NEOHAPSIS - Peace of Mind Through Integrity and Insight

Messages sorted by: [date] [thread] [subject] [author]

> -----Original Message-----
> From: Windows NTBugtraq Mailing List
> [mailto:NTBUGTRAQ@LISTSERV.NTBUGTRAQ.COM]On Behalf Of Russ
> Sent: Friday, November 02, 2001 7:14 PM
> To: NTBUGTRAQ@LISTSERV.NTBUGTRAQ.COM
> Subject: Re: Call to arms - INFORMATION ANARCHY - pause
>
>
> Pause, think, come back Sunday. No more messages on this
> subject until then.

OK, so please post this on Sunday. I haven't read the whole thread - there's
some thoughtful comments in there, esp. Garry McGonigal and Arne Vidstrom.
There's also a fair dose of paranoia and speculation when the writers clearly don't have the facts.

I'm writing this speaking from my own perspective, and this is not to be construed as a statement on behalf of Microsoft. I'm also writing this as someone who has found and reported a large number of security bugs in different vendor's products. My perspective also includes being part of the
security response process for a very large network (Microsoft), so I well understand the plight of a security admin. I also have written and continue to write network security auditing tools.

I work closely with the vulnerability response process at Microsoft, so I
have first-hand knowledge of intent. I'll speak to that first. The intent is
to develop more secure products. I work hard every day to improve
product
design, coding practices, usability of security features, and get problems
fixed. Regardless of whether the discoverer or reporter of a problem
wants
their name in lights at the bottom of an 'advisory', we want to get security
issues fixed. Period. I want to get them fixed as quickly as possible, and I
know there's a lot of people that work hard with me towards that goal. I
can
also see clear progress - Win2k is a big improvement over NT 4.0 and
XP/.NET
has many improvements over Win2k. I want to continue that trend. This
isn't
marketing fluff - it's my job, I take it seriously and work hard at it. I've
also worked with Michael Howard to write a book on improving secure
coding
practices - "Writing Secure Code" should be back from the publisher in
the
next week or so. I hope this will help developers avoid creating new
security bugs.

I also have to point out that anyone who makes a statement about "the
vendors" is being hopelessly overly general. I have worked with vendors
who
simply fixed things whether they were threatened or not. I have worked
with
other vendors who strung me along for months and then refused to fix the
problem. I've seen other vendors who were simply ignorant and told me
nonsense like "buffer overruns aren't exploitable on Win32" - yeah, right.
Anyone who treats a diverse group as being uniform isn't thinking through
the problem very thoroughly. This statement applies to vendors,

vulnerability reporters and hackers - each group displays a wide range of behaviors.

For example, "hellnbak" says:

"... the vendors have no motivation to fix their code, or improve
on their programming methodologies without full disclosure. Vendors have
proved this in the past so there is no reason to believe that things will
be different."

This is an example of a simplistic approach and flawed logic. Some vendors
might not be motivated, others might. To believe that vendors all behave the
same way is flawed. To believe that behaviors don't change over time is to
dismiss people's ability to learn from both their own mistakes and others.
For example, I proved in the past that I was stupid enough to drive around
with no seatbelt. After flipping my VW Bus over end to end, I don't go out
of my driveway without one. People and groups of people clearly change
behaviors over time in many instances.

So a vendor who won't fix bugs unless their customers are threatened with
active attack is a very different problem than one who fixes problems when
they are reported. If you turn the clock back several years, there were more
vendors in the first category than the second, and faced with that past
reality, Full Disclosure (tm) was a reasonable response. I will argue that
it may not be a reasonable response for all vendors today.

As a short digression, I think that "Full Disclosure" has become a bit of a
religeous term, along with the requisite true believers and heritics. You're

either one of us, or you're one of them. And those who we call them are always on the wrong side of everything. IMHO, this tends to discourage rational thought. This isn't to say that all proponents of Full Disclosure are irrational, just that we ought to think about ways to make the whole process work better. As security people, we're supposed to think outside the
box, so I find it curious that we set about constructing our own boxes. <g> People sometimes construct some very interesting reality tunnels.

Now let's rationally analyze what sorts of vulnerabilities lead to widespread attacks. I've seen a lot of vulnerabilities go by, some big, some
small, and not all of them lead to widespread attacks. It's bad enough to have a problem in a system you're trying to protect without a bazillion monkeys (er, script kiddiez) all trying to hack you at once. Now, like any complex behavior, we have to deal with statistical trends - there's always going to be exceptions. Would it be possible to find security problems, get a fix created, get them applied, and do this without being subject to widespread attacks? I think so, because this happens sometimes. Can we possibly find ways to behave so that this happens more frequently? I think so.

For example, one day at ISS, I came in to find every NT system on our network blue screened. Something had gone horribly wrong both in NT and our
UNIX scanner. Within a couple of days, Microsoft gave us a private patch to
test, it worked, and shortly thereafter they shipped a hotfix (post SP-2 NT 4.0, I think). There was no advisory, no arm-twisting, and no public attacks. It would be hard to find a system today that would be vulnerable. It would be harder still to find the exploit. This is an example of the process working right.

Now let's consider what widespread attacks have in common - there is
typically a user-friendly attack created by someone. It is often preceded
by
published "proof-of-concept" code. It is sometimes preceded by an
attack
created in the black hat community that's leaked out. While discussing
this
with Mudge, he pointed out to me that black hat attacks leaking seem to
be
more common with UNIX exploits - neither of us are sure why. I would
assert
that creating and distributing a user-friendly attack is irresponsible, as
it will most often lead to widespread attacks.

Let's think about how to set about actually finding vulnerabilities in
systems. I've written several hundred checks for vulnerabilities.
Sometimes,
you either exploit the vulnerability or not - e.g., if you can log in via
telnet as root:root, that's the only way you know if root has a stupid
password. In many other cases, a published script isn't very useful. Proof
of concept code is often extremely poorly written and unreliable. It
frequently has very undesirable side-effects, like crashing the system or
leaving it more vulnerable than it was in the first place. Now, if you're a
malicious hacker, that's just fine fun - but if you're a security admin
responsible for an operational network, it isn't much use. In some, but not
all, cases the information needed to write a solid network auditing check is
different than the information you get from the actual exploit. Part of the
motive for releasing proof of concept code is to allow people to test for
themselves whether a system is vulnerable - and we security admins
don't
always have access to check the file version. That test can often be
conducted without actually firing the exploit. If that's possible, then it
is best to tell people how to tell the difference between a system that

needs a fix and one that is fixed without providing something that
facilitates attacks.

Something else to consider is that from the vendor's POV, if something
was
reported externally, they can't be sure it hasn't leaked into the black hat
community. I've seen this happen more than once. You also can't be sure
that
someone else hasn't discovered it independently, or even that an exploit
has
quietly existed in the black hat community for some time and is just now
coming to light. So, if you're going to treat a security report responsibly,
you have to assume the worst - your customers could be under attack
now, and
might be under widespread attack at any time. I have to take this very
seriously, because some customers might be performing important
functions
like catching terrorists.

I also have experience as a development manager, and I can tell you for
sure
that the more pressure there is to get a fix out quickly, the less likely
the fix is going to be thorough and the more likely it is to create new
bugs. This is true of _any_ bug, not just security bugs. It's a good thing
to give people enough time to fix things thoroughly. That's not to say that
anyone should be allowed to just delay without cause - there's a balance
to
be struck here. A responsible vendor responds promptly, and responsible
reporter gives a vendor enough time to be thorough - you may have just
uncovered the tip of the iceberg, and the quick fix to your problem might
be
simple. The fix to the bigger problem might be complex. I'd rather see one
fix that is comprehensive than a series of quick fixes for the same thing.

Chris Wysopal (Weld Pond) has been doing these same sorts of things even
longer than I have, and he's reached some similar conclusions. Think about
how we can encourage vendors to make better products, get fixes created and
applied, and do so without encouraging what amounts to network terrorism.
That's the goal. Let's try and find ways to get there.

David LeBlanc
dleblanc@mindspring.com

- **Messages sorted by:** [date] [thread] [subject] [author]