} possibilities, and more than 89 bits of cryptographic strength.

A 15-character random password is comparable to an 89-bit secret key and perhaps a 1024-bit public key. Much less strength is needed when the password is allowed only a few tries, or when the time for each try gets increasingly lengthy. Preventing brute-force attacks is something all password-receiving devices (or sites) should do, but we probably cannot depend on it.

Password Managers

A password manager keeps all our various passwords in a little encrypted database file that we can keep on a flash drive or in cloud storage or send and save as an email attachment. A password manager allows us to keep any number of long, random passwords and user names and all the site information we want, which even reminds us which sites we have already joined. A password manager allows us access to a range of good passwords using a single memorized password.

Using the password manager in the browser is not a good idea. In 2009, researchers were able to hijack the Torpig botnet for analysis:

> "It is also interesting to observe that 38% of the credentials stolen by Torpig were obtained from the password manager of browsers, rather than by intercepting an actual login session. (Sect. 6.1)

Some password managers advertise an ability to remove passwords from browsers *without needing a master password*. Malware could grab the passwords just as easily.

Since emergencies eventually happen to all of us, our partners or at least *somebody* should have access to our current password database and the master password to access our web accounts.

For a cross-platform Linux and Windows password manager, I recommend "LastPass.com" on the Web, with the Firefox add-on. Various plug-in and bookmarklet implementations are available for a variety of platforms. Even off-line, the browser plug-in should be able to use (but not add to) the automatic local backup of the encrypted keys. A separate portable version allows use completely outside any browser.

Basic SSL

SSL is short for "Secure Sockets Layer," a security protocol introduced by Netscape in 1995. It has since been updated various times with the current version actually known as TLS or "Transport Layer Security," but still often called SSL. The intent is to establish a

secure connection between browser and web page, so that nobody along the line can snoop.

SSL capitalizes on Public Key Cryptography to generate the same transient encryption key on both ends of the conversation. The trick is that the key is not exposed to anyone monitoring the network. Once a transient key has been established, full data encryption starts that protects the communicated data from exposure.

The Internet Protocol (IP) is a store-and-forward network moving information from node to node in little chunks called "packets." Each and every node between the two ends can be thought of as being "in the middle" of the conversation. If we accidentally establish a secure channel to somebody in the middle, they can decrypt our data, read it, and re-encrypt it properly to send to the far end. With a "man-in-the-middle" attack neither end notices anything wrong, but the man-in-the-middle is reading the conversation.

For SSL to work, we must know exactly who is on the other end of the communication. Authenticating the web site occurs by receiving a certificate linking the web site address to an issuer whose certificate is included in the browser or the OS. Most SSL security problems involve an attacker trying to pass off a different certificate, or hide a mismatch or claim it does not matter.

SSL is particularly important for hiding passwords, especially for banking sites. But SSL only protects communications. SSL does not make an insecure system suddenly secure. If a botnet or key-logger is resident in the OS, it can easily capture passwords and whole login sequences, making SSL security irrelevant. This sort of sneaky background attack is exactly what is happening with "banking trojans" that install a bot whose whole purpose is to hide and watch for logins. The problem is not SSL, the problem is system security.

One class of attack tries to exploit the leading edge of SSL protection, as an SSL connection is about to be established. In an open subnet like a Wi-Fi hotspot, it is possible for one of the users to innocuously put itself "in the middle" with an "ARP Spoofing" attack. The problem is that many site login pages are not secured by SSL, but only have a button which is supposed to be secure, as in "sign in securely." In the original page the button probably includes an "https://" link to open an SSL connection. But someone in the middle can modify that page and remove the "s" from the "https://" so that clicking the button sends the password in the clear for the guy in the middle to collect. (For a related but different class of attack, see: Breaking Web Browsers' Trust.)

ARP Spoofing exposure is prevented by establishing a full SSL connection *before* entering any login data. Currently the user has to check this. It is important to only enter passwords into pages which already have an established SSL connection. For every login, you want all SSL, all the time. If that is not supported by a website, you need to complain.

You cannot trust the content on any page not delivered by SSL. You cannot trust your entered data to be secret on any page not delivered by SSL. Even SSL cannot be trusted on an infected PC. Since malware scanners cannot find every possible infection, we are necessarily reduced to re-installing the OS from scratch and keeping it pristine, just to depend on on SSL. Since all security requires an uninfected OS, we should be using a Linux live DVD for browsing, unless Microsoft comes out with an improved response.

ARP Spoofing is yet another reason to secure your Wi-Fi. Obviously, unsecured or open access would allow anyone nearby to download and re-distribute child pornography, copyrighted programs and music that could be traced to your broadband connection. In addition, ARP Spoofing would allow someone to steal your passwords, unless you and your family are just as careful at home as you need to be in a coffee shop. (Also see: YOUR Unsecured Wireless Internet is the Dangerous Weak Link.)

Graze Away from the Herd

Criminals design attacks which encounter users at random. The one product a random user is most likely to be using is Windows. So attacks which focus on Windows are most likely to encounter their target. Only attacks which encounter their target can succeed and profit the criminal.

According to Wikipedia (Usage share of desktop operating systems) the different OS's are used to browse the Web approximately in the proportions of: Windows 93pct., Mac 5pct., and Linux 1pct. The Market Share results (Operating System Market Share) are the same. If you were a malware designer who had to pay for your attacks, and those attacks only could find computers at random, for which target would you prepare?

The Mac benefits also from being a less-popular OS. Roger Grimes says, in Macs' low popularity keeps them safer from hacking and malware:

> "If anything, Macs have more known vulnerabilities -- by far -- than Windows and are often patched slower."

Similar comments occur in other articles, for example: Researchers: Macs are less secure than Windows PCs, and Risky Keyboards and Weaponized iPods: Is Apple Security an Oxymoron?

Currently, many attacks can be avoided simply by using a different operating system when on the Web. An attack which is designed to exploit OS faults, or simply use OS services, is unlikely to succeed on another OS. However, the CPU instruction set coding is largely the same on modern machines, so machine language should work on any OS, if it can be inserted somehow into running code. Interpreted systems like JavaScript and Flash also execute on any OS. The browser itself is an issue: A Browser's View of Your Computer.

Commentators and experts are recommending a Linux live CD for on-line use:

- Philip Dorrell (26 August, 2007): Ten Ways to Make Internet Banking Safer.
- Jens Porup (Sep 15, 2008): How to: Bank Securely While Traveling.
- Michael Horowitz in eSecurity Planet (August 11, 2009): Consider Linux for Secure Online Banking.
- Michael Horowitz again, this time in Computerworld Blogs (September 26, 2009): Crimeware gets worse - How to avoid being robbed by your PC: "Do online banking from Linux using Firefox."

- Joe Paniatowski, on suite101 (Oct 5, 2009): Protect Yourself Against Banking Crimeware.
- Australian Police Detective Inspector Bruce van der Graaf from the Computer Crime Investigation Unit in itnews (Oct 8, 2009): NSW Police: Don't use Windows for internet banking.
- Brian Krebs in The Washington Post (October 12, 2009): Avoid Windows Malware: Bank on a Live CD.
- Brian Krebs again (October 12, 2009): E-Banking on a Locked Down (Non-Microsoft) PC.
- Adrian Kingsley-Hughes in ZDnet (October 13th, 2009): Time to ditch Windows for online banking and shopping.
- And Brian Krebs again (October 20, 2009): E-Banking on a Locked Down PC, Part II.

Is Unix Malware-Immune?

Unix also benefits from being a less-popular OS, but is in no way malware-immune. Those who claim that Unix and Linux are immune to serious security issues have not been paying attention. In fact, the very first worm, the Morris worm of 1988, attacked BSD Unix, not Microsoft Windows 2.0 running on DOS. A brief scan of the past few of years of security news finds:

- February 2007: Linux botnets.
- October 2007: eBay: Phishers getting better organized, using Linux.
- February 2008: Major Linux security glitch lets hackers in at Claranet.
- May 2008: Alarming Open-Source Security Holes
- May 2008: After Debian's epic SSL blunder, a world of hurt for security pros.
- September 2008: Open source release takes Linux rootkits mainstream.
- November 2008: Buffer overflow bug bites Linux wireless component.
- March 2009: The First Linux Botnet.
- March 2009: Worm breeds botnet from home routers, modems (Linux routers).
- March 2009: Stealthy router-based botnet worm squirming.
- March 2009: 'Psyb0t' worm infects Linksys, Netgear home routers, modems.
- July 2009: Clever attack exploits fully-patched Linux kernel.
- August 2009: Bug exposes eight years of Linux kernel.
- September 2009: Linux webserver botnet pushes malware.

Comments:

- Apparently the July 2009 vulnerability was almost universal across Linux distributions, having been introduced by compiler optimization which was not apparent in the source code.
- Since Linux is open-source, we have actual evidence that simply being open-source does not solve the security problem.
- The Linux router botnet examples are particularly disturbing because very, very few users would notice a bot in their router.
- Technical attacks on the OS do happen, but attacks on browsers, applications, interpreted file codings, user inattention and so on also happen, and will happen even if the OS is perfect.

Only Idiots Get Infected?

Staying away from the bad side of the Web is no longer enough to stay safe. Malware is no longer confined to porn sites, but instead commonly comes from legitimate business sites: Researchers Hijack a Drive-By Botnet, and Hijacked Web sites attack visitors.

We cannot know what we have in our own computers. Bots try to hide so they can stay in service. Bots that are found and removed are the failures, and those are all we know. Since we do not find successful bots, we cannot believe we have them.

Malware infections may be much more common than we know. Antivirus scanners *cannot* certify a computer as uninfected (let alone the router outside that computer). The limitations of conventional tools make it far easier to *claim* to be uninfected than to actually *be* uninfected. The way to *be* uninfected is to frequently reboot the OS from hardware write-protected storage (like a DVD), and allow only controlled and authorized changes to that storage.

Travel with Your Own OS

Everybody wants to sit down at some random computer and sign on to handle personal business, but that is a very bad idea. Key-loggers, screen-loggers and password-stealers for Windows may be resident and waiting. Good security requires that you either have your own computer, or that you boot your own OS DVD on any computer which may have an infected boot drive. Do not use any public computer for email without booting your own OS DVD. That means:

- no hotel computers,
- no conference computers,
- no friendly bar computers,
- no customer computers, and
- no good friend computers,

***unless* you boot up your own OS DVD and establish a full SSL connection before you enter any password**. Since malware scanners cannot find every infection, even your own hard drive is potentially infected, so leave the drive at home unless you absolutely cannot do without it. For secure net access, take your own laptop, and even then boot up your own OS DVD.

Booting Linux from DVD automatically avoids most resident malware. Currently, the ideal choice seems to be Puppy Linux, which has the unique ability to save browser and add-on updates to a DVD in another session. The updated browser is then used on subsequent boots. Puppy Linux also supports Firefox, which provides cross-platform support for a wide array of add-on programs and important security features. Firefox provides automatic updates as needed for browser and add-ons.

Booting Linux from DVD *does not* defeat network snooping, so use SSL.

Switching from Windows to Linux is not a trivial change. Windows is a pricy product where we expect and get a certain level of polish and fit for our money. If we are disappointed by Windows, we have every right to complain, and perhaps even vent some outrage. In contrast, Linux generally is a free product and often somewhat rough. As a free product, when we are disappointed by Linux, our outrage may be sadly limited to mere questions and comments.

Taking your computer when you travel places your hard drive at risk, and you may not know what is on that drive. A surprisingly useful option is to simply remove the hard drive and use the laptop by booting into Puppy Linux from DVD. Put the hard drive in an external USB box, and leave that at home. It seems strange, but browsing *without* a hard drive is almost completely as usable as browsing *with* a hard drive. Without a hard drive, the laptop is a little lighter and the battery lasts longer. Also take a small router for use as a hardware firewall. Use wired connections wherever possible. Use only WPA2 AES-CCMP wireless security.

On-line accounts need long, random passwords for security. Since we cannot remember such passwords, we need a password manager, and we cannot trust the browser. The best cross-platform (Windows and Linux) alternative I know is an on-line account with LastPass.com. They have a Firefox add-on that automatically fills in the username and password fields for most login pages. There is a portable version for off-line use. Supposedly we can trust LastPass because our computer encrypts the passwords before they are sent to LastPass storage.

It is important to use SSL whenever possible to protect your data. When you do have an SSL connection, you are secure unless you have installed a bad certificate. If you go to a site which should allow SSL but does not, someone may be intercepting your traffic.

It is *crucial* to protect on-line account passwords with SSL before they are sent on line. Firefox has a relatively new add-on (SSLPasswdWarning) that warns when passwords are about to be sent in the clear.

Sometimes we need to save downloaded or created files, or just URL's, when browsing on a diskless machine. Modest size files are easily saved by attaching them to an email to self, using SSL-protected Gmail for secure access and filing. Of course, even huge files are easily stored to a modern USB flash drive, but then the security of a drive becomes an issue. Bookmarks can be saved to Google bookmarks.

Malware

Malware is deliberately malicious software installed without permission of the computer owner. Some (non-exclusive) categories include:

- **Virus** -- a malicious computer program that makes copies of itself on removable storage (like USB flash drives) to infect other systems.
- **Worm** -- a malicious computer program that uses network connectivity (such as email addresses) to infect other systems.
- **Trojan Horse** -- an apparently good file or program which actually contains a malicious computer program.
- **Rootkit** -- a malicious computer program that patches the OS file system so the malicious files are not visible, even to the OS itself.
- **Bot** -- a malicious robot program controlled by a hacker and hidden from the computer owner. Typically inserted on many different computers by Trojan horse files, thus forming a "zombie botnet" to carry out virtually unlimited hacker instructions.

Although malware is often compared to biological viruses and worms, malware is not alive. Malware does not infect humans or animals. Malware is just a sequence of values stored electronically. When malware is executed, the sequence of values tells the processor to make a sequence of data changes that we may not like.

A malware is a computer program, and all programs consist of data, usually in a disk file, and execution. If we can prevent a malware file from being present, it cannot do anything. If a malware file is present, but cannot get executed, it also cannot do anything. Malware has no independent "life," and it does not "lurk" or "wait." Malware does not jump out and take over. Malware must be executed by something before it takes action. Unfortunately, it is easier for an attacker to get a malicious file in place and executed than one might think.

Modern web practices commonly download code (JavaScript, Flash, etc.) and execute it in the browser. Attackers can get their code executed by hacking a web page and changing it to download and execute their code. Attackers can change a web page to claim the user needs to download and install a new player, which of course will be malware. Attackers can send their code as an attachment in an email.

Malware Goals

Modern malware is built to make a profit. Expert development groups build very sophisticated programs specifically for criminal use. Some possibilities include:

- finding financial accounts and passwords ("spyware"),
- stealing personal information to start new accounts,
- redirecting browsers to specific ad or fraud pages,
- directly selling fake software (e.g., in popups),
- extorting purchases ("scareware"), or
- generating "clicks" for advertisers ("click fraud").

Or the malware may join a botnet herd for:

- denial of service attacks (e.g., website extortion),
- hosting illegal or stolen files,
- distributing advertising "spam," or
- mass attacks on Internet structure in support of the usual online fraud.

Malware Examples

It should be interesting to look at current malware, how it is acquired and what it will do. Unfortunately, what we mainly have is statistics about malware scanner detection classes. The August 2009 Threats Update summary from the ESET Threat Center lists the most-often-found ESET scanner detections. Some are apparently a particular code sequence used in a wide variety of malware, which is efficient for detection, but not particularly insightful.

- **8.6pct. Win32/Conficker. Worm. Multiple variations.**
  Propagates in multiple ways, including unsecured shared folders and removable media like USB flash drives via Autorun, which should have been disabled. Restarts from the registry. May start an HTTP server. May download and run multiple files, which may be bots.
- **8.3pct. Win32/PSW.OnLineGames. Trojan. Keylogger and possible rootkit.**
  Steals game credentials and sends them to the attacker.
- **7.8pct. INF/Autorun. Many variations. Malware that uses the autorun.inf file.**
  Intended to run automatically when a removable media is inserted. Autorun should have been disabled.
- **3.6pct. Win32/Agent. Trojan. Many variations.** Downloads files. May install a backdoor. May interfere with security programs.
- **1.8pct. INF/Conficker. Worm. Conficker using autorun.inf.**
  Autorun should have been disabled.
- **1.6pct. Win32/Pacex.Gen. Malware that uses a particular obfuscation method. Many variations.**
- **1.4pct. Win32/TrojanDownloader.Swizzor. Trojan. Many variations.**
  Downloads malware.
- **0.9pct. Win32/Qhost. Trojan/Worm.**
  Can propagate by email. Modifies DNS settings, may not allow connection to security companies. May perform banking man-in-the-middle attack.
- **0.9pct. Win32/TrojanDownloader.Bredolab. Downloader. Variations.**
  Started by registry. Downloads malware.
- **0.8pct. WMA/TrojanDownloader.GetCodec. Trojan.**
  Converts audio files to WMA format, then points each file to a new "codec" to be downloaded, which of course is malware. Do not download an unknown player or codec, instead get a well-known player with internal codecs like KMPlayer, VLC Player or try Media Player Classic Home Cinema. If that does not work the problem may already be resident.

Malware Detection

Malware is thought to be detected by scanning files looking for a signature of each known malware, but that theory has serious problems:

- **Detection Issues:** For a signature to be in the database, malware first must have been detected by somebody in some other undescribed way.
- **Time Issues:** For malware to be detected in a downloading file, it must have been widely-distributed (so it could be found), then analyzed and the signature distributed *before* users encounter the file, even though it is widely-distributed.
- **Reliability Issues:** Often enough, malware is mistakenly detected in files which have no malware. Humans are then called upon to distinguish the "false positives" from real malware detections, a task for which we are not well suited, which is why we have signature scans in the first place.
- **Hiding Issues:** Targeted malware is not widely-distributed and so is unlikely to be in a signature database. Unless targeted malware exposes itself, it may remain active and unfound for years (see: Botnet PCs stay infected for years), until overwritten by an OS re-install.
- **Numbers Issues:** Huge numbers of new malwares mean that only some of those can be chosen for signature detection, while the others will remain undetected.
- **Signature Issues:** Polymorphic malware often avoids signature detection by changing itself, particularly when propagating.
- **Diligence Issues:** Heuristic detection is useless when malware authors get those same systems and then change the malware until it is not detected.
- **Irritation Issues:** Periodically scanning hundreds of thousands of unchanged files is time-consuming and enormously inefficient.
- **Recovery Issues:** The days of removing malware are over. Detecting malware means loading a saved backup image, or a full re-install from CD.

Malware must add new files or change old files (or the BIOS or boot sector) to live beyond a reboot. Malware can be exposed by detecting the changes that occur from infection, provided we know or record the state as it was before infection.

Malware can operate without infecting, and a DVD boot will not prevent that. A DVD boot does provide an uninfected system next time.

MALWARE STRATEGY AND RESPONSE

Malware is not the only threat to computer operations. Many problems are just software issues, the result of faulty or arrogant programming. One serious threat is the loss of stored information due to disk crash or motherboard failure or software problems or human error. We probably cannot prevent system failure, but we can minimize information loss by using appropriate backup procedures.

It is easy to prevent malware: Just do not use the Internet. Just do not use Wi-Fi or connect to a LAN (local area network). Just do not share flash drives with computers that are on a LAN. This is not a joke, but instead a viable option, and, indeed, an actual *requirement* for serious security. Virtually any modern computer can be hacked from the Internet if the attacker is willing to put enough effort into the problem. If you have secrets of your own to keep, get off the Net, get off the LAN, get all the way off all automatic interconnections. Reinstall your OS from scratch, and be very careful with flash drives.

Using the Internet *and* avoiding malware can be tricky. Only a limited amount of protection can be bought with firewalls, antivirus scanners and anti-malware programs. Targeted malware probably will not be found by scanning. Probably the most important protection is to not use Windows for browsing. If using Windows, at least keep the system updated and patched. Use an external router that has a hardware firewall (most do). Learn to not click on a malware link. Unfortunately, links do not describe themselves as malware. Attackers deliberately lie to get your click.

Attackers use modern advertising techniques to tempt users to click a link, supposedly to install a program or open an email attachment. Seemingly trustable web pages may have been hacked to download malware, while calling it anything else. Perhaps 9 out of 10 malware infestations are traceable to users specifically allowing attack code to run, believing it is something they want or need. Users must learn to hover the cursor over a link and look at the Status Bar (typically at the bottom of the browser) to see where the click would take the browser. Before clicking, the user should think whether the actual target is the claimed target.

It is possible to operate as a user, as opposed to an administrator. The historical reason for having users and administrators seems to have been a need to protect the OS from user damage in a multi-user environment. That environment concedes that an attacked user will be fully compromised, but hopes that reduced privileges will prevent the OS and other users from being compromised as well. In contrast, many modern PC's are single-user systems where *there are no other users to protect.*

The advantage of operating as a user is to gain some amount of protection for OS files, not user files. Any important files a user can access, an attacker who exploits that user can also access, for exposure or damage. Protecting OS files is the extent of what limited user privileges can prevent on a single user system.

OS files reside on a drive which is completely exposed to the attacker when the OS is defeated or bypassed. Changing OS boot files is how malware survives a reboot.

OS files can be protected by a "live DVD" boot, where the critical OS files reside on DVD, and cannot easily be changed by attackers. For malware to survive a DVD reboot, it would have to write to the boot DVD, which is both difficult and obvious, easily avoided by removing the DVD, and the DVD is easily replaced in any case.

Live DVD security is fundamentally superior to the classic hard drive boot, and makes limited user privileges unnecessary. While Puppy Linux is designed for a live DVD boot, Windows is not. For Windows, various other issues would come into play, such as extended boot time and registry update and protection.

Anti-malware systems can detect *old* and *widely-distributed* malware when scanning downloads and attachments. New "zero day" viruses probably will not be detected, nor will viruses targeted at particular individuals, groups or offices. And once malware runs, not much can be done. In past years, later scans might find and remove the malware. Nowadays, malware files can still be removed, but modern malware has broadband access and often joins remote-controlled bot-herds. There is no way to reliably know what the malware did under remote control, or how to reverse it. One possibility is that more extensive and better hidden malware has been installed. On a system which sees any valuable data or passwords at all, an uncorrected malware infestation can have serious consequences.

The correct response to a malware infestation is a lengthy process of saving current data and then re-installing the operating system (OS) or an uninfected image. Then the system must be brought up to date, and configured for individual use. Then we have to check any other computers and USB drives that may have been attached to an infected system, and all the systems they may have attached to. All equipment with programmable on-board flash ROM (e.g., motherboards, video cards, possibly hard drives, CD drives, routers, printers, etc.) should be re-flashed, a process which is not only tedious, but also carries substantial risk of failure. The settings on programmable equipment (e.g., routers and network printers) should be set up again. All this can take days of effort, spread over weeks of computing interruption, and is so much work that nobody ever really does it.

The best approach is to not get malware in the first place. Malware is easily avoided by not using a computer, or by not connecting to the Internet. Unfortunately, the Internet usually seems too advantageous to avoid, which is why we have performance-sucking scanners and awkward hardened systems that irritate users. Nobody wants to deal with layer after layer of protection, but we *really* do not want to do what should be done after getting infected.

MALWARE DEFENSE

There seem to be some fundamental approaches to malware defense:

1. **Keep Malware Out.** The conventional approach is to prevent malware from entering the system, or to prevent malware

execution. This includes the firewall, the perfect OS coding, the perfect application coding, and the sandbox. If this approach actually worked, by now there would be no malware problem. The approach does not work because of the number of allowed values and actions is too large to test, and too complex to get right.

2. **Remove Malware.** The usual approach is to use a virus or malware scanner to detect malware and remove it. If this approach actually worked, by now there would be no malware problem. The approach does not work because scanners cannot detect all malware and instead often detect false positives. Even if the malware is detected, it is not possible to know or remove whatever the malware has done while in a bot herd.

   A related alternative would be for Microsoft to supply a live boot DVD that checked every necessary Windows file and replaced any that needed replacement. A full and aggressive implementation of this approach holds perhaps the best near-term promise of a malware fix for Windows. Another live DVD might provide a secure environment for browsing, but would have to not be Windows underneath.

3. **Do Not Use Windows on the Web.** Since the vast majority of attacks depend upon Windows, the obvious approach is to use an OS which is not Windows, and then not bother too much about malware protection. That has been very effective for Apple and Linux users so far. The advantage of an alternate OS is not that it is better, but that it provides a worse return on attack investments. The malware advantage of an alternate OS does depend upon criminals doing what we expect, but that has been a good bet so far.

4. **Reboot Fresh.** Malware must change OS files to re-install itself on every Restart or "boot." Since files on CD cannot be changed, the simple use of a CD or DVD boot disk automatically boots malware-free every time. Booting from CD can provide security even on infected machines.

   While a simple Windows boot CD would help by preventing changes to boot files, Windows is large, thus slow to boot on an optical drive, and we would just end up with the Windows target again. Windows-oriented malware could still break through individual sessions, just not be able to save itself for next time. The Windows updates so important for hard drive systems actually mitigate against an unalterable boot DVD. The DVD boot disk advantage seems best when applied to a small (fast loading) and stable (no updates) OS.

   A useful expansion from the CD boot is to actually have no hard drive at all (using, for example, Puppy Linux). When there is no hard drive, there is no hard drive for malware to infect. Computing without a hard drive really is very pleasant and practical. Beyond saving the occasional file to a USB flash drive, and the relatively slow DVD boot, the lack of a hard drive may not be noticed.

   An extremely useful option is a writable multi-session DVD which is practically ideal for program updates. Browser and add-on updates are simply downloaded as usual and then automatically appear in subsequent boots, just like on a hard drive. At the end of a work session, the user can tell Puppy Linux to write changed files to a new DVD-R or DVD+RW multi-session, or not. Previous file versions remain on the DVD unchanged, and the Puppy Linux OS is responsible for loading the most recent copy into the memory file system for use. In Puppy Linux, after boot time, the DVD can be removed, or the DVD mounted to access archived older versions of files.

   Multisession writes do carry some risk of malware infection, but the user need not save a session unless there are updates, and even then only if nothing seemed strange. Saved sessions can be viewed and inspected. Saved sessions can be voided if infected or even if some program configuration has gone wrong.

   Booting from a flash drive is another possibility, potentially faster than a DVD, but unfortunately carrying a very real risk of malware infection. For an effect similar to a DVD load, it may be necessary to have a flash drive with a hardware write-enable/disable switch (such as an SD-card in a small USB card reader). Ideally, we could remove the flash drive to enable writes, then replace it for an end-of-session update, something not currently possible in Puppy Linux.

   Yet another possibility is to boot inside a virtual machine (VM). Unfortunately, if the base machine is infected, we no longer can trust the VM, and even a CD boot cannot be trusted. In contrast, when we boot the physical machine from a live DVD, the only relevant infection is whether the DVD is infected.

5. **Protect Boot Data.** Our hard drives are vulnerable to having stored OS files changed by malware. Regulating changes to the official boot files on a hard drive would prevent malware from being there and thus prevent malware from being re-installed during boot. Owners would have to specifically authorize changes, even for cryptographically authenticated updates. Unfortunately, current hard drives do not have this facility, and even if they did, Windows probably would require substantial changes to use it. The hard drive storage authorization approach does have the very substantial advantage of being a technological fix for a whole class of Windows malware.

   A similar issue exists with respect to the motherboard BIOS, the video card BIOS, and every other flashable BIOS in the system. These all need hardware protection against unauthorized change and the ability to easily authenticate their contents. The OS cannot provide this protection.

   Hardware storage authorization also might be useful for USB flash drives, and in general, all drives.

BASIC WEB PROTECTION

We can start with some basic Web protection:

- The single most important protection against malware, sadly, is to use anything other than Windows when browsing. Not using Windows removes the main malware target.
- The next most important protection is to boot from a live DVD. Booting from DVD starts a new session without malware. It does not prevent new attacks, which is why we need something other than Windows. In contrast, booting from a hard drive means "once infected, always infected," which is just flat-out dumb.
- Firewalls are needed both in OS software and in an external router. But firewalls can only stop unrequested attacks coming from the outside; they cannot protect against users inviting bad things in. Software firewalls are easily deceived about which program is sending what outgoing data, so a fancy firewall usually is not much advantage.
- Use a password manager (e.g., LastPass) and set up different long, random passwords for each site and router.
- Use OpenDNS instead of the default domain name system (DNS) server from your internet service provider (ISP). OpenDNS will not allow contact with known problem sites and has up-to-date patching that may not seem important to the local ISP.
- Many browser attacks can be defeated by using an alternate browser with security add-ons. Use Firefox with Adblock Plus, BetterPrivacy, ForceTLS, NoScript, SSLPasswd Warning, and WOT add-ons. (Opera has no comparable add-ons, and it is still early to discuss the security of the Chrome browser.)
- Get web email (e.g., Gmail) and they will scan your attachments and let you browse them without a raw download to your machine. Gmail also supports SSL for all browser mail operations. The SSL connection supports fairly secure storage for attachments sent only to self and fairly secure messaging to others on Gmail.
- For Windows, use the Microsoft Security Advantage scanner.
- For Windows, download the free "Spybot - Search & Destroy", and use immunize.

Firefox add-ons like NoScript expose the difference between casual browsing and cautious use: If we consider security as a wall around our protected town, good security involves closing every possible access, opening only when we need to, and only for what we need. We thus enable access to specific pages, which is a "white-list" or "opt-in" policy, as opposed to the usual policy of leaving everything open unless we actually catch some bad things sneaking in. Sadly, NoScript will actively prevent many exciting user interactive things which are potentially dangerous and so cause some anguish. However, if desired, we can easily enable everything on a particular site. Finer control is available, and very useful, once we realize that the annoyance is in our own best interests.

Just using safer programs is not enough:

- Users need to be cautious and aware of deceptive links. Users should mouse over a link and examine the address on the status line before clicking.
- Users should never click on a link that claims it will "fix" something on the computer.
- Users should try to use SSL (the **https://** protocol) at all times, on all their membership sites, but especially when they enter a password.
- Users must never approve a new SSL certificate for their use, especially when using public Wi-Fi in a coffeehouse.
- Users should make a long random password for each site, and use a password manager to store those passwords.

Top Recommendations

The top recommendations are:

1. Keep 3 independent copies of important data files.
2. For Windows, image the installed OS weekly or monthly.
3. Instead of Windows, use Puppy Linux from DVD+RW.
4. Use an external router with firewall.
5. Use a software firewall.
6. Use OpenDNS (208.067.222.222 and 208.067.220.220).
7. Currently, use Firefox with Adblock Plus, BetterPrivacy, ForceTLS, NoScript, SSLPasswd Warning and WOT add-ons.
8. For Windows, use the Microsoft Security Essentials malware scanner.
9. For Windows, use "Spybot - Search & Destroy" with the "immunize" function.
10. Use a cross-platform password manager (e.g., LastPass.com).
11. Make a different long random password for every device and every site and every account.
12. When browsing, use SSL (**https://**) whenever possible.
13. Never enter a password unless that page is already under SSL (**https://**).
14. Never approve a new SSL certificate.
15. Use on-line email through SSL (e.g., Gmail).
16. Never download unexpected email attachments, even when they claim to be from someone you know.
17. Always mouse over links and examine the target address before you click.

Consider Browsing from Puppy Linux

The single most important defense against malware is to not be browsing from Windows.

Users should take a deep breath and at least consider using Puppy Linux on DVD (or some other Linux "live" DVD alternative) as their OS for the Web. When Puppy is running Firefox, the browsing experience can be very similar to Windows, albeit with some occasional glitches. When Puppy Linux is booted from DVD, the resulting security should be better than a whole raft of Windows add-ons. Puppy browsing is just a boot away on almost any PC. Even better would be Puppy Linux on a machine with the hard drive removed, or simply disconnected.

BASIC SECURITY

How can we keep our doors closed, when we have to open them to deal with the outside world?

The Heart of Security

Sometimes security is described as "lock every gate," but the locking part may be the lesser problem. Gates imply a wall, and if a security wall does not make weakness obvious, there could be gates anywhere. The real problem lies in finding all the possible gates. Closing each newly-found gate is some help, but overall may not make much difference if other open gates remain undetected.

All good security works by preventing almost everything, then paying close attention to the few things allowed. Avoiding all but a few selected alternatives avoids the need to analyze each and every possibility as it occurs to see if it is a threat. Avoiding analysis is important for computing, because exposing an apparently innocuous file as a threat may require a deep understanding of content which computers do not yet have.

On the Internet almost every alternative is allowed, so the potential for security problems is almost unlimited. Since computers do not understand content, full analysis cannot occur in real time, leaving the net inherently unprotected against new deceptive attacks. The current approach of trying to patch each bad alternative after it has been found cannot provide security and will never be complete.

The Walled-City Model

Many of the issues in general security can be modeled by analogy to a city surrounded by a defensive wall. The wall cannot be perfect, and large gates are necessary. On the Internet, each piece of "wall" might be the "ports" controlled by a particular application or service.

A real city has a continuous flow of people and goods in and out, and that flow is necessary for life. If we want a live city, we had better not permanently close all the gates, even if that would be secure. Similarly, using the Internet implies a flow of data in and out. We can cut off outside connections, but what we want is to have those connections and be safe anyway. With limited information, that may not be possible.

At the walled city, we can minimize the number of gates. Then we can put guards on the gates, and prevent the random wanderer from coming in. We can oppose an organized attack. But many of those entering will do so on invitation from someone inside, and there really is not much guards can do about that. Similarly, our Internet firewalls mostly prevent random external access, but pass anything requested from inside. All an attacker needs is for somebody inside to request an entry. The only way we could hope to prevent that is to know a lot more about the external petitioners than web lets us know. In a sense, we need a "photo ID" for every page that responds to us.

Once inside the city, anonymous plotters can infect previously well-working organizations and systems. In a real city, it is tough to find illegal infiltrators, so we generally relax until they do something very illegal. In a computer, it should be possible to know and check each and every allowed file, and we can or do have their hash value "photo ID". Ideally, all the critical files would be in the same general place, *and no other files would be allowed*. Unfortunately, that is not our current reality.

Computer Security

Modern personal computer (PC) operating systems (OS's) are large, detailed and immensely complex. Since complexity makes strong security impossible, our OS's are by design vulnerable to attack and control. These large complex systems require frequent patching, and so necessarily reside on easily writable hard drive storage, which is easily infected by malware. Ideally, the system would allow only *authenticated* changes, but when an OS has been subverted it cannot be trusted to perform authentication. An alternative would be some sort of live-CD patcher that makes and validates updates, but no such system is known. Ideally the OS and hardware would conspire to absolutely prevent unvalidated changes to OS code on the hard drive, but currently that does not happen.

For whatever reason, Windows is the OS of choice of most computer users, making it also the target of most malware attacks. General OS and Windows-specific complexity allows malware to hide among tens of thousands of legitimate files, registry entries and distributed BIOS code. Malware necessarily must add or change system code simply to come up again after reboot. System code changes should be very detectable from a booted live CD, yet no such system is in common use.

Although Windows OS updates and security patches are a valiant effort, they cannot even fix all implementation errors, let alone security faults in fundamental design or in application programs. Running an unprotected Windows system is demonstrably dangerous for many users. Recovering from infection often requires a full-image restore or a lengthy full OS re-install because malware has changed the system on the disk. A hardened Windows system is difficult to develop, can be unusably slow, and for all that still cannot guarantee security.

PREVENTING POSSIBLE PROBLEMS

Our best strategy is to prevent problems, or prepare to handle them. We seek to protect passwords, on-line accounts, financial accounts, private data, and the usability of our computers. We need to identify as many distinct problems as possible, and then innovate responses to handle those problems:

Trojan Horse programs are avoided by users knowing how to not run programs coming from outside their system.

- Do not open unexpected email attachments.
- Do not click on links to financial sites; instead, go to the site normally.
- Do not try to update a player from an application web page.

- Downloads are safely obtained by explicitly going to the website of the manufacturer.

Attacks on operating system flaws are avoided by installing updates as soon as they are available.

Worm attacks are prevented by using an external router with NAT (Network Address Translation) and firewall. The only incoming packets allowed are specific responses to previous outgoing packets.

Attacks that reprogram the BIOS flash are prevented if a write-enable hardware jumper exists and is physically removed. Absent some form of BIOS write protection, it may be necessary to re-flash the BIOS whenever it might possibly have been virus-flashed.

Flash-drive viruses are opposed by disabling autorun, and by using an SD memory card USB reader (instead of a USB flash drive) and flipping the SD write-protect switch.

Incoming computer viruses are rejected by using an antivirus program to scan every drive write operation. Unfortunately, antivirus scanners cannot detect what has not yet been found and analyzed, making perfect detection impossible.

Installed computer viruses might be detected by a general scan of all files. Detection obviously depends upon whether the virus has come to the attention of the antivirus maker, but any detection can be a "false positive" error. Then, whenever a "virus" is detected, somebody has to decide whether to believe the antivirus scanner, or discount the report. "Virus" detections in newly-downloaded programs may be both more believable and more significant than "virus" detections in old graphics files. But if we believe the scanner, we generally need to re-install the system.

Rootkit malware that has somehow gotten through the defenses may not even be visible from inside the OS. Directory listings in the altered OS may not show malware files. The data in a malware-modified file may even appear unmodified to the altered OS. Malware scanners running under an altered OS may not be able to find some rootkit malware. To expose reality, it may be necessary to access the file storage from an uninfected OS, such as a live DVD, or as an external drive in another system.

Dangeous websites are hidden by using OpenDNS (and possibly a hosts file) and are indicated by WOT (Web Of Trust).

Attacks on web accounts are defeated by using long, random password values stored in a password-manager (absolutely not in the browser).

Attacks on web account credentials are defeated by using random password values for challenge question answers. The answers are saved in the password-management program.

Browser attacks are addressed by using the strongest acceptable browser, security add-ons, and the latest updates.

Software problems, or at least their effect on the computing platorm, are minimized by automatic daily backups of the registry (ERUNT) and periodic registry cleaning (CCleaner).

File loss is minimized by making backup copies of important files. As a rule, any important new file should be copied to at least 2 other independent storage places. Storage is "independent" when any single failure cannot damage more than 1 copy. **Multiple copies on the same physical drive will not help when that drive fails.**

Hard drive failure (a) is completely fixed by getting a new drive and loading it with a saved drive image. We have to make drive images periodically, and save them on an external hard drive used for that purpose. The more often we make images, the less we have to lose from a crash.

Hard drive failure (b) sometimes can be recovered in place by SpinRite, which is a commercial, bootable, live CD. SpinRite is not the usual data recovery program, but instead repeatedly reads bad sectors until a good read occurs, then writes that out, possibly to a replacement sector. Although the SpinRite program must be executed on a PC compatible system, it can process Linux, Mac and Tivo drives. For SpinRite analysis, drives do need to be directly connected to a PC motherboard, and not just in an external drive box. So:

- **DO NOT JUST GIVE UP ON A FAILED DRIVE.**
- **DO NOT DISCARD OR RETURN THE OLD DRIVE IF YOU STILL WANT THE DATA ON IT.**

BACKUPS

A backup is a copy saved where it cannot be damaged with the original. Computer backups can be file copies or drive images.

Backups save data as it was at a particular time, with the intent being to recover that data if the original is damaged. But as time goes on, originals are edited, updated or changed to new and improved versions and the old backups become correspondingly less useful. When failure occurs, we want to recover the system we had just before the failure, not as it was a week or a month ago.

Backups have a surprisingly limited useful life, so only a couple of backups need be retained. Backups should be stored independently, since keeping backups on the same physical drive as the originals risks a disk crash that takes out everything.

File backups just save a particular file or set of files, which saves the file data as it was at a particular time. The file system also saves a little housekeeping information, such as the file name itself, the size, the creation date, as well as the location of that file within the drive. Sometimes there is an issue of file backup software failing on very long paths or strange characters in filenames, resulting in unrecoverable errors or emergency user interaction in the middle of lengthy backup operations.

Image backups generally copy each sector of a logical drive, thus inherently including both data and file system structure. Ideally, only *used* sectors are copied, and the result may be compressed. Image backup can be faster than file backup, because an image does not deal with individual file names and dates or subdirectory structure or multitudes of short files. It takes time for the operating system to "open" each file, so dealing with many short files can involve substantial OS overhead.

Some imaging programs allow an image to be mounted as a drive, making individual files accessible for copying, even though they had not previously been identified as important. Because an image backup generally includes boot sectors which are hidden from a file system, an image often can be recovered to a different drive and then simply booted into operation. Image backup recoveries thus avoid lengthy OS re-installs otherwise needed to support files. Image backups also retain the original OS configuration.

Multiple copies are important so that if one is damaged, the others remain. Moreover, to be considered a separate copy, each should be in an independent place, so that damaging one copy does not also damage another. Thus we save files in various general locations:

- **Locally** (on the same drive)
- **Externally** (on a USB flash drive or a USB hard drive that is powered only when backups are made)
- **Off-site** (as an email attachment, or on your web site, or with a commercial backup service, or in the car, or with a friend, or all of the above). Encryption may be desirable for off-site storage, which also implies keys and key management issues that cannot be fully automated.

Cloud backup services are available with a modest amount of free storage. Various distinguishing issues include the amount of storage, the number of computers on a single account, security, etc., along with details of how files are sent and later received. Some cloud services are:

- Dropbox
- SpiderOak
- SugarSync
- DropIO
- IDrive
- JungleDisk

Frequent backups are crucial to minimizing loss. Backups going back years are rarely helpful. When failure occurs, we are interested in recovering what we had yesterday, not last month or last year. On the other hand, we can have the computer make a copy every 5 minutes, but if we put those copies on the same drive as the original and the drive fails, we then lose the backups along with the original. It is important to have multiple *independent* copies, where damage to one will not also damage the others.

Automated backups are risky because automatic things tend to fail unexpectedly in unexpected ways. For example, automatically rotating through a limited number of backup copies fails us as soon as we need to go back farther than the earliest remaining copy. Similarly, putting multiple copies on the same drive fails us when that drive fails. In contrast, we can easily survive drive failure by making backups and saving them in multiple independent places. Manually copying a file to a flash drive and/or sending it to yourself as an email attachment surely could not take more than 5 minutes, yet could avoid massive recovery efforts and actual data loss.

Image recovery is impossible (in general) when drives are actively in use. Typically, image recovery requires a "live CD" that boots from CD and runs a replacement operating system in memory. That OS then can analyze and recover even the original boot drive.

Image recovery is dangerous because it will overwrite the entire file system for a particular drive. Frequently some files have changed with that being unnoticed or forgotten until they are no longer available. **It is important to save an image of the damaged system (when possible) before recovering an earlier image.** It may be possible to examine that image later, and recover forgotten files that way.

PUPPY LINUX

Puppy Linux is a free alternate operating system that most ordinary Windows users can use quickly. Puppy Linux supports the Firefox browser and most Firefox add-ons, thus delivering a browsing experience similar to Windows. There are various advantages:

- **Puppy Linux is not Windows.** Malware that infests Windows generally assumes the presence of Windows services, data structures and program code. When Windows is not present, the malware generally cannot operate successfully.
- **Puppy Linux can boot from DVD.** If an infection is acquired during operation, even in Linux, a simple DVD re-boot solves the problem.
- **Puppy Linux updates the boot DVD.** A modern browser is updated frequently to patch security flaws. With a DVD OS, updates would normally mean burning a new DVD and doing a new configuration. In contrast, Puppy Linux allows writing file changes and browser and add-on updates as another session on a multisession boot DVD. The next time we boot we have the updated browser just like we would with a hard drive OS. If a work session seemed strange, we do not have to save it. If Puppy starts acting funny, we can void the previous session and restart to a session before the problem.
- **Puppy Linux does not need a hard drive.** When there is no hard drive, there is no hard drive to infect. Browsing without a hard drive is surprisingly normal. We have many options to save information beyond relying on a vulnerable hard drive:
    - Save files with the session to the DVD.
    - Save files to a USB flash drive.

    - Save files as Gmail attachments in email.
    - Copy and paste text into an email.

- **Bookmarks are a self-made burden.** For synchronizing bookmarks between Windows and Puppy I had been using Xmarks with great success. Eventually, though, my rapidly-expanding bookmark list made that intrusive by needing a ponderous download for every sync operation. Now, by exporting bookmarks as HTML then sending that file to myself as an email attachment, I can have all the old and obscure bookmarks available by viewing email. I save important new bookmarks to Google bookmarks "in the cloud," to avoid extending the DVD on every session. I do keep a few everyday bookmarks (tabs) locally.

Puppy Linux is not the whole security solution. Using a different OS can protect the computing environment from malware, including viruses, key-loggers and bot-nets. Browser, password, and connection security issues remain, although Firefox add-ons can be a big help. In addition to using Puppy Linux, users should:

- Use an external router firewall and the Linux firewall.
- Use on-line email, and view but do not download unexpected attachments (Gmail).
- Use the Firefox browser and keep it up to date.
- Use security add-ons, especially to support secure (SSL/TLS) connections (BetterPrivacy, Force-TLS, Permit Cookies, SSLPasswd Warning).
- Have and use different long, random passwords for each site account (LastPass).
- Force SSL (https://) page connections whenever entering anything, especially on social sites.
- Only enter passwords on clearly SSL-protected pages, particularly when using an unsecured wireless connection or free Wi-Fi.
- Never approve new SSL certificates.

Puppy Linux is free, voluntary software. New versions have been coming out every few months, none of which are oriented specifically toward secure use. Apparently Puppy has had multi-session DVD operation for years, and may be the only live-DVD Linux supporting browser updates.

The Linux Development Environment

The reality of free Linux can be a shock to Windows users. Microsoft has a few basic flavors of Windows, all of professional quality, which stay basically the same for years. Puppy Linux especially has any number of different versions, often going in different directions with a few part-time developers, which may last a few months to the next version, or just die out. The way to assess the new work is to download a likely .iso, burn it to DVD and see what it does.

Business users often point to the lesser quality in the free Linux distributions, but if Windows was really doing a quality job, we would not be here. Windows has simply failed to provide the on-line protection we need. Reasonable people can disagree about which OS is better and yet still prefer Linux to Windows for particular situations. In most cases, we can take an ordinary PC, boot Puppy Linux in a couple of minutes, do our online work in much greater safety, then reboot Windows. Those who do most of their work in the browser may not have as much need to reboot Windows as they first expect. However, when it comes to devices that are in any way unusual or specialized or new, Windows always has a driver, and Puppy generally does not.

In free systems, it is common for things to not work as well as in a professional product. In the case of Puppy Linux, usually there are work-arounds, since many other people have to do the same things as you anyway. Or there may be a new fixed version in a few weeks or a month. On the whole, Puppy Linux is very usable.

Puppy Linux Resources

A surprising amount of information is available, although some of it is dated and some is hard to find. Here are some starting points:

- A remarkable visual .PDF introduction: Introduction to Puppy Linux: Installation on a USB Flash Disk.
- Various video tutorials: TutorialYouTube
- A basic starting point: Puppy Linux
- Another starting point: Basic info - getting started
- The Wiki home page: HomePage
- A Wikibooks presentation: Puppy Linux
- A manual: English Manual for Puppy 4.0
- The Barry Kauler site: Puppy Linux
- Puppy itself includes help, which brings up a little HTML viewer linking to some of these pages and a whole list of internal how-to's and applications documentation.
- The Puppy on a CD (or DVD) page: Puppy on a CD
- The multisesson DVD page: multisession live-DVD (and CD)

Write Puppy to DVD+RW as Multi-Session

For security use, Puppy should be booted from DVD and **not** installed to a USB flash drive or hard drive. Any boot medium that is immediately writable can be trivially infected. To freeze out malware, we need to not provide a easily infectable environment. The following instructions are for Puppy 4.3.1 and similar.

Although the documenation recommends DVD-R, I think the results depend upon particular equipment and especially discs, which seem to vary a lot. I have had better luck with DVD-RW, and I can erase those and start over.

1. In Windows and Firefox, from:
   - ftp://ftp.oss.cc.gatech.edu/pub/linux/distributions/puppylinux/, or
   - http://puppylinux.com/, or
   - http://puppylinux.org/wikka/HomePage, or
   - ftp://ibiblio.org/pub/linux/distributions/puppylinux/ (very slow).
2. Look around to find pup-431.iso (105MB) or newer and download.
3. Burn the .iso to a DVD+RW, using special software if necessary (try CDBurnerXP). Use session-at-once and unselect "finalize" or select "Mode 2 XA multisession". (We want a "closed" session, but not a "closed" disc. If Puppy cannot write to the disc at the end of the first Puppy session, use Puppy Menu/Multimedia/Burniso2cd to burn another .iso copy using the Puppy burner.)
4. An .iso file is just a drive-image of a CD or DVD. An .iso should not be burned as a normal file, because it already has a file structure, with files in place.

Download Extra Programs

We want Firefox, and it need not be recent because it is so easy to update. We also want the latest possible Flash Player, which is harder to update. Some Puppy versions (e.g., puppies-431.iso) include Firefox and an updated Flash Player, and so do not need .pet installs. The main Puppy version (e.g., pup-431.iso) usually does not have Firefox, and Flash Player quickly becomes outdated. Firefox and Flash can be installed (or re-installed) by finding and downloading .PET files. To install a .PET, copy it into Puppy memory (perhaps /tmp), click to install, then delete the file.

1. Go to a .PET download site like:
   - http://www.puppylinux.ca/tpp/bugs/ (user: puppy, password: linux, twice) or
   - http://puppylover.netsons.org/dokupuppy/ or
   - http://dotpups.de/puppy4/dotpups/ or
   - http://petstore.puppyspace.org/ or
   - http://www.wisdom-seekers.com/puppy.html.
2. Depending on what your Puppy version contains, possibly download .PET files to USB flash (then copy them to the Puppy drive before clicking to install):
   1. Firefox -- firefox-3.5.pet or later (easily updated on-line)
   2. Adobe Flash Player -- adobe_flash_player-10.0.32.18.pet or later

There is a Linux program called Wine which emulates Windows and allows some Windows programs to run in Linux. You may be tempted to install Wine, but do not do it! Wine has gotten good enough to support a range of Windows malware, which is precisely what we are trying to avoid. If we need to run Windows programs, we probably should stop browsing and re-boot into Windows.

Example Puppy Install

When used for improved security, Puppy Linux should ***not*** be installed to a hard drive but should instead boot from DVD on every session. As programs and configurations are added to Puppy, the changes should be saved to the Puppy DVD, provided nothing strange has happened. If any attack files actually do make it into the RAM file system and are backed up on the DVD, that session can be marked bad later. This example install is limited by my ancient monitor. **This is a working example for my particular equipment--do not follow it blindly!**

A. **Startup**
   1. Boot the Puppy DVD
   2. confirm US keyboard layout
   3. confirm English,USA language and country
   4. select US/Central timezone
   5. tab and select XVESA display (preferred on my equipment)
   6. for video changes, follow Menu / Setup to Xvesa Video Wizard and click
   7. select 1024x768x24 or 1280x800x16 and CHANGE to try it
   8. use control-alt-backspace to recover, if necessary
   9. confirm OKAY for video mode

B. **Access .PET Files**
   1. insert USB drive with .PET files (e.g., Firefox and Flash)
   2. USB flash icon appears named sda1
   3. single-click on sda1 icon
   4. green dot indicates flash drive mounted
   5. (right-click and select "Unmount sda1" before removing)
   6. file manager window opens with flash root
   7. single-click to navigate to downloaded .PET files

C. **Install PET Packages**
   1. (run .pet packages from Puppy RAM instead of USB flash)
   2. single-click on desktop "file" icon
   3. file manager window opens into "~" and subdirectories
   4. single-click the green up-arrow
   5. file manager window changes to deeper directory level
   6. control-click to highlight each desired .pet file in flash
   7. click to drag all highlighted into "tmp"

8. select "copy" so the source version is not erased
9. single-click on "tmp" to open subdirectory
10. for each .pet file, single-click on file, click "OK" to install
11. right-click and hold on USB drive icon, select "Unmount sda1" and release
12. remove USB drive
13. close file manager windows to move on

D. **Set Up Firewall**
1. follow Menu / Network to Linux-Firewall firewall and click
2. select OK, press Enter
3. press Enter to move on

E. **Configure Internet**
1. on the desktop, click connect
2. click on internet by network or wireless LAN
3. click on eth0
4. click on Auto DHCP, connection succeeds
5. possibly save configuration, which gives an easier startup,
   but may cause problems when DVD boots on a different computer
6. click Done to move on

F. **Update Firefox**
1. on desktop, click browse to start Firefox
2. in Firefox, follow Help to select "Check for Updates"
3. click "Update Firefox"
4. click "Restart Firefox"
5. close Firefox

G. **Save Changes to DVD+RW, then Reboot**
1. follow Menu / Shutdown to Reboot computer and click
2. Tab to "SAVE TO CD" and select (press Enter)
3. Tab to "SAVE" and select
4. on laptop, close DVD tray
5. select "OK" (press Enter)
6. Puppy comes back up
7. if network configuration not saved, on desktop click connect and
   set up internet connection as before
8. on desktop click Browse to start Firefox
9. in Firefox, follow Help to select "About Mozilla Firefox"
10. confirm updated version

H. **Install Firefox Extensions**
1. in Firefox, follow Tools to "Add-ons" and click
2. select "Get Add-ons" and click "Browse All Add-ons"
3. Mozilla "Add-ons for Firefox" page opens in browser
4. search for and select each desired add-on and download into Firefox
   (if not updated on Mozilla, go to author's site for latest version)
5. at least get important / security add-ons, shown in **bold**
   - **Adblock Plus** -- hide ads to improve speed
   - **BetterPrivacy** -- manage Flash cookies and DOM storage
   - Down Them All -- fast download manager
   - **Facebook Secure** -- use SSL for Facebook
   - FireFTP -- FTP client
   - **Force-TLS** -- remembers to use SSL on some sites
   - JSView -- expose external stylesheets and JavaScripts
   - **LastPass** -- encrypted passwords in the cloud
   - **Long URL Please** -- exposes target of short URL's
   - MD5 Reborned Hasher -- check hash in normal downloads
   - **NoScript** -- scripting control and other issues
   - NoSquint -- page and text sizing per site
   - PDF Download -- better PDF control
   - PageDiff -- show differences between HTML pages
   - **Permit Cookies** -- allow specific sites to save cookies
   - Save Complete -- File / Save Page As... improved
   - Shooter -- capture screen or entire page as graphic
   - SearchMenu -- fast dictionary, thesaurus, everything
   - **SSLPasswdWarning** -- warns when sending password w/o SSL
   - **Tab Mix Plus** -- tab recovery after crash (also use Bookmark All Tabs)
   - Uppity -- URL up-one-level
   - **WOT (Web Of Trust)** -- danger colors on search result links
6. each can be uninstalled or disabled later from Tools / Add-ons...

7. when done, select restart Firefox
8. when Firefox comes up, use Tab Mix Plus Session Manager, accept everything

I. **Configuring Firefox for Windows and Linux**
   1. if you can configure Firefox on your own, do so
   2. In Windows, follow Tools to Options and select.
   3. In Linux, follow Edit to Preferences and select.
   4. Set up a Home Page URL.
   5. Follow View / Toolbars to deselect "Bookmarks Toolbar"
   6. In the Main tab, Downloads,
      - for Windows select always ask where to save file
      - for Linux select Save files to and browse to the bottom of the file system to select /archive.

J. **Configuring Firefox**
   1. In the Tabs tab, unselect warnings.
   2. In the Content tab, uncheck Enable Java.
   3. In the Privacy tab,
      - at "Firefox will:" choose "Use custom settings for history".
      - Select "Clear history when Firefox closes", click Settings and check all History selections, plus "Saved Passwords" and "Offline Website Data" and click "OK".
      - at "When using the location bar, suggest" select "Nothing"
   4. In the Security tab, unselect "Remember passwords for sites".
   5. Click "Close" to move on.

K. **Configure Tab Mix Plus**
   1. In Firefox, follow Tools to Tab Mix Plus Options and select.
   2. In the Events tab,
      - under Tab Closing, for When closing current tab focus, select last opened.
      - under Tab Features, Max number of closed tabs to remember enter 50 and select.
   3. In the Display tab, under Tab Bar ("Show on Tab bar")
      - Select "New tab button" and "on Left Side".
      - Select "Close tab button".
      - Unselect "All..." and "Extra..." options.
      - For Hide the tab bar, select "Never".
      - For When tabs don't fit width, select "Multi-row".
      - For Max number of rows to display, select "5".
   4. In the Display tab, under Tab
      - Highlight "Current tab" only.
      - for "Show on Tab" unselect "Close tab button".
      - for "Tab width" use 30 to 250.
   5. In the Session tab,
      - select "Enable Session Manager" and "Enable Crash Recovery" only.
      - On Start/Exit tab, for When Browser Starts, select "Ask Before Restoring".
      - For When Browser Exits, select "Save Session".
      - For Startup Session, select "Last Session".
      - In Preserve tab, select everything.
   6. Click "OK" to move on.

L. **Quit or Reboot to Save**
   1. close any open windows
   2. right-click on and unmount any USB flash and remove
   3. follow Menu / Shutdown to Reboot computer
   4. tab to and select "SAVE TO CD" and select "SAVE" then "OK" to reboot
   5. the saved configured Puppy comes up

Files in the /tmp directory are not saved to DVD at the end of a session. Files in the /archive directory are saved to DVD, but not recovered in the next boot. Otherwise, changed files are saved to DVD without overwriting the older versions, and only the most recent version recovered on boot. This tends to archive the progress of a project over time in a way that does not occur in normal computer file systems. Each different session of files on the DVD can be read under Linux or Windows.

DVD Issues

Optical storage simply is not as reliable as hard drive storage. Although DVD problems tend to be infrequent, in the Puppy application they can be quite costly. After every end-of-session save, Puppy announces: "Have saved session to live-DVD (unless it has not which is an error)." This amusing message unfortunately becomes much less funny when the session really has not been saved.

Because optical storage is less reliable than we want, verifying an optical burn is not optional, unless we are happy to lose the data, in which case why are we saving it? Puppy does not verify end-of-session DVD writes. In general, it would not be uncommon for an optical write to not verify, and then Puppy should allow the user to select re-trys, try new discs, use USB drive storage and other options to prevent loss of user data. Some user control is available in Puppy by using the "save" bulls-eye on the Puppy desktop

before ending the session. The "save" icon writes the session changes to DVD without shutting down, but it also does not verify the result, and it is not clear whether a write error would be recognized. Currently, using "save" without a normal end-of-session save does produce a "unclean exit" error for "X" on the next boot, which is easily ignored.

Since all storage systems are somewhat unreliable, our Puppy response is just to be more rigorous than usual. For example, I manually back up important local work (like this article, during development) before the end of every session. I copy work files to a USB flash drive, I send the files to myself as email attachments, and save them to a Windows drive, if present. An email save generally takes under 2 minutes, total. A flash save generally takes under 1 minute. Only after everything has been saved do I shut down.

Sometimes upon restart Puppy comes up (the splash screen shows), but then fails upon reading the last saved session. We can void the last session by starting Puppy again and entering the command "puppy pfix=1" at the splash screen input.

Rarely, we can find that the last session save has made the disc completely unreadable, at least for boot purposes. Then we need to start over with a new disc. It is useful to have made copies of a fully-configured system, but it can be quite difficult to copy an unclosed multi-session DVD.

One way to "copy" configured Puppy DVD's is to first boot from a fully-configured DVD. Then load a blank DVD and burn a new .iso. There is a lot of drive door opening and closing going on, and I have found it important after any door closing to watch the burner LED and wait for it to settle down before issuing a burn or verify command. Then follow Menu / Shutdown to "Reboot computer", which triggers an update burn on the new disc. All this is more tedious than one might hope.

A better option might be to follow Menu / Setup to select "Remaster Puppy live-CD". The goal is to end up with a slightly-larger .iso including Firefox with add-ons and their configuration which can be saved as a file. Presumably we would update this from time to time. Unfortunately, I have been unable to make that include the Firefox add-ons and their configuration. Possibly the Firefox FEBE add-on could recover a saved configuration in a different way.

DETAILED COMPUTER SECURITY LIST

This list of "do's and dont's" started out much shorter. We should all be sorry that security is this hard:

1. BACKUPS
    - Copy each new user file to at least 2 other storage devices.
    - External: USB flash drive or hard drive, try SyncBack Free.
    - Web: try Dropbox, SpiderOak, SugarSync, DropIO, IDrive or JungleDisk.
    - Periodically save a system image to an external hard drive.
    - Noncommercial: try Macrium Reflect Free.
    - Commercial: try PING.
2. PASSWORDS
    - All password security depends on using long, improbable values.
    - We hide the correct password or key by making it a "needle in a haystack," a random selection from among a huge number of similar-appearing possibilities.
    - It is vital to use a separate password manager to handle long, random passwords or keys (e.g., LastPass.com).
    - Make and use a different random password for every device and every site and every account.
    - Make 15 random characters the minimum password length, but make each as long as the site or equipment can accept.
    - Email safe to partner for emergencies and backup.
3. WIRED AND WIRELESS ROUTER
    - Use an external hardware router.
    - Use wired (CAT5 ethernet) connections to the greatest extent possible.
    - 1. Set up a unique user name.
    - 2. Set up a long random password.
    - 3. Disable WAN remote management.
    - 4. Disable Universal Plug-n-Play (UPnP).
    - 5. Enable the router firewall.
4. WIRELESS ROUTER
    - Wi-Fi routers not certified to the WPA level 2 spec (ca. 2004) should be replaced, updated or the wireless disabled.
    - 1. Disable wireless admin access.
    - 2. Select WPA2 security.
    - 3. Set up a creative and unique wireless network ID (SSID).
    - 4. Set up a 63-character random pre-shared key (PSK) (try grc.com/passwords).
5. WIRELESS ISSUES
    - Wireless routers MUST NOT use WEP, because the WEP encryption key can be found faster than typing it in.
    - Wireless routers SHOULD NOT use TKIP (even in WPA2), (also called WPA PSK TKIP), because a fast exploit exists to falsify network control traffic. Some apparently unusual physical conditions seem to be required, seemingly making the attack unlikely, but it is the nature of the security business to be surprised. A successful attack might be able to spoof a critical ARP packet, leading to DNS poisoning as a way to move the user to a spoofed site for phishing a userid and password. However, SSL should fail to connect so the user should not be entering anything anyway, if we can trust the user.
    - Wireless routers SHOULD use AES-CCMP (usually WPA2), (also called WPA2 PSK AES) because only this remains secure.
    - *Any* form of password protection can be attacked by brute force (exhaustive search), which is why we always need long random values.
    - WPA security extends only from laptop to router, so packets are exposed to the router owner, and perhaps all of the local

subnet, as well as being exposed on broadband as usual.
    - Always assume that every packet on broadband can be examined by anyone in the neighborhood.
    - It is crucial to be using SSL (**https://**) when entering web passwords (try SSLPasswd Warning add-on).
    - **USE SSL! WHENEVER POSSIBLE, USE SSL!** (try facebooksecurelogin add-on)
6. OPERATING SYSTEM
    - Consider using Puppy Linux from DVD without a hard drive.
    - Consider using Puppy Linux from DVD.
    - Consider using Puppy Linux.
    - Keep a Windows OS up to date and apply all patches.
    - Use a simple software firewall with unnecessary exceptions disabled.
    - Use OpenDNS (208.067.222.222 and 208.067.220.220).
    - Use "Spybot - Search & Destroy" in Windows to "Immunize" against attack.
    - Disable autorun in Windows:
      Use the Microsoft PowerToys "Tweak UI": select My Computer and AutoPlay and disable in every possible way.
      Create and run "NOAUTRUN.REG" as in Nick Brown's Blog.
    - Protect flash drives from infection:
      Instead of flash drives, use SD flash cards with write-enable switch off, in a plug-in USB reader.
      Create a directory called "AUTORUN.INF" in the root of each flash.
    - Use a lightweight realtime antivirus scanner to check disk writes (e.g., Avira).
7. BROWSER
    - Never use or allow a browser to save your passwords.
    - Currently, use Firefox with Adblock Plus, NoScript, WOT and BetterPrivacy add-ons.
    - Where possible, use Firefox under Puppy Linux.
8. BROWSING
    - Always mouse over links and examine before you click.
    - Wherever possible, use SSL (technically TLS, the **https://** protocol).
    - Without fail, always use SSL when typing on-line passwords.
    - Never approve a *new* SSL certificate.
    - It is OK to use an *existing* SSL certificate that is somewhat out of date.
    - It is OK to use an SSL certificate for a subdomain (e.g., www.site.com instead of site.com).
    - Never get talked into exposing a password.
9. EMAIL
    - Use webmail (on-line email) because they scan your mail for malware better than you can, and before giving it to you.
    - Use webmail that offers SSL connection for all pages, not just the long-on password (e.g., Gmail).
    - Read email attachments on line.
    - It is never OK to download and run an unexpected email attachment, even if it *claims* to be from someone you know.
    - Mouse over email links and examine before clicking.
    - Avoid the most popular PDF reader (try Foxit).
10. HARDWARE
    - On a new OS install, consider changing drive partitioning.
    - Create a 40GB system partition which can be imaged and restored without affecting files on another partition.
11. MALWARE RECOVERY
    - Prepare a response to finding malware on your system.
    - Make system images periodically.
    - Use anti-malware with download protection (e.g., Spybot S&D).
    - Scan for malware and remove tracking cookies periodically.
    - Do not remove running malware programs; recover an image from before they came, or re-install the OS.
12. TRAVEL
    - Assume the whole neighborhood is on the same sub-net and can see your every packet.
    - Good security requires you to carry your own laptop.
    - An alternative to your own hardware is to reboot a local computer into Puppy Linux from DVD.
    - Best security boots Puppy Linux from DVD on your laptop from which you have removed the hard drive.
    - Avoid carrying any power-off data storage (e.g., hard drive or flash). Instead use SSL access to cloud email or documents.
    - Take along a small external router and use a wired CAT5 network connection whenever possible.
    - Your travel router could be wireless, and conveniently pre-configured for WPA2 AES to your particular laptop.
    - Secrecy is attained by the user establishing SSL (or VPN) connections.
13. NETWORK INSECURITY
    - ***No computer on a network can be considered secure*** .
    - Keep private data on an external drive or a separate machine.
    - Do not get on the Net with really private data in readable storage.

THIS DOCUMENT

This document started as an attempt to explain the curious fact that the efforts of a great many very smart people have not managed to make PC's secure. The attackers are, in fact, winning.

My original goal was to fix security holes in Windows. I expected to find a recipe of settings and added programs that would make most users secure, and much of this document is just that sort of information. Sadly, I ultimately was forced to accept that good Web security is just not available within the current Windows environment.

Absent an actual winning approach, our best bet is to move away from using Windows on the Net. The first step in malware protection is to use any OS but Windows for browsing. The second step in malware protection is to boot your browsing OS from DVD.

For some reason, everyone seems to have gotten stuck on the idea that the way to win against malware is to make perfect software with perfect security protocols. But that is only "winning," if you never want to end the race. Perfection is impossible in the real world, and if we are to win, we must must tolerate reality.

In my view, Microsoft and their legions of committees and doctoral experts have been working the wrong end of the problem for many years:

- Malware scanning will never detect all malware, and
- Patching programming errors will never close all holes.

The way to win against malware is to detect the changes successful attacks make in known files or memory images, then fix those things. If we have to do this on startup or shutdown, so be it. If we could detect malware changes to memory in real time with a separate hardware processor, so much the better. Or new hardware designs could prevent critical files and memory images from being changed by anything without both data authentication and owner authorization.

At The OS Attack Surface

From the point of view of the operating system, attacks come from the "outside," so there would seem to be an opportunity to identify and repulse attacks before they succeed. One of our problems is that we cannot specifically define what an attack looks like. We might define an attack as "an external interaction that changes our OS in undesirable ways." To understand the effect of data flowing from outside to inside the OS, and then classify it as dangerous, we must understand what the OS will do as a result of the data. To understand OS actions, we must infer the entire inside context of the OS (and the running applications it contains), which is *just what we try to avoid* by thinking of "the OS" as a single unit. Knowing what the OS will do, without knowing the vast dynamic state inside, is just impossible. Externally identifying a new attack (i.e., an outside interaction with the danger quality) by inspecting only outside data (seen, for example, by a firewall or a malware scan) is necessarily difficult, and absolute certainty is virtually impossible.

Improve Internet Design

The possibility exists that some design for outside surface interactions could guarantee security based on outside data alone. While it is unclear whether such a design is possible, our real world has nothing like that. In the real world, Internet protocol design has never had (as far as I know) a goal of identifying dangerous interactions. Indeed, bad interactions are not even defined in terms of outside data. Dangers are generally seen as "higher level" issues, and quite independent of lower-level protocols and data. Unfortunately, the low-level data are all we can see from the outside, and the only information we have to infer danger. The Internet re-design approach seems unlikely to be much real-world help, at least in the near term.

Attack Success Means Detectable OS Damage

An attack which is unidentified and not repulsed may be successful and essentially become part of the OS itself. Attacks that are not stopped damage the OS, so the OS can no longer be trusted with any form of protection. Surely OS damage is undesirable, but far worse is an inability to know with absolute certainty that damage has occurred, where it is, and how it can be corrected. An inability to identify and correct OS damage (which is, after all, just wrong data in memory or on a hard drive), must be seen as a fundamental failure of security design in the OS itself.

Hardware Boot Protection

One way to start winning is to prevent malware from modifying our saved critical OS files so as to get itself re-loaded on every boot. Probably the OS will need modification so critical files are not normally changed in operation. Then the critical files can be collected in a single directory or partition, and protected by serious hardware control.

Perhaps the hard drive would prevent changes to critical files, and must do so even if OS file system was subverted and file attribute read-only bits ignored. Perhaps the hard drive would accumulate new file versions but not perform replacement until each had full cryptographic authentication coupled with an overall owner authorization. The drive should hold multiple configuration steppings, to make it possible to revert to earlier configuration.

Sadly, our PC hardware design does not include selective hardware file protection, nor is it likely that the current OS could use that if it existed.

OS File Check and Repair

Absent selective file protection in hardware, it is at least possible to imagine periodically checking and absolutely repairing every single critical file. Check and repair does not mean scanning for viruses or malware, it instead means checking each critical file for existence and contents, and correcting whatever needs correcting. To avoid rootkit effects, such checking probably would require a reboot into a "live" DVD, and may further require a deep analysis of the registry so it could be similarly repaired. Device drivers would be another issue, as would hardware BIOS flash memory contents on various boards.

Boot from DVD

Another possibility is to boot our OS from some form of fixed or read-only storage, such as a DVD. Although much slower than a hard

drive, the DVD boot automatically destroys previous memory contents including malware (excepting, of course, BIOS infections). Unfortunately, the large size of the Windows OS may make DVD reboots so undesirable as to be almost unusable. Even if we accept the inconvenience, we just end up with Windows again, which is the malware target.

Since software updates are our reality, it would make sense to use the multi-session capabilities of DVD+R or DVD+RW storage. Any such system can and should involve cryptographic authentication that malware cannot provide. Unfortunately, software control necessarily depends on the OS not having been subverted, which may not be true. Absent separate authentication hardware in the DVD reader, this whole approach may have only temporary worth. Of course our systems are themselves temporary, and when the next step becomes obsolete, we may have better hardware to deal with it.

Boot from Hardware Protected Flash

Perhaps more ideal than DVD, and certainly faster, would be some sort of flash memory boot. Flash memory itself is hardware, but here we refer to hardware *protection*, the ability to prevent unauthorized changes to critical files. Common USB flash drives seem unlikely to be very helpful because the critical security aspect is not holding or reading data, but instead preventing malware from changing that data. Write-protection must occur in hardware (or firmware in a separate hardware system), because when malware is present the OS has been subverted. Ideally, we would have full cryptographic authentication and hardware selective file protection inside a USB flash boot device, all of which is perfectly reasonable. What we need is a modified OS and a USB flash boot device with selective file protection, update authentication and owner authorization for updates.

Terry Ritter, his current address, and his top page.