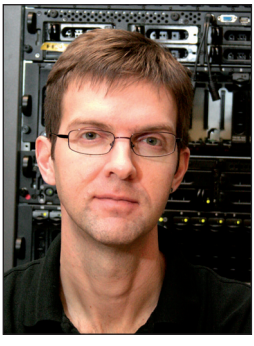
OPINION

STUX IN A RUT: WHY STUXNET IS BORING

John Aycock
University of Calgary, Canada



The much-storied Stuxnet worm is unworthy of the hype surrounding it. The biggest surprise is that Stuxnet contains no surprises, and as such it suggests a general failure of security to respond to threats that are well known. The erroneous characterization of Stuxnet as 'game-changing' does raise other questions, however: what are the hallmarks of real game-changing security events, and why don't we see more of them?

INTRODUCTION

In case you've somehow managed to avoid hearing about Stuxnet, here are the essentials: Stuxnet is a computer worm that has spread and infected machines primarily in Iran. It seems to have targeted uranium enrichment facilities, the output of which can be used in nuclear reactors as well as nuclear weapons; the latter is naturally of some concern to the governments of countries that don't see eye-to-eye with Mahmoud Ahmadinejad. The source of Stuxnet, meanwhile, has been conclusively narrowed down to Planet Earth.

We are accustomed to media reports about malware containing hyperbole and gushing superlatives, but perhaps the biggest thing to note about Stuxnet is that security professionals – not the media – have been describing it using terms usually reserved for reviewing Broadway productions: 'a game-changer', 'a watershed moment', 'I've never seen anything like it'. Stuxnet must be impressive indeed.

There is always one curmudgeonly, contrary Broadway critic, however. That would be me. My view is that Stuxnet is really not that interesting at all, and is at best yet another unfortunate illustration of the fact that the security emperor has no clothes. Let me explain why.

CHANGING GAMES AND SHEDDING WATER

As in history, key events and developments in computer security are best judged in the fullness of time. It is only

then that we have the luxury of hindsight and a surer understanding of the ramifications of a particular event.

If history repeats itself, however, then we can learn some general lessons that can be applied to the present, when hindsight is not an option. That would seem to suggest that there might be some general lessons we can learn about computer security, too. Can we identify a game-changing, watershed event when it occurs?

I propose the following tests. A security threat that meets at least one of the following criteria may be well on its way to becoming a watershed event:

1. Do defences have to be changed in a substantial way to respond to the threat?
2. Does the threat constitute a major shift in motivation for the adversary?
3. Is the adversary using a new business model?

The first test is the most important indicator of a noteworthy threat. Being unable to effectively respond to a threat without a substantive change in defences is a clear sign that something significant has occurred. The latter two tests can be seen as precursors: a change in motivation or modus operandi likely indicates that users are to be targeted in some new ways, that will eventually demand new defences.

Applying these tests, the first game-changer was the computer virus. Not the first known virus in the wild in 1969 [1, 2], not *Apple II* viruses in 1982 [3, 4], and not Fred Cohen's research in 1983 [5], though. The game-changer was the growing glut of PC viruses in the late 1980s – including Brain, Stoned and Jerusalem – that necessitated the development of anti-virus software, a substantial change in defence. The virus is still the embodiment of malicious software to the media and general public.

The next game-changing threat was polymorphism in the early 1990s. Simply put, a polymorphic virus changes its appearance to one of millions or billions of new forms on each new infection. Looking for static signatures is no longer sufficient for detecting polymorphic viruses because it is infeasible to enumerate all possible ways the virus may manifest itself. This led to another major shift in defence, anti-virus emulation, where suspect code is coaxed to run in a safe environment, in the hopes that it reveals any malevolent intentions or (ideally) a unique signature. This nontrivial change in defences involved a 'painful rearchitecting' of anti-virus software [6, p.264].

Moving forward to the mid-1990s, we come to macro viruses. While a proof-of-concept macro virus existed in 1989 [7], they went mainstream in 1995 with the release of Concept – which, ironically, was inadvertently shipped by *Microsoft* on a *Windows 95* compatibility test CD-ROM [8]. A macro virus is a virus written in a macro language, a programming language whose code can be embedded, in this case, in *Microsoft Word* documents. Macro viruses changed the security game because defences could no longer focus solely on executable files. Even data files could now be a threat, and defences had to adjust accordingly.

The second test, a major shift in motivation, leads to the next game-changer: money. Money made by stealing people's data, to be precise. For reference, phishing started in the mid-1990s, and spyware started ramping up through the 2000s. The coming together of four factors set the stage for this shift in motivation. To start with, there had to be a lot of computers connected to the Internet. Then, online banking and online commerce services needed to appear, followed by people using them (simply building a service doesn't guarantee that people will use it – just ask any dotcom startup). Finally, the adversary needed to realize that there was money to be made. Users were no longer just bystanders, owners of the computers on which viruses and worms happened to be spreading. Users and their data became a target; this led to defences such as anti-phishing toolbars and anti-spyware programs.

The final game-changing example is the botnets that started appearing in the early 2000s, compromising computers connected via the Internet that can be controlled by an adversary from afar. The necessary condition was the appearance of a large pool of vulnerable, always-on, always-connected computers. These computers have been repurposed by adversaries, unbeknownst to their owners, for stealing information, sending spam and conducting distributed denials of service. Effective defences now have to look beyond a single computer, beyond a single network, and beyond a single country. Maliciousness scales.

An important observation is that none of the examples above are singular. It is not one virus or one worm or one Trojan horse that is noteworthy by itself. In computer science terms, this makes perfect sense. One thing is a special case, and can be dealt with as a special case; a group of like things, on the other hand, demands a general solution.

HONOURABLE MENTIONS

Some threats looked promising as game-changers, yet didn't quite make the cut.

The Internet worm, aka the Morris worm, is an obvious contender and perhaps the closest parallel to Stuxnet. Released in 1988 by Robert Morris, Jr. (now a faculty

member at MIT), the worm pounded the prehistoric Internet. It was a singular instance of malicious software, and not a watershed moment according to the above criteria. If anything, the worm was an indictment of programming and system administration practices, but these poor practices were not news.

In *Hamlet*, Shakespeare wrote 'the lady doth protest too much,' and it is interesting to note that the tenor of some worm analyses was not only technical, but at times laced with disdain and derision; Spafford belittling the worm author's programming skills comes to mind [9]. Certainly the worm did cause disruption, but the security community was caught with its proverbial pants down, and it's not inconceivable that some psychological projection was in play. The Cornell Commission investigating the worm found that 'At least one of the security flaws exploited by the worm was previously known by a number of individuals, as was the methodology exploited by other flaws' [10, p.707]. In fact, the buffer overflow technique used by the worm was known as far back as the Anderson report in 1972, 16 years previously [11]. The Internet worm was not a turning point, it was an embarrassment.

The causal chain of other potential game-changing events is too long to claim any real impact. The inclusion of a TCP/IP stack in *Windows 95*, for example, went a long way towards creating a large pool of vulnerable machines. It was hardly a security threat in itself, though. And where is the line drawn – could the finger not be pointed as easily at the development of the Arpanet? It's a slippery slope.

Some would-be security game-changers are simply too early in their lifecycle. Cellular (smart) phones and social networking have undeniably transformed our lives, yet there are no radical, widespread new security threats from them. There are threats, yes, but for the time being they are old threats, repackaged for new platforms. Unfortunately, this is bound to change. The smartphone as an e-wallet, for example, dangles a tantalizing target for adversaries.

HOW STUXNET STACKS UP

Stuxnet doesn't fare well as a game-changer when reason, rather than rhetoric, is used. There are eight key features of Stuxnet (see the analysis in [12]):

- Stuxnet is large and complicated.

  If being a large and complex piece of software was a valid criterion, then every release of *Microsoft Office* would be a game-changer.

- Stuxnet is a targeted attack.

  Targeted attacks have been a concern for years – long, long before Stuxnet appeared.

- Stuxnet spreads in multiple ways.

  Again, Stuxnet is unoriginal, as a web search for 'multipartite virus' will attest.

- Stuxnet targets industrial control systems.

  The poor security of industrial systems comes as no surprise, and in fact Stuxnet isn't the first example of an attack against them (the 2000 sewage incident in Australia is a good example [13]).

- Stuxnet uses multiple exploits, some of which are zero-day.

  Stuxnet isn't the first threat to use zero-day exploits, and having several of them deserves some kudos but does not require any different defence.

- Stuxnet contains rootkits to hide itself.

  While having two rootkits (one for *Windows*, one for the programmable logic controller) is an interesting idea, rootkits themselves aren't new.

- Stuxnet misused code-signing certificates.

  This would be most distressing if certificates hadn't been abused or wrongly issued before (they have), and if signed code really provided the safety and security guarantees that people want (it doesn't).

- Stuxnet's motivation was espionage and/or sabotage.

  This is not a new motivation for malicious software. I would argue that Stuxnet could be considered a failure in some sense, because it was discovered (with the possible exception of Bond-esque antics, covert operations ideally remain surreptitious).

While Stuxnet is impressive in terms of the effort and investment it took to create, it is just not the dawning of a new age. There is nothing new to see here, there is nothing that security professionals haven't seen before. Especially telling is that fact that so much of the Stuxnet hullabaloo is focused on its analysis, ignoring the fact that, once its existence was known, anti-virus products could pick it off without difficulty. Or, as is phrased in *Symantec*'s threat assessment of Stuxnet, 'Removal: Easy' [14].

MALICIOUS MONOTONY

As a long-time observer of malicious software trends, the lack of game-changers is disappointing, in a perverse way. It is also understandable: the adversary is now typically a

businessperson, whose goal is to make money. If this goal is being met, i.e. enough malicious software is making it past anti-malware defences, then that's sufficient.

One example is the rapid repacking of malicious software. A packer is a relatively small investment, and can be bought from a third party then used to evade defences by overwhelming them with 'new' (or at least new-looking) samples of what is actually the same malicious software.

In contrast, Stuxnet is a bad investment. In general, there is no business case for one-shot, complicated, novel malicious software. The only exception is where a widely deployed defence is too good. Then, and only then, does the business-oriented adversary have the incentive to innovate. This back-and-forth was seen clearly with early spam and anti-spam, and arguably the success of early anti-virus was an impetus for polymorphic code.

This implies that security is not only risk management. 'Good' defences are the ones that keep adversaries in a sweet spot, where the adversary succeeds enough to be satisfied but doesn't fail enough to evolve. It's a strange notion, that losing the security game once in a while might be necessary to strike a healthy balance overall.

It could also be argued that malware like Stuxnet and the Morris worm has an educational value. Thanks to them, the issues are now in the public eye; they are prominent examples that can be used to justify funding for security, designing security into systems and additional security measures. But the sword slashing into popular consciousness is double-edged. Overwhelming evidence from the last decade suggests that hysterics over large-scale security events may lead to an unnatural obsession with the last attack, rather than promoting activity of any real benefit.

The education argument is fair, however, in that a game-changer must be considered relative to a particular audience. To the public, Stuxnet is a game-changer, without a doubt. To the security professionals charged with safeguarding everything from smartphones to critical infrastructure, Stuxnet and its successors should have been imagined long ago. Stuxnet should be little more than validation of security professionals' fears, not a surprise.

ACKNOWLEDGEMENTS

An early version of this paper was presented at the University of Glasgow as a SICSA Distinguished Visitor Talk. The author's work is supported in part by the Natural Sciences and Engineering Research Council of Canada. Thanks to Darcy Grant, Mike Locasto and Tim Storer for their insightful comments. The names of the security professionals mentioned in the introduction have been purposely omitted.

REFERENCES

[1] Benford, G. Worlds Vast and Various. EOS, 2000.

[2] Benford, G. Catch me if you can. Commun. ACM, 54(3):112-111, 2011.

[3] Dellinger, J. Re: Prize for most useful computer virus. Risks Digest, 12(30), 1991.

[4] Skrenta, R. Elk cloner. http://www.skrenta.com/cloner.

[5] Cohen, F. Computer viruses: Theory and experiments. Computers & Security, 6(1):22-35, 1987.

[6] Ször, P. The Art of Computer Virus Research and Defense. Addison-Wesley, 2005.

[7] Highland, H. J. A macro virus. Computers & Security, 8(3):178-188, 1989.

[8] WinWord.Concept. Virus Bulletin, October 1995, p.3. http://www.virusbtn.com/pdf/magazine/1995/199510.pdf.

[9] Spafford, E. H. The Internet worm program: An analysis. Technical Report CSD-TR-823, Purdue University, Department of Computer Sciences, 1988.

[10] Eisenberg, T.; Gries, D.; Hartmanis, J.; Holcomb, D.; Lynn, M. S.; Santoro, T. The Cornell commission: On Morris and the worm. Commun. ACM, 32(6):706-709, 1989.

[11] Anderson, J. P. Computer security technology planning study: Volume II, Oct. 1972. ESD-TR-73-51, Vol. II.

[12] Falliere, N.; O'Murchu, L.; Chien, E. W32. Stuxnet dossier (version 1.4). Symantec, February 2011. http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf.

[13] Slay, J.; Miller, M. Lessons learned from the Maroochy Water breach. In Goetz and Shenoi, eds, Critical Infrastructure Protection, pp.73–82. Springer, 2008.

[14] Shearer, J. W32.Stuxnet. Symantec Security Response, 2010. http://www.symantec.com/security_response/writeup.jsp?docid=2010-071400-3123-99.