Get a PDF

Software Security Myths and Facts

The author's irresponsible, immoral, and rather bleak **personal** opinion

By Eugene V. Bobukh

Fact: There is no such a thing as "secure" browser

CVE is commonly considered as the most trusted independent data source for tracking security defects in software. Let's check out its records for 2009:

Browser [all versions]	**Public security vulnerabilities in 2009:**	**Market share as of 01/2010, for reference (per [05])**
Microsoft Internet Explorer	72	62.12%
Mozilla Firefox	120	24.43%
Google Chrome	34	5.22%
Apple Safari	67	4.53%
Opera	40	2.38%

As you see, each browser had dozens of publicly disclosed security holes in just one year.

Yes, this was a "dirty" search by keywords only so it contains notable noise. Yes, vulnerabilities of different nature and impact are all mixed in one lump here. Still this is sufficient to justify the point: none of major browsers is free from security defects. Even Google Chrome with its security-in-mind architecture is far from that, as having 34 issues in a year does not really count as "clean" :)

Of course, [nearly] all those defects are timely fixed by manufacturers. But the mere count of publicly known issues suggests there are plenty of unknown ones. Alas, there is no magic "safe" browser. Forget about it.

Myth: Firefox is more secure than Internet Explorer

Or MacOS is more secure than Windows. Or Internet Explorer is more secure than Google Chrome.

Some of those statements are merely illusions some people hold to. And the rest are unconfirmable.

The reason is that nobody really knows today how to meaningfully compare security of different products. Well, except for the most egregious cases, maybe.

"Wait a second" -- you'd say, -- "but what about the number of vulnerabilities per year? That looks like a good yardstick. 67 is less than 120, thus Safari *must be* more secure than Firefox?"

I'll give Firefox fans couple minutes to finish laughing, cool down and put aside their bats... then proceed.

A simple comparison of the vulnerabilities count is flawed for many reasons. First, it's affected by product market share: the greater it is, the more attention is paid to the software, and the more issues tend to be discovered. Then, as was already mentioned, vulnerabilities are not "born equal". Some are severe, some are almost benign. Simply throwing them all on a scale and weighing doesn't make much sense.

So this is difficult. There are nearly as many ways to compare security as those doing the comparison, and each way has its peculiarities.

If you say your product had fewer security holes last year, "they" would claim that it's not the number of holes, but the time they remained un-patched that matters.

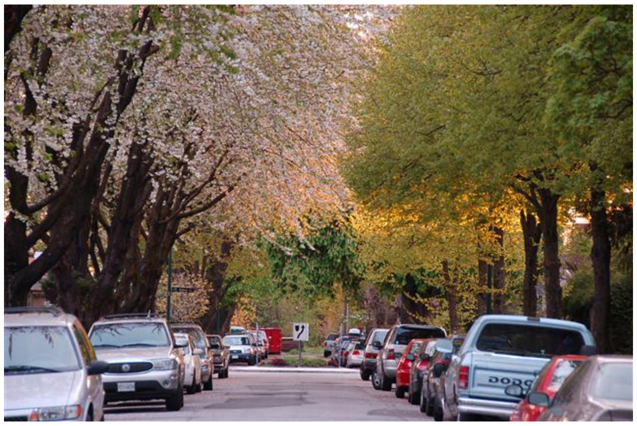
If you show that your patching time is better **and** you've had very few security holes, "they" would say that too few vulnerabilities means not many people have looked at your product and thus it is under-tested so it must be full of bugs. Chuckling? Please don't. I have heard people seriously making this argument, believe it or not.

You can say that it takes an average hacker 5 minutes to break into "theirs" software, while yours survived the attacks for 5 days. "They" would respond it's because nobody's really interested in your software and thus there is a lack of in-depth research on breaking it.

Frustrated, you would go back to CVE database to count vulnerabilities in each class of the severity. So what? "They" would point out that CVE is not objective since not all software vendors report all vulnerabilities. And with that argument, you can blame just about anything on conspiracy :)

And this is only the tip of the iceberg. Underneath, there are way tougher problems. For instance, some products are "secure" in a particular environment or use scenario, and not in another, with non-default options enabled. How do you measure that?

In a sense, asking "which product is more secure?" is like asking "which picture is greener?"



The one with more green dots? Or the one with fewer red dots? Or the one with the greenest dot? Or the one which is greener after you blur it to suppress any detail? Depending on what is important, you can name either picture as the winner.

Measuring software security is a similar problem. It is a strange and complex mix of many things. It includes the number of vulnerabilities found and not found; their severity; the number of customers affected by defects and the total number of customers; the time vulnerabilities stayed unfixed, and many other parameters. And of course some plain, blatant hype and PR.

People have yet to come up with a universal measure for software security.

So usually I ignore statements like "A is more secure than B!" I know how a slightest change in the way "security" is defined can turn a statement of that kind into its opposite. Rather, I stay with this: all major products with similar functionality have roughly the same number of security problems in them. That's because they do similar things, have similar complexity and, as a result, contain approximately the same number of defects in them.

Fact: During the 2008 CanSecWest PWN 2 OWN security contest, the MacOS was taken over by the remote hacker in 1.5 minutes

I did not even get a chance to grab my camera.

Windows Vista survived for two days, while Linux stayed alive for the whole three-day competition. Though to be fair, only one person was working on breaking it.

Is that to say that MacOS is worse that Windows? No. Just that there is no perfect "secure" OS.

Fact: Client side antivirus software does not offer much protection and can be even dangerous

Overall, the very idea of catching "known viruses" is a day-before-yesterday dinosaur of software security which should've been extinct for 10 years already. Somehow it survived though, so it deserves some attention.

Two things are important to know about the realities of antivirus [AV] software.

First, sometimes it is impossible to predict what a program does without actually running it. Hence you can't always tell whether a *setup.exe* received from the Internet is a virus by the power of analysis. It's impossible. Naturally, this is a variation of Catch-22: in order to learn if a program is unsafe, sometimes you have no other option but to run it. Remotely related to the famous Halting Problem, in practice this situation is caused by the mind-boggling states transition complexity that an EXE can exhibit during running. The intricacy of it defeats even the remote possibility of a complete analysis.

Now imagine an antivirus that receives an EXE (or a script) and tries to make a "friend or foe" decision. Harmful or benign? Malware or useful? Time is limited, the user is waiting, and by the way here is some bad news: there is no way to solve this problem precisely!

Always being in that stalemate, antiviruses have to resort to fast but less accurate techniques. Those include signatures recognition, heuristics, and attempts to partially emulate execution of the code in question. Unfortunately, none of them are completely reliable.

If you run contemporary AV over a large sample of old **known** viruses/malware/trojans, the results would be impressive. An average AV has a detection rate of **known** malware around 95%, and the best antivirus products reach 99.8% on that benchmark according to testing conducted by AV-Comparatives.org [10]. Not bad at all!

It looks much bleaker when anti-virus programs face new malware not seen yet. With signature detection impossible, AVs have to resort to all kinds of fortune-telling and guessing. The difference is striking: the best product [as of 05/2009] offered only a **69% detection rate**. And typically across the industry it is in the range of 20-60% [20, 30].

This does not imply that anti-virus authors do a poor job. It is just fundamentally impossible to get it done much better. Malware authors have tons of evasion techniques at their disposal. They can create variations -- and for each new virus, sometimes thousands of modified forms are released into the wild to confuse and overload the detection logic of known AV engines. Then, viruses can encrypt themselves. They can re-shuffle and re-package their payloads in incredibly complex ways. They can mutate on the fly and download more code from the Internet. They can carry their own execution engines that interpret the payloads written in secret "languages".

Overall, I don't think there is a way to make antivirus detection much better than the current ~70%. This is a figure you should keep in mind: even the top-notch antivirus products offer only ~70% protection against new attacks.

What does it mean practically? It means that if you rely upon AV as your only defense against trojans, and have poor browsing habits, your machine will get compromised in a few months. Simply because hundreds of new viruses are created each year, and it takes only 3-4 attempts to bypass the AV.

So don't bravely open strange attachments just because your AV is "up to date". The AV should be the last, not the first line of defense -- and it is a pretty weak defense actually.

The second important thing about antivirus programs is their complexity resulting from dealing with complex input. A typical AV needs to parse thousands of non-trivial file formats to scan for bad code. Nearly any well-known file type -- .exe, .doc and .docx, .pdf, .xls, .wmv, .mp3, .vb -- must be understood by AV engines, and that carries the risk of errors.

Opening a file to scan for malware implies parsing. Parsing means dealing with memory, as antivirus has to re-create some parts of the input file in its memory to analyze them. Typically, hundreds of memory operations are required for each file. Multiplied by the many thousands of supported formats, this gives millions of complex memory tricks an antivirus needs to perform flawlessly during its daily business. And often it takes only one memory error to introduce a security hole. A tiny buffer overflow or even a simple crash may result in malware injecting itself into the antivirus engine and running there, converting your ally into a covert resident enemy on your computer.

Naturally, this is paradoxical: the more file formats AV try to understand in order to trim down customer risk, the more risky they become! For any AV architecture there must be a practical limit beyond which supporting more file formats means **less** security. Are we at that point? I'm far from knowing for sure. But I on my home machine I keep my AV shut off. I don't let it swallow just whatever crap may arrive from the network. And when I need scanning, I make it an explicit action.

So to summarize: good browsing habits and being precautious with strange email is what really protects one from viruses. Relying on AV security? Not as much.

Myth: The press is a true, reliable, and adequate source of information on security issues

In fact, it's usually the opposite. Most journalists are outrageously undereducated on software security. Nevertheless, it does not make them shy when it comes to making bold statements in the press :))

Use CVE or software manufacturers' databases [such as the Microsoft Security Bulletin Search] for looking up accurate data on security vulnerabilities. And don't trust TV or newspapers too much... their goal is not to give you objective information. Their goal is to sell well.

Myth: Viruses can be reliably removed from infected computers

Well, in some cases they can. The problem is, it is impossible to know for sure whether the removal was successful :) At least, if a virus had a chance to run as root/Administrator just for a second. Often, even smart people from the IT industry do not realize that. The Internet is full of dialogs like this:

> A: i've got a virus! my <*browser name*> automatically redirects me to the porn site!
>
> B: oh yeah I know! this is an <*XYZ virus*>
>
> A: what should I do now?
>
> B: oh, it's easy, just run the <*ABC scanning & removal tool*> and/or delete the HKLM<*Foo**Bar*> registry key and youll be fine.
>
> A: (In one hour) thank you thank you thank you! it's gone! my machine is clean and runs as before! thanks for saving my online banking, whoa!

Wrong. A malware author is probably thinking "whoa!" at that moment, too. Here is why.

In Windows, the Administrator is God **.** The Admin can control **each bit** of the information on their machine -- on the hard drive, in memory, and of course all pixels on the monitor. No barriers like ACLs can stop well-educated Admins.

So let's take a look at some nasty virus running on the system as Admin and therefore having unrestricted access to each and every bit of information on that computer.

A happy user runs a scanning tool and "removes" the malware. The malware pretends to be removed. It displays the "scanning tool" on the screen [remember? each and every pixel!], then displays fake "registry" and performs fake "key removal". With none of that happening in fact. Finally, it goes to sleep for a while to calm down the user. The end result: the virus is still on the system. The user thinks it is gone.

Obviously this is a contrived example made up to demonstrate the very possibility. No malware trying to conceal itself would go that complex path. Instead, it would probably just hook file and system APIs to become invisible to your "ls -a" or "dir /s" or Task Manager. Examples of this kind of behavior are known, the most famous of them being the now-classic Sony rootkit ([40, 50]).

Yes, 99% of viruses do not go to such extremes of stealth. Yet it is **possible**. Plus, remember that many viruses come in hundreds of variations. A "cleanup tool" may simply not know how to deal with a particular one to completely remove it from the system.

Sure, you can take chances and keep doing banking on the machine where the malware has resided once. Maybe it is truly gone. Most likely it has. But **you will never know**. There always will be a 1% chance remaining that this machine is not yours anymore. That the browser you are using is not yours anymore. That the bank account you are visiting is not either. And neither your SSN, backups, personal pictures, music, your girlfriend hot talks ICQ logs -- none of that.

So if you really want to be sure these assets are protected, and you've got a malware/trojan/virus/infection, you will:

1. Unplug the machine from the network.

2. **Nuke** your hard drives. I mean, format them all, from A:\ to Z:\. If you have a writable BIOS, reset it, too.

3. Install a clean OS.

4. Immediately apply all possible service packs and patches.

5. Restore your data from backups.

6. And turn on automatic updates, if you haven't done so yet. I'll speak more about that in a minute.

Did I say "backups"?

Yes, that's right. Backup like crazy.

Forget about security holes for a second. Just consider this: all hard drives are mortal. They have finite lifetimes, and those are short -- we are talking about 3 to 10 years on average.

Disk failure and data loss is not a matter of "if". It is a matter of "when". When the hard drive fails, and all data on it is gone to the Great Realm of Entropy, and there is no backup, what will you do? I mean, besides banging your head against the wall? :)

I have seen people who kept only a single copy of all their important data. When they lost it, their lives became miserable, and so they were for quite a while thereafter. They've lost everything: work-in-progress, address books and contacts, legal documents, and of course music, photos, logs, software -- you name it.

So if you do not have a backup copy of at least **the most important** files of yours, stop reading right now. Stop and copy that data to another machine, or to a writable DVD, or onto a Flash USB drive, or even email it to yourself as an attachment. Seriously. Do not resume reading until you have done so. If there is any valuable advice in this article, then this is it: backup! And if this was the first backup in your life ever, believe me you'll be much happier when the data loss eventually happens.

Of course, a single backup is better than nothing. But it is hardly helpful if the data in it is 2 years old. So the next point is this: a good backup is something that happens regularly, at least a few times per year.

If you are reasonable like I am [although some folks call it "paranoid"... indeed strange they are :))] you'll probably go even further. You'll be backing up daily and to several locations ideally in different countries, so that only a global nuclear war would destroy the most important of your data. And of course, your backups would be encrypted :)

Why am I giving so much attention to the subject of backup here? Well, I'm trying to build a realization that in today's world security holes are inevitable. Hope for the best, but be prepared for the worst. If your data is backed up, the recovery in a case of a security issue is much easier.

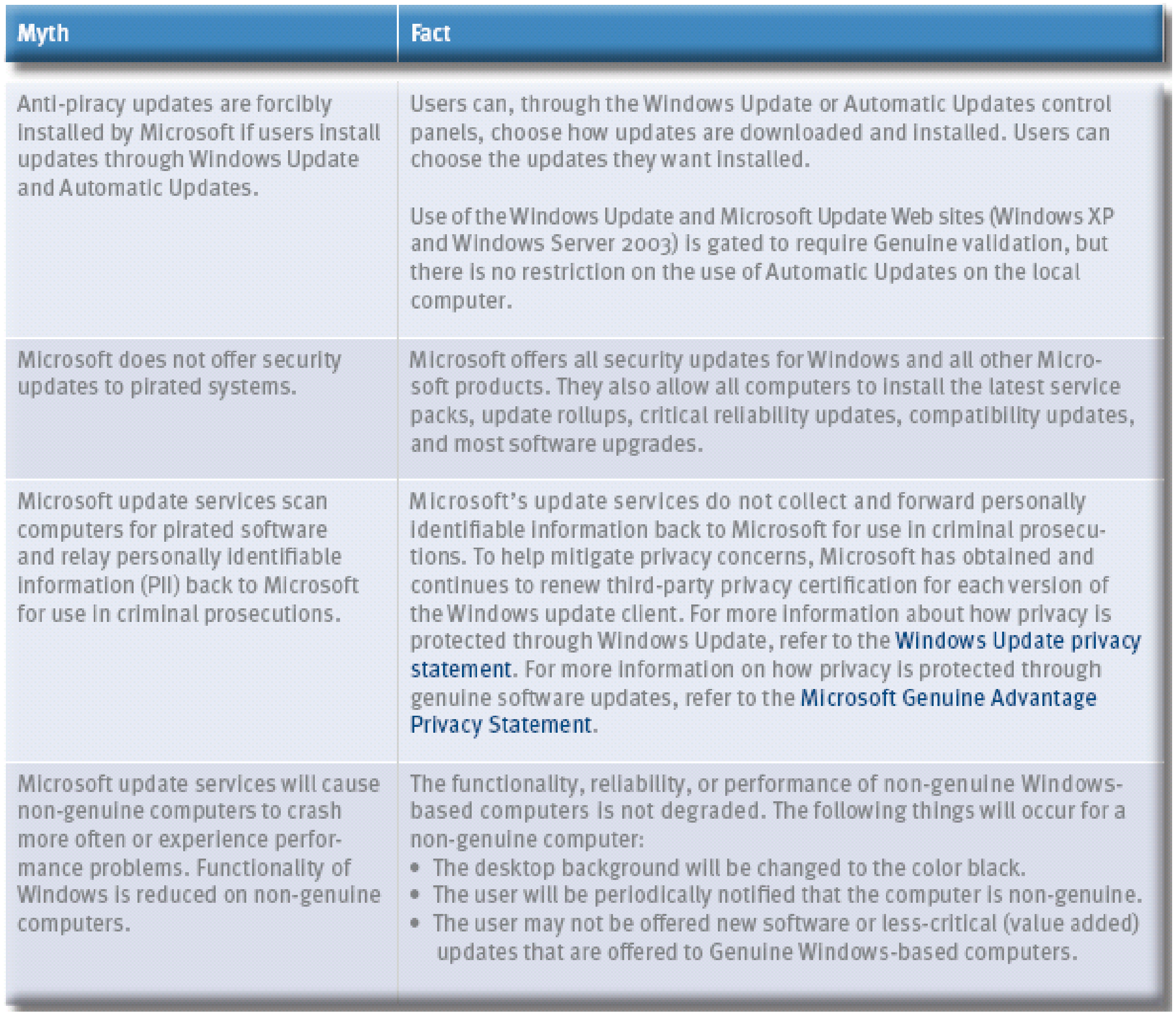
Fact: Automatic security updates are vital to your computer security

No, they are not created to sniff the user's data or pin-point piracy across the world :) No, they are not to push more bugs onto users' machines, so that Bill Gates would laugh menacingly each time anything crashes :) Yes, they are usually well tested, because the failure of an update is an enormous headache for the software company in the first place.

Linux, MacOS, Windows, Safari, IE, Firefox, Chrome, MS Office, Adobe Acrobat, iTunes, Flash -- they all have had **dozens** of security holes already, and more are discovered at the rate of 10-100 per year. Yes, a new one each month or sometimes more frequently.

If you turn off automatic updates, your software becomes vulnerable to attacks from the Internet in couple months, if not days. That's why all major products can automatically update themselves today.

Here is a nice table called "Myths about Microsoft update services and software piracy" from [170 (by Microsoft)]:



Myth	Fact
Anti-piracy updates are forcibly installed by Microsoft if users install updates through Windows Update and Automatic Updates.	Users can, through the Windows Update or Automatic Updates control panels, choose how updates are downloaded and installed. Users can choose the updates they want installed. Use of the Windows Update and Microsoft Update Web sites (Windows XP and Windows Server 2003) is gated to require Genuine validation, but there is no restriction on the use of Automatic Updates on the local computer.
Microsoft does not offer security updates to pirated systems.	Microsoft offers all security updates for Windows and all other Microsoft products. They also allow all computers to install the latest service packs, update rollups, critical reliability updates, compatibility updates, and most software upgrades.
Microsoft update services scan computers for pirated software and relay personally identifiable information (PII) back to Microsoft for use in criminal prosecutions.	Microsoft's update services do not collect and forward personally identifiable information back to Microsoft for use in criminal prosecutions. To help mitigate privacy concerns, Microsoft has obtained and continues to renew third-party privacy certification for each version of the Windows update client. For more information about how privacy is protected through Windows Update, refer to the Windows Update privacy statement. For more information on how privacy is protected through genuine software updates, refer to the Microsoft Genuine Advantage Privacy Statement.
Microsoft update services will cause non-genuine computers to crash more often or experience performance problems. Functionality of Windows is reduced on non-genuine computers.	The functionality, reliability, or performance of non-genuine Windows-based computers is not degraded. The following things will occur for a non-genuine computer: • The desktop background will be changed to the color black. • The user will be periodically notified that the computer is non-genuine. • The user may not be offered new software or less-critical (value added) updates that are offered to Genuine Windows-based computers.

The bottom line: if your software has automatic security updates, turn them on. Even if you run a pirated version of Windows -- be a good citizen, please do so! Don't participate in spreading the disease by leaving your computer without an immune system and getting infected with some crap. Nobody is after you through the updates.

And if you chose not to have the "bloody" automatic updates "messing" with your machine, then please don't blame manufacturers later when the machine gets a nasty rootkit.

Emotional people might exclaim here: wait a second, are you blaming me [the user!] for having security holes in your [manufacturer's!] software??? You must be kidding! Learn how to write programs without bugs, and then you just won't need those stupid updates! Difficult? Fire crappy programmers and hire good ones!

To that the answer will be:

Fact: Even if there is bug free software, it is useless

Because it is either so primitive that it does not do anything useful, **or** it is more expensive than the Space Shuttle and virtually nobody can afford it :))

Everything else has bugs, errors, defects in great abundance, and nobody so far has figured out a way of getting rid of them completely. "Nobody" means not just a company like Microsoft or IBM or Google, but no individual or group ever so far.

Personally, I think the problem has its roots in human nature. We humans are very good at making mistakes. Some of them propagate into design and code of the products to become vulnerabilities. According to [80 ("Software Reliability" By Hoang Pham)], **professional** programmers make about 6 errors per 1,000 lines of code. In typical commercial software, the "normal" post-release rate of defects is 0.1-20 per 1,000 lines of code. Since browsers and operating systems consist of millions lines of code, you can see how it comes they are peppered with issues.

Preventing and catching many errors is definitely possible with certain tools and practices. Allegedly [90, "Code Complete" by Steve McConnell], NASA has achieved programming quality of less than 0.02 errors per 1,000 lines of code. That is a tremendous feat. Yet this is not zero if you consider programs of any practical size, and the cost of NASA's software development is usually prohibitive for businesses or consumers.

So the question of "how much quality?" boils down to the development cost. The more money is spent, the cleaner the resulting code is. But apparently throwing even billions of dollars on purifying software does not remove **all** defects. Check out the space exploration industry, where quality standards are probably the highest across everything that humans do, and yet where dire consequences have been seen due to bugs in programs:

* A software error caused the explosion of Ariane 5 rocket 40 seconds after the take-off.

* Outdated software onboard Soyuz TM-5 spacecraft resulted in a one-day delay of return from orbit and put the lives of the crew in serious danger.

* An erroneous program uploaded to the Soviet Phobos-1 spacecraft *en route* to Mars deactivated the attitude control system and caused the loss of mission.

Numerous and costly software errors have been seen in other mission critical industries (e.g., military), too.

Sadly, there is no such thing as bug-free programming today. And any truly reliable software costs **a lot** to develop.

It is interesting to note that most of that cost is associated not with some super-developers making only one error in 25 years of working :) Certainly, high profile projects attract the best people who are paid well. But in the companies with high development standards a lot (or probably most) of money is channeled to tools and processes that improve the quality of code as it is being written, or right after that. That means the tools that make day-to-day programming easier. That means smart compilers capable of detecting poor coding patterns. That includes static code analysis tools which reveal more flaws. And sure enough, that includes testing, which is a huge and complex world in itself. So, no humans -- just tools and processes.

To close on this, here is the famous story about Knuth's reward check [110, 120]. I guess it proves that even the world's best engineers make errors even in relatively small programs :)

Hypothesis: People's irrational perceptions primarily define the quantity of security holes in the industry

As we saw, software companies can never drive the number of defects to zero, though they can cut them down by investing extra effort in the process. The question is: how much to invest?

Had the world consisted only of people like me, Microsoft still would've been selling NT4 as their primary OS, and *nix folks would've been sitting in Afterstep 1.8.11. But those pieces of software would've become reliable like the law of gravity. "Blue Screens" would have been seen no more often than a blue moon and priceless screenshots of those would've been shared on discussion boards. And yes, there would have been 5 times fewer features in anything than today.

However, the world is not mine... yet :) It is full of customers. Customers are humans of special kind who want completely different things. They want features. Rich, abundant, colorful, playful, moving-singing-dancing-animated features for everything. Moreover, they want a dedicated feature for each basic action. There must be an "**Email**" button, a different button called "**Internet**", and another one for "**YouTube**". Not to mention that network connectivity must be duplicated in MS Word, in Media Player, in Real Player, in QuickTime, and in Search of course.

Software manufacturers have to obey that. They may realize that having three buttons for essentially the same thing is redundant. They may even think: "this is stupid!" But they would never say that. For these are the customers who pay them money. So companies go ahead and implement features according to customer expectations. And if they won't, their competitors will, and this can happen so quickly that you'll be out of business in couple of years if you don't follow the trend.

[I'm curious to know how many people have read at least to this point. Please let me know if you did :)]

So, software companies add numerous features. The more features, the more revenue is generated. What prevents the feature count from growing beyond the infinity? The answer is: support cost. If you have too many features which expose too many errors, the cost of fixing them and making them work becomes overwhelming.

Software businesses operate with fixed resources -- such as developers, testers, support folks, time & money. Within these constrains, only a certain number of features can be fit. If more are squeezed in, quality suffers, and the product has more crashes and security holes. And when customers are burned by too many security holes, they tend to shy away from the product.

So this is it. Too many features -- and you lose cause you can't make them all stable and secure. Too few features -- and you lose again cause customers don't perceive your product as "cool". Hiring more developers may seem like a nice way out of this problem, but in reality any extra resources are better be spent on expanding the product functionality while preserving quality at the "minimally acceptable" level :)

But who defines what is acceptable? Who controls the position of the equilibrium point?

No, it's not the manufacturer or the vendor. Nor it is the experts. It is... the general public! The average computer-uneducated majority of the population. The "average Joe" and the "average mom". They define where the "right" balance between security and functionality is, not through the scientific process, but mostly through their intuitive feeling of what "secure" and "cool" are :)

Do I blame them? No. Just describing the reality. And the reality is this: although computer pros may cry out loud about the bugginess and insecurity of the software, the number of security flaws in it will not change significantly unless the opinion of "average Joe" changes!

Why? Because there are only 100 experts out there, but 100 million Joes, and these are the Joes who unambiguously express with their purchase power what the software must look like!

Of course, I exaggerate it a bit... but fundamentally I think this is how it works. As a [partial] confirmation, consider customer groups like medical or military or aerospace. Software they use is typically more reliable than average PC programs. Is that because those customers demand higher quality and shift the equilibrium point from features to higher reliability?

You think I just came up with all this stuff on my own? Not really. Examples of irrational socially controlled equilibriums are known [140]. The best one is probably the Smeed's Law [130]. It states that the number of deadly accidents on the roads does not depend much on road quality, or rules, or signs -- but rather only on the population density and the number of registered cars. "...*People will drive recklessly until the number of deaths reaches the maximum they can tolerate. When the number exceeds that limit, they drive more carefully. Smeed's Law merely defines the number of deaths that we find psychologically tolerable*..."

In short, when you build safer roads, people just drive faster. Quite possibly, it is the same with software security: if you create more secure software, people find crazier uses for it, until most of the security benefits are virtually nullified :)

So stay educated. Ask for reliable software. Teach people around how to achieve more with fewer features. Maybe, this will help with making our world a little bit better place...

Fact: Most of today's computer infections are caused by the lack of user education

People download random EXEs from the Internet. They turn off Windows Update. They navigate the Web with IE 5.5 on Windows 2000 as Administrators. They turn off UAC. They click very strange random links received from I-have-no-clue-whom. And when s##t happens, they blame software manufacturers for not protecting them.

Seriously, can you believe that?

Let's look at some statistics:

* [150]: Of the data losses that happened in 2009, 44% were caused by a lost/stolen media/drive/computer/laptop. Only 16% are caused by hacks and 1% by virus. Well, this is not exactly about software security. But it shows that the threat of hacking fades in comparison to "good old" stolen laptops.

* [160]: Next is the list of top 10 Windows infections as of August 2009. The column on the right is the result of my research with Symantec data and shows the propagation mechanism for each malware:

Family	Machines Infected	Propagation Mechanism
Taterf	463,000	Worm. Propagates by creating a copy of itself and injecting into Autorun.inf file at all drives on the infected machine. If you view the drive of the infected machine in explorer and have autorun enabled, you get infected, too. Arguably, this is the software fault, with Windows Autorun being the precise culprit. But we can't even dream of having it OFF by default when so many people around still ask for it to be ON...
Renos	228,973	Trojan. User runs EXE downloaded via link clicking or received via attachment.
Alureon	211,441	Trojan
FakeRean	162,328	Trojan
Bancos	158,152	Trojan
Koobface	134,139	Worm. Spreads via social engineering techniques on Facebook and other social networking sites. Recipients who visit the URL are confronted with a message telling them that they must download an updated version of Adobe Flash Player to watch the video. The supplied executable is actually the Koobface installer. With a little bit of education, users could've easily avoided this.
Frethog	132,827	Trojan
Cutwail	110,840	Trojan
Rustock	90,788	Trojan
Tibs	84,081	Trojan

Ouch! 72% of computer infections as of August 2009 are the result of users downloading and running the malware **by their own hand,** often against the warning of the operating system! There no need for buffer overruns or weak permissions or other sophisticated security holes to get most of the population infected :))

* Symantec Report on Rogue Security Software from June 2009 quotes this: "*The most common distribution method observed by Symantec during this reporting period was intentional downloads, which were employed in **93 percent** of the attempts of the top 50 rogue security software scams*".

OK. **Maybe** the end users truly are the biggest problem with software security, but how does it help them? What can they do?

Really, what should I do?

1. Use caution. There is no need to be a PhD in Computer Science to avoid most trojans. My mom can barely use email, yet I'm reasonably confident in the security of her laptop. Why? Because she knows the limits of her education and won't click or run anything she is not absolutely sure about.

The following are typical indications of a computer-related social engineering attack:

1.1. It is out of context. Seriously, do you expect an embarrassing video of yours from a friend whom you haven't seen for 3 months?

1.2. It is often emotionally charged, causing you to lose your head over fear or anger or urgency. Make a habit to never reply to email or install anything if you don't feel calm and cool-headed. This is a useful habit in itself.

1.3. In the end, it asks you to do unusual things: install an update or a "tool", send money, or give a credit card number. Thou shalt think thrice before doing so!

When you detect all three signs together, be sure: it is almost certainly an attack. Report it, or just ignore.

2. Backup regularly. See the discussion on that above.

3. Your Facebook account should not be **you**. Don't store too much personal stuff on the Internet. Pretty much anything put on the Web becomes accessible by the Whole World and Their Dog in the very next moment. Exceptions to this rule are slim, and if you don't know them well, it's better to assume there are none.

4. Make sure automatic security updates are on. Right now, please :)

5. Turn off or uninstall unneeded software. It is ridiculous how much useless stuff sits on a typical computer, especially with vendors placing a lot of their code into pesky auto-run modes. Turn it off, throw it away. It will reduce your attack surface and make your machine run faster and more reliably.

6. Whenever possible, avoid running as root/Administrator.

For Linux/Unix people this is easy. For Windows folks, it's not. There is still too much software around that runs only as Administrator -- for example, Facebook. If you upload photos on Facebook via Internet Explorer, you have to install an ActiveX control. That requires Admin privileges [there are workarounds, but they are extremely cumbersome]. Nearly everyone uploads photos, so I conclude that a lot of people are guilty with running Facebook as Admin! Ridiculous. It's like giving someone a full access to everything in your house so that they can pick up your garbage.

Nevertheless, options exist, and I will list them here despite fully realizing this may be cryptic for many:

A) By default run as non-Admin and elevate when needed via one of the following methods:

A.1. RunAs.exe utility [available in Windows 2000 and all later versions]

A.2. Fast user switching [exists in Windows XP, Windows Vista and Windows 7]

A.3. Have a separate Admin Remote Desktop session opened against your box [possible and preferable in all Windows Server OSes]. Alternatively, run as Admin but open a non-Admin session against localhost for untrusted operations like browsing.

B) Run as restricted Administrator and elevate via UAC prompt [a rather modest protection, but it's on by default in Windows Vista, Server 2008 and Windows 7]

C) Run IE with a restricted token account or as a non-Admin account via RunAs.exe. This solution is technically challenging and incomplete, but it's available down to NT4 if that's what you are stuck with.

D) Upgrade to normal OS, after all :) This is the only option if your box still runs Windows 95, 98, or Millennium.

Feeling lost? Then I guess I'll have to set up a workshop some day to demo all this stuff... :)

25.03.2010

===

===