RE: Why Easy To Use Software Is Putting You At Risk

Source: http://www.derkeiler.com/Mailing−Lists/securityfocus/security−basics/2006−02/msg00363.html

- *From*: "Craig Wright" <cwright@xxxxxxxxxxxxx>
- *Date*: Sat, 25 Feb 2006 09:27:16 +1100

Hi Al

I do agree that the additions and changes to Solarius will make it more secure and that this is good. At the same time the addition of ECC and other functions from TS (trusted Solaris) will not make Solaris equivilant to TS.

In the case of the bus − people (at least for a very short time) will speak out and try to ensure that something changes (Ferries here in Sydney is another issue)

Call me old fashion, but I still like a structure approach to programme testing with both black and white box tests and

Regards
Craig

−−−−−Original Message−−−−−
From: Al Sutton [mailto:asutton@xxxxxxxxxxxxxxxxxx]
Sent: Sat 25/02/2006 9:04 AM
To: Craig Wright; 'dave kleiman'; 'Darren W Miller'
Cc: 'defendingthenet'; security−basics@xxxxxxxxxxxxxxxxx
Subject: RE: Why Easy To Use Software Is Putting You At Risk

Craig,

Just to tie this up with my other post, the move on Trusted Solaris is not about dumping an additional product because it costs too much, it's about making the base product more secure. I would disagree that poorly written code is prevailing, instead I would say that the view that security is something that must be in all code is prevailing and Sun are doing a good thing by stopping the sale of two versions of an OS (a secure and a not so secure version), and instead working towards a single reliable system.

I would also disagree that everyone should take responsibility for software failures. If I ride on a bus and the wheels fall off it's not something that

I have directly caused, similarly if I use a piece of software for a purpose it's sold for in a manner approved for my environment I should not be responsible for it if it causes problems, it's the problem of the supplier, tester, and/or the people maintaining it.

I would also disagree that rapid development processes are flawed. Extreme Programming has some great ideas. Writing the tests before the code ensures that tests are not fudged to fit in with what's written, and that the spec isn't interpreted in a way that the developer has decided because it would be easiest to code. The functionality cards concept gives a great way of showing project managers and customers that if you want to put a new card in the deck, the time either increases, or you have to take cards out of a similar time value, and although I'm not a fan of shoulder surfing programming, peer reviews are important. It's like anything, it's not all bad, there are some good things in there.

Al.

−−−−−Original Message−−−−−
From: Craig Wright [mailto:cwright@xxxxxxxxxxxxx]
Sent: 24 February 2006 12:51
To: Al Sutton; support@xxxxxxxxxxxxxxxxxx; dave kleiman; Darren W Miller
Cc: defendingthenet; security−basics@xxxxxxxxxxxxxxxxx
Subject: RE: Why Easy To Use Software Is Putting You At Risk

Hi Al

I do agree with what you have stated and in fact the whole rapid development process is flawed from a code integrity view.

I do disagree still with the terminology "prove it". However there needs to be a quality of testing that may be enforcible and in itself subject to due care. I am unsure as to if developers would choose the first code from google. generally they would in my experiance choose the least expensive. This is not to state that this is a better method ;)

It also should not be the IT comunity. It should be everyone. We all have to start taking more responsibility. Developers, engineers coders, testers, and even users. Trusted Solaris is being discontinued. This is not as it is difficult to write, but as end users do not want to pay the premium for well designed software. So poorly written code prevails.We as the IT professionals need to take a stance to change this and to do this we need to be able to communicate to the people in management and finance.

These people understand Risk and figures. Cost and Accounting. To get an understanding accross the true costs of patching and maintance of poorly design software needs to be "sold" in a manner they understand. To do this annulised costs associated with the increased risk give a foundation to the arguement.

Overall a more integrated approach to development and testing works to a far higher degree.

Regards
Craig

−−−−−Original Message−−−−−
From: Al Sutton [mailto:asutton@xxxxxxxxxxxxxxxxxx]
Sent: Fri 24/02/2006 8:01 PM
To: Craig Wright; support@xxxxxxxxxxxxxxxxxx; 'dave kleiman'; 'Darren W Miller'
Cc: 'defendingthenet'; security−basics@xxxxxxxxxxxxxxxxx
Subject: RE: Why Easy To Use Software Is Putting You At Risk

Craig,

Nobody's perfect, but other forms of engineering fair far better than software development. If you looked at the first 5 years of the software I'd expect that you'll see a figure far greater than 2.7% becoming vulnerable or failing because of a fundamental problem. From personal experience I've had to apply patches to at least 70 % of the software packages installed on our internal systems within five years of their release due to either security issues or potentially fatal bugs from issues which are well known (such as buffer overflows, SQL injection, poor handling of low storage space, poor handling of loss of power to the system, etc.).

Firewalls are routinely deployed partly because of a general lack of confidence in the ability of existing software to safely handle anything that can be thrown at it. If the same view was held of building you'd see everyone living in big domes with concrete floors which have foundations streaching tens or hundreds of meters into the ground to strictly control the environment in which the house exists.

It's interesting you mention the Hatfield Rail Crash, the cause of that was a cracked rail which was not delt with due to a poor maintainence

and
monitoring plan (see sidebar at

http://news.bbc.co.uk/onthisday/hi/dates/stories/october/17/newsid_2491000/2
491425.stm). While software does not develop faults over time in the
same
way, a poor maintainence and monitoring plan combined with poorly
written
software will leave systems outdated and potentially vulnerable to
"script
kiddies" who've just downloaded the latest exploit. If software had
a higher
level of quality monitoring would be far less important, and patch
management would be far less of an issue, but as a many recent
surverys have
shown one of the biggest headaches for IT deparments at the moment
is
testing and deploying all of the patches for all of the software
they run.

The original point I was trying to make is that the IT community
should look
to take a harder stance on developers who allow shoddy code to be
released,
and not stop developing software just because it looks tricky. This
is
inline with the views of people commisioning buildings and the
archiects who
designed the buildings which failed under normal load (such as the
gerrards
cross rail bridge, paris airport, etc.), after all would you want to
hire
someone to build your house where the last house they designed
collapsed?

If a developer chooses a library they should use test cases to
proove it
operates safely under the conditions they would use it, and the
conditions
under which the library can be abused due to their program (i.e. if
the
developer isn't checking the length of a copy and destination buffer
then
they should check the library doesn't go wrong when the length of
the copy
exceeds the destination buffer). Picking the first library that
comes up on
google which offers the functionality a developer needs is like
choosing the
first plot of land you find on which to build your house, and if
architects

and builders did that then I'm sure the 2.7% figure would be a lot higher.

If we can improve the quality of software then hopefully one day architects
will look at IT and go, "Now if we designed things the way the IT guys
design their systems we'd have fewer problems....." ;).

Al.

−−−−−Original Message−−−−−
From: Craig Wright [mailto:cwright@xxxxxxxxxxxxx]
Sent: 23 February 2006 23:29
To: support@xxxxxxxxxxxxxxxxxx; dave kleiman; Darren W Miller
Cc: defendingthenet; security−basics@xxxxxxxxxxxxxxxxx
Subject: RE: Why Easy To Use Software Is Putting You At Risk

I am sorry − but I can not help responding to the point on architects;

>From Elsevier − "Engineering Failure Analysis", about 2.7% (95% CI) of
>homes suffer structural damage caused by soil subsidence within the
>first 5 years that should have been determined and countered in the
>design. If we look to the expected lifetime of 20 or 25 years for a
>home... Well things are worse.

Examples based on design failures follow (these are only the catastrophic
failures). Would you like more? I have the references for all the examples
below if you wish to read more than the headlines?

Is more solid proof required?

You have stated that you are a scientist, would you like me to provide an
ANOVA table for the above figures?

Regards,
Craig

PS − I may not always put every piece of data in a post, but I always have
it handy when I am writing the post. I am ALWAYS more than happy to

flood
anyone who requests it with the data.

See

http://www.elsevier.com/wps/find/journaldescription.cws_home/30190/descripti
on#description

Railway tunnel collapses at Gerrards Cross

A 20−metre section of a partially completed railway tunnel at
Gerrard Cross
in Buckinghamshire collapsed.

Roof Collapses at Paris Airport

A 120−foot section of a new terminal at the Charles de Gaulle
international
airport collapsed killing at least five people, injuring seven and
burying
an unknown number of others.

Girder collapse in Colorado

A 40−ton steel girder dropped from a freeway overpass construction
site into
morning traffic, crushing one car and killing all three people
inside.

Four Construction Workers Died after Crane Collapse in Toledo, Ohio

Three iron workers were killed and five injured Monday afternoon in
the
collapse of a crane on a construction site outside of Toledo, Ohio.

Crane Collaped in Stratford Bridge Project, Killing the Crane
Operator

A $96−million bridge replacement job in Stratford, Conn., two
barge−mounted
cranes collapsed, killing the crane operator.

Moscow Roof Collapse Kills 21, Hurts 106

The snow−covered glass roof of a Moscow water park collapsed
Saturday

evening onto hundreds of people, killing at least 21 people

A Partially Finished Bridge Collapsed in California, USA

An approximately 100−foot section of a partially finished bridge collapsed,
killing one worker and injuring seven others.

A Casino Garage in New Jersey, USA, Collapsed

The top five stories of a parking garage under construction at a casino
collapsed. Three people were killed.

Flooded Subway Project Causes Subsidence in Shanghai, China

An underwater tunnel connected with Shanghai's planned fourth subway line
has collapsed, causing several buildings to tilt and subside.

Rhode Island Nightclub Fire

A pyrotechnics display ignited the stage of a Rhode Island nightclub, which
caused the blaze to spread throughout the building. At least 98 people were
killed and 160 injured.

South Korean Subway Fire

A formal mental patient set fire to the packed subway train in Daegu, South
Korean, killing up to 200 people.

Chicago Club Fire

At least 21 people were killed at the Club when they panicked and tried to
escape a fight.

Building Collapsed in San Antonio

A five−story building collapsed in downtown San Antonio, 3 people injured.

A Schoolhouse Collapsed in An Earthquake in Italy

26 children were buried in the collapsed house while most of nearby buildings stand.

N.Y. pedestrian bridge collapse

A pedestrian bridge under construction collapsed as concrete was being
poured onto its steel girders, killing one worker and injuring 10 others.

Panels and roofing metal collapsed in Western Australia

A concrete "tilt−up" slab at a Western Australia construction site crushed,
killing a construction worker.

Miami bridge−tower collapses

The control tower on the Flagler Street bridge in Miami collapsed, injuring
a woman.

A Dam in Northern Syria Collapses

A dam in northern Syria collapsed, killing at least two people.

Apartment building in St. Petersburg collapses

A nine−story apartment building in St. Petersburg collapses, killing three
people.

Russian Cosmodrome Roof Collapses

Part of the roof of Russia's space launch complex in Kazakhstan has collapsed, injuring at least eight people.

Beirut Building Collapse Kills Four

A seven−story building collapsed into a pile of rubble Saturday,

killing
four people and crushing cars.

Falling Scaffolding in Chicago Killed Three People

Scaffolding from the 43rd floor of John Hancock Building fell to the
downtown street, killing three people.

Convention Center Girders Collapses in Pittsburgh

Steel girders collapsed at the David L. Lawrence Convention Center
under
construction, killing a Moon ironworker and injuring two others.

Scaffolding Collapsed at A Manhattan Office Building

Five construction workers were killed and 10 others were injured
when a
scaffolding collapsed at a Manhattan office building.

Wedding Hall Collapses in Jerusalem

An over−crowded wedding reception hall collapsed Thursday night in
Jerusalem, killing at least 25 people and injuring 250.

Steelwork Collapses at Convention Center Site

Part of the new D.C. convention center collapsed.

A Bridge Collapse in Portugal Kills up to 70 People

A 116−year−old bridge in Portugal collapsed. One of support pillars
gave way
under pressure from river water.

Selby rail disaster

Caused by a piece of metal from a Land Rover which had plunged onto
the
track falling onto the line, the accicident killed 13 people,
injured a
hundred.

Dulles Airport Tunnel Collapse

Part of a pedestrian tunnel under construction at Dulles International
Airport caved in trapping a worker in the rubble.

Construction Trench Collapsed in Texas, USA

A construction trench collapsed, killing three workers who were buried in 14
feet of dirt.

Hatfield Rail Crash

A high−speed train crash north of London that killed four people and injured
34 put the safety of Britain's railways in question on Wednesday.

Kansai International Airport

Six years after its completion, Japan's second−largest airport is sinking
into the ocean much faster than expected.

High School Gym in Cleveland, USA

The roof of a Cleveland, Ohio, high school gym collapsed, injuring three
students and two adults.

Building Collapse in India

Twenty−three people are reported to be killed in building collapse in
Tundla, India.

Moscow's Giant TV Tower Collapse

Completed in 1967, the Europe's Telecommunications towe's exposed prestressing cables inside are vulnerable to blaze.

SW China Bridge Collapse

A newly built pontoon bridge collapsed in Luzhou, a city in

Southwest
China's Sichuan Province, killing at least two people.

Wall Collapse on Construction Site, Maryland, USA

Two people were killed and three others were hurt when an eight inch thick
cinder−block wall collapsed at a construction site in suburban Baltimore.

Winery Terrace Collapse in Ohio, USA

A terrace loaded with tourists collapsed at an island winery in Lake Erie,
Ohio, USA

Overpass Collapse Shuts down Quebec Highway

A huge concrete beam fell on the vehicle as it was passing under the viaduct.

Millennium Bridge Sways

This newly completed bridge in London had to be closed because it swayed.

Speedway Bridge at North Carolina, USA

A concrete pedestrian walkway spanning a four−lane highway in front of the
speedway collapsed, injuring more than 100 people.

−−−−−Original Message−−−−−
From: Al Sutton [mailto:asutton@xxxxxxxxxxxxxxxxxx]

Sent: 24 February 2006 8:33
To: Craig Wright; 'dave kleiman'; 'Darren W Miller'
Cc: 'defendingthenet'

Subject: RE: Why Easy To Use Software Is Putting You At Risk

Hi,

I too am very open to being proven wrong, but as a scientist I need solid
proof which involves cold hard facts, not statements such as "I can't go
into all the details for various reasons.".

I've been involved in many development projects, and at the end of the day a
product ships with bugs from a library then it's the developer who is
responsible for their choice of libraries. The attitudes Darren describes
are typical in Development, the "If it ain't in my code it ain't my problem"
is one of the most fundamental problems of current development mentality.
How many architects do you know that would design for the side of a hill
without making sure the hill could support their design?, or design an
extension to a house without ensuring the house was sound?, the same is true
of code, if you're writing software you need to make sure your libraries
support it securely, if not, then you're not doing your job. Developers can
add verification code before they send code to libraries, and if they have
concerns of a library this is what they should be doing (after all why
rewrite a string copy routine when you just need to check that the length of
your copy is less than the length of your destination buffer?).

My view is that the original paper was FUD, intended or not, that's how it
appeared, that's how it read, and it it walks like a chicked and clucks like
a chicken people are going to call it a chicken.

Al.

−−−−−Original Message−−−−−
From: Craig Wright [mailto:cwright@xxxxxxxxxxxxx]
Sent: 23 February 2006 21:10
To: dave kleiman; Darren W Miller

Cc: Al Sutton; defendingthenet
Subject: RE: Why Easy To Use Software Is Putting You At Risk

Hello,

Dave stated; "Craig.. And be gentle Craig will pick apart opinions and bring
back factual information without batting an eye."

True and I am always open to being proved wrong. The thing is that I have to
be PROVED Wrong. Opinion and anecdotal evidence is not proof. Validated
points and correctly collected statistical data are.

As much as many people find this difficult to believe (even my wife) I enjoy
being proved wrong. It is both a learning opportunity for my self and a
demonstration that others are engaging in serious peer review processes
outside of academe.

In the past 20 years I have performed close to 5,000 engagements. At the
moment I am conducting one of the largest vulnerability and risk assessments
ever conducted in Australia in association with the Attorney Generals CNVA
programme.

The first issue to address is yes you found a vulnerability and it was
exploitable. What is the risk? The impact threat vectors and other analysis
factors need to be considered. Vulnerabilities do not matter by themselves.
They create a risk potential. When you understand this you will both serve
your clients more effectively and also add value in a manner they will
understand. You need to sell to management. They understand finance and
risk. Vulnerabilities are FUD. They do not help.

As for engineering something not to fail. This is where I have an issue with
people who think they are engineers. Engineering is the process of building

something to a set specification. An example is giving a 95% Confidence
Internal of a 5 year expected life. It involves the analysis and design of
hazard functions and survival processes.

Regards,
Craig

PS this is about as nice as I get unless people actually seek to open their
minds and learn.

−−−−−Original Message−−−−−
From: dave kleiman [mailto:dave@xxxxxxxxxxxxxxx]

Sent: 23 February 2006 4:25
To: 'Darren W Miller'
Cc: Craig Wright; 'Al Sutton'; 'defendingthenet'
Subject: RE: Why Easy To Use Software Is Putting You At Risk

Darren,

I am going to explain this to you, since you are new here on this forum, or
at least I have only saw one or two of your posts go by recently. I am not
the form moderator, nor do I have any influence over the posts that make the
forum.

First, I wanted to give you a friendly heads−up, because you are throwing
"articles" out to this forum and they are your opinion.

Secondly, I am a nice guy :), maybe you are taking this personally, but you
need to read through the archives, this s what we do here debate!!

""""I don't have the time to keep this discussion (if that I what we are
actually having) going for an infinite amount of time""" You posted this to
a Security Discussion board, that is what we do here.

Do not get me wrong you have the right to post almost anything you want
pertaining to security, but if throw your opinion out here, expect to have

to defend it, and back it by fact. Because it is going to get torn up by the professionals.

I have seen threads, that is what you started a thread, go for 20−30 days. See "Forensic/Cyber Crime Investigator" in the archives, it went from mid−Jan until Feb 15th, and I thought Craig was going to kill me on that one, but that is how this forum goes, you make a statement expect educated well−informed/experienced responses, a lot of them you will not agree with, but will not be able to tap dance away from.

Craig.. And be gentle Craig will pick apart opinions and bring back factual information without batting an eye. He and I have gone toe−to−toe on many a subject on this and other discussion forms.

Darren, I know you are used to posting articles at CastleCops were the home user is the basic audience and nobody is retorting, but when you step into this arena you will see some serious professionals in varying fields and they will not let misinformation slide. You of course do not have to respond to the responses, but expect even heavier discussion when you post and disappear.

By the way if you were to post this at a higher level forum such as pen−test, they would eat your below write−up for breakfast. But since you left it off post, I did the same....however I know Craig loves pen−testing so he may not.

Dave

−−−−−Original Message−−−−−
From: Darren W Miller [mailto:Darren.Miller@xxxxxxxxxxxxx]

Sent: Wednesday, February 22, 2006 20:06
To: Craig Wright; dave kleiman
Cc: Darren W Miller
Subject: RE: Why Easy To Use Software Is Putting You At Risk

Gentlemen,

I don't have the time to keep this discussion (if that I

what we are actually having) going for an infinite amount

of time. But let me give you a couple high−level examples

of what I am talking about here. The key word is

high−level, I can't go into all the details for various
reasons.

In the last 3 months I have performed 5 assessments. Phase

I of these assessments involved penetration testing of

external public facing systems. Out of the 5, we achieved

total systems penetration / compromise of 4. All 4 of

these systems were web based services. All 4 of these

systems were compromised by exploiting "custom" code or

modules. During post−assessment meetings the developers

(who were independents) were present. When they were shown

what modules were used to achieve the compromise everyone

one them blamed it on other external modules they used (or

re−usable code / modules,) and that they had no idea these

bugs existed. They further explained that some of the

source code, at least the ones they had access to, were so

extensive and complex that they probably would never had

found the bugs. One gentleman even stated that it was not up to him to make sure code developed by others is secure even if he is using that code. That did not go over well in the meeting, trust me

AS far as "engineering something not to fail", I don't even think that is possible at this point in time. Or ever will be. Quite frankly, if someone were to tell me that a particular system, any system, was fail−proof, I'd say that they were off the wall. Let me just include a couple bullet point items that may fall into this category of "complex systems" and security:

1) Compromise of internal network systems using citrix as an entry point. End users thought that the citrix remote desktop profiles were secure because of how they were setup but never realized that flaws in something as simple (or complex) as ms−word would allow an isolated compromise to lead to additional systems compromise.
2) System A interacts with System B which interacts with system C. End users are aware, to an extent, about the flaws in system A & B and their interaction, but not aware of much regarding system C. In fact, they were not even aware there was a system C. That interaction with system C resulted in a security breach. In this case, complex systems interacting with other complex systems, some of

which were unknowns, leading to security breaches.

3) IT department decides to increase the over all security of authentication methods so increase complexity rules and other related items such as aging.... However, they have poor auditing measures internally and have know idea that there are 150 user accounts for people who no longer work for the company. Even though authentication measures / procedures have been changed on the system, these particular accounts will not have them applied until the next time they are used. Several of these accounts are compromised because they don't meet even basic complexity rules for passwords. However, the end user thought that the system would take care of this and force all accounts to abide by the same rules immediately. Did not happen.

Here is the bottom line. Either I did a really poor job at trying to get my message across in a high−level way, or I am just being totally misunderstood. I would suggest it's a little of both based on this dialoged.

Note: One final point. I would rather you not make the statement that I am using FUD as a selling tool. The fact is that is not true and is not my intention. If either of you new me personally you would know that. I would never, and have never, made that kind of assumption without

knowing for sure. Quite frankly, I'm not sure I would make

that kind of statement about anyone, even if I knew for

sure that is what they were all about.

Regards,

Darren W. Miller

−−−−−Original Message−−−−−
From: Craig Wright [mailto:cwright@xxxxxxxxxxxxx]
Sent: Wednesday, February 22, 2006 5:41 PM
To: dave kleiman; security−basics@xxxxxxxxxxxxxxxxx
Cc: Darren W Miller; defendingthenet
Subject: RE: Why Easy To Use Software Is Putting You At Risk

Hello

Here I have to state that I agree 100% and categorically with
Dave.

FUD − Fear Uncertainty and Doubt is a common tool used by

vendors to sell security. It is also one of the greatest

threats to security today.

It makes people inured to security in the long run (i.e.

cry wolf) and in the short term results in a lot of

technical solutions that generally fail to address the issue.

NASA uses hazard and survivability models to determine

risk. They do not engineer to not fail − they just reduce

the probability of an incident. What needs to be

remembered that is that 1 in a million occurrence happens

all the time in the real world. Even a 1 in a billion

occurrence will happen daily somewhere in the world.

Welcome to the world of risk.

So as to the original post, how would complex software

make you less risk prone?

Regards,
Craig

−−−−−Original Message−−−−−
From: dave kleiman [mailto:dave@xxxxxxxxxxxxxxx]

Sent: 23 February 2006 2:23
To: security−basics@xxxxxxxxxxxxxxxxx
Cc: Darren.Miller@xxxxxxxxxxxxxxxxxxx; 'defendingthenet'
Subject: RE: Why Easy To Use Software Is Putting You At Risk

Inline....

−−−−−Original Message−−−−−
From: defendingthenet [mailto:mlapidus@xxxxxxxx]
Sent: 20 February 2006 14:35
To: security−basics@xxxxxxxxxxxxxxxxx
Subject: Why Easy To Use Software Is Putting You At Risk

Title
−−−−−
Why Easy To Use Software Is Putting You At Risk

Can Easy To Use Software Also Be Secure
−−−−−−−−−−−−−−−−−−−−−−−−−−
Anyone who has been working with computers for a long time

will have noticed
that mainstream operating systems and applications have

become easier to use
over the years (supposedly). Tasks that use to be complex

procedures and
required experienced professional to do can now be done at

the push of a
button. For instance, setting up an Active Directory

domain in Windows 2000
or higher can now be done by a wizard leading even the

most novice technical
person to believe they can "securely" setup the operating

environment.

Where does it claim that it is "securely" setting up AD in

the wizard?

This
is actually quite far from the truth. Half the time this

procedure fails
because DNS does not configure properly or security

permissions are relaxed
because the end user cannot perform a specific function.

Sounds like you have had this problem a few times, maybe

you should not use the wizard, or attempt AD setups.

Do you understand how to "securely" setup AD, for your

comments here, I would say no.

Instead of using the "sky is falling routine" suggest how

to do these things securely instead of syaing "look how

terrible this is"

If It's Easy To Develop, Is It Also Secure
−−−

Are you referring to shared code? In case you do not know

what that is, it is code that is shared by apps for the

same routines.

For instance, back in the old days when I started as a

developer using assembly
language and c/c++, I had to write pretty much all the

code myself.

Are you suggesting your code was more secure back in the

"old" days, when security was not a concern in coding?

Now everything is visually driven, with millions of lines
of

code already
written for you. All you have to do is create the

framework for your
application and the development environment and compiler

adds all the other
complex stuff for you. Who wrote this other code? How can

you be sure it is
secure. Basically, you have no idea and there is no easy

way to answer this
question.

Secure Environments Don't Exist Well With Complexity
−−−−−−−−−−−−−−−−−−−−−−−−−−−
The reality is it may look easier on the surface but the

complexity of the
backend software can be incredible. And guess what, secure

environments do
not coexist well with complexity. This is one of the

reasons there are so
many opportunities for hackers, viruses, and malware

to attack your
computers. How many bugs are in the Microsoft Operating

System? I can almost

guarantee that no one really knows for sure, not even

Microsoft developers.
However, I can tell you that there are thousands, if not

hundreds of
thousands of bugs, holes, and security weaknesses in

mainstream systems and
applications just waiting to be uncovered and maliciously

exploited.

How Reliable and Secure are Complex Systems?
−−−

protect failsafe
systems, and even more failsafe systems to protect other

systems. Systems
like the space shuttle need to perform consistently, cost

effectively, and
have high Mean−Time−Between−Failure(MTBF).

All in all the space shuttle has a good record.

One thing

it is not though
is cost effective and consistent. Every time there is a

launch different
issues crop up that cause delays. In a few circumstances,

even the most
basic components of this complex system, like "O" rings,

have sadly resulted
in a fatal outcome. Why are things like this missed? Are

they just not on
the radar screen because all the other complexities of the

system demand so
much attention? There are million different variables I'm

sure. The fact is,
NASA scientists know they need to work on developing less

complex systems to
achieve their objectives.

Ok now you have stepped out of bounds, first of all I love

NASA and have the utmost respect for them and all the

astronauts who have braved the frontier.
However, the record of the shuttle is 110+ scrubbed

launches. That is more than the number of launches. You

can do the math for the rest, but it does not add up to a

good record, you might have to use one of those "complex

systems" though to run calc.

So your saying a more simplistic system would create a better record, maybe they should try fly the Kitty Hawk to the moon.

I am just going to stop here and say Hogwash.

My advice to you is stop selling fear and your opinion, and start selling solutions to problems. Next time tell us how to fix your proposed problems.

Respectfully,

_____

Dave Kleiman, CAS,CCE,CIFI,CISM,CISSP,ISSAP,ISSMP,MCSE

www.SecurityBreachResponse.com

This same principal of reducing complexity to

increase security,
performance, and decrease failures really does apply to

the world of
computers and networking. Ever time I here associates of

mine talk about
incredibly complex systems they design for clients and how

hard they were to
implement I cringe. How in the world are people suppose to

cost effectively
and reliably manage such things. In some cases it's almost

impossible. Just
ask any organization how many versions or different brands

of intrusion
detection systems they have been through. As them how many

times the have

had infections by virus and malware because of poorly

developed software or
applications. Or, if they have ever had a breach in

security because the
developer of a specific system was driven by ease of use

and inadvertently
put in place a piece of helpful code that was also helpful

to a hacker.

Can I Write A Document Without A Potential Security

Problem Please
−−−

databases, and execute
web based functions.

Do you know what I use it for, to write documents. nothing

crazy or complex,
at least most of the time. Wouldn't it be interesting that

when you first
installed or configured Microsoft Word, there was an

option for installing
only a bare bones version of the core product. I mean,

really stripped down
so there was not much to it. You can do this to a degree,

but all the shared
application components are still there. Almost every

computer I have
compromised during security assessments has had MS−Word

installed on it. I
can't tell you how many times I have used this

applications ability to do
all kinds of complex tasks to compromise the system and

other systems
further. We'll leave the details of this for another

article though.

Conclusion
−−−−−−−−−−
Here's the bottom line. The more complex systems get,

typically in the name
of ease of use for end users, the more opportunity for

failure, compromise,
and infection increases. There are ways of making things

easy to use,
perform well, and provide a wide variety of function and

still decrease
complexity and maintain security. It just takes a little

longer to develop
and more thought of security. You might think that a large

part of the blame

for complex insecure software should fall on the

shoulders of the
developers. But the reality is it is us, the end users and

consumers that
are partially to blame. We want software that is bigger,

faster, can do just
about everything, and we want it fast. We don't have time

to wait for it to
be developed in a secure manner, do we?

You may reprint or publish this article free of charge as

long as the
bylines are included.

Original URL (The Web version of the article)
−−−−−−−−−−−−

http://www.defendingthenet.com/NewsLetters/WhyEasyToUseSoft
wareIsPuttingYouA
tRisk.htm

About The Author
−−−−−−−−−−−−−−−−
Darren Miller is an Information Security Consultant with

over seventeen
years experience. He has written many technology &

security articles, some
of which have been published in nationally circulated

magazines &
periodicals. If you would like to contact Darren you can

e−mail him at
Darren.Miller@xxxxxxxxxxxxxxxxxxxx If you would like to

know more about
computer security please visit us at

http://www.defendingthenet.com.

−−−

recipient, you must not use or disclose the information.

If you have received this email in error, please inform us

promptly by reply email or by telephoning +61 2 9286 5555.

Please delete the email and destroy any printed copy.

Any views expressed in this message are those of the

individual sender. You may not rely on this message as

advice unless it has been electronically signed by a

Partner of BDO or it is subsequently confirmed by letter

or fax signed by a
DISCLAIMER
The information contained in this email and any attachments is confidential. If you are not the intended recipient, you must not use or disclose the information. If you have received this email in error, please inform us promptly by reply email or by telephoning +61 2 9286 5555. Please delete the email and destroy any printed copy.

Any views expressed in this message are those of the individual sender. You may not rely on this message as advice unless it has been electronically signed by a Partner of BDO or it is subsequently confirmed by letter or fax signed by a Partner of BDO.

BDO accepts no liability for any damage caused by this email or its attachments due to viruses, interference, interception, corruption or unauthorised access.