Stuxnet is embarrassing, not amazing

As the New York Times posts yet another breathless story about Stuxnet, I'm surprised that no one has pointed out its obvious deficiencies. Everyone seems to be hyperventilating about its purported target (control systems, ostensibly for nuclear material production) and not the actual malware itself.

There's a good reason for this. Rather than being proud of its stealth and targeting, the authors should be embarrassed at their amateur approach to hiding the payload. I really hope it wasn't written by the USA because I'd like to think our elite cyberweapon developers at least know what Bulgarian teenagers did back in the early 90's.

First, there appears to be no special obfuscation. Sure, there are your standard routines for hiding from AV tools, XOR masking, and installing a rootkit. But Stuxnet does no better at this than any other malware discovered last year. It does not use virtual machine-based obfuscation, novel techniques for anti-debugging, or anything else to make it different from the hundreds of malware samples found every day.

Second, the Stuxnet developers seem to be unaware of more advanced techniques for hiding their target. They use simple "if/then" range checks to identify Step 7 systems and their peripheral controllers. If this was some high-level government operation, I would hope they would know to use things like hash-and-decrypt or homomorphic encryption to hide the controller configuration the code is targeting and its exact behavior once it did infect those systems.

Core Labs published a piracy protection scheme including "secure triggers", which are code that only can be executed given a particular configuration in the environment. One such approach is to encrypt your payload with a key that can only be derived on systems that have a particular configuration. Typically, you'd concatenate all the desired input parameters and hash them to derive the key for encrypting your payload. Then, you'd do the same thing on every system the code runs on. If any of the parameters is off, even by one, the resulting key is useless and the code cannot be decrypted and executed.

This is secure except against a chosen-plaintext attack. In such an attack, the analyst can repeatedly run the payload on every possible combination of inputs, halting once the right configuration is found to trigger the payload. However, if enough inputs are combined and their ranges are not too limited, you can make such a brute-force attack infeasible. If this was the case, malware analysts could only say "here's a worm that propagates to various systems, and we have not yet found out how to unlock its payload."

Stuxnet doesn't use any of these advanced features. Either the authors did not care if their payload was discovered by the general public, they weren't aware of these techniques, or they had other limitations, such as time. The longer they remained undetected, the more systems that could be attacked and the longer Stuxnet could continue evolving as a deployment platform for follow-on worms. So disregard for detection seems unlikely.

We're left with the authors being run-of-the-mill or in a hurry. If the former, then it was likely this code was produced by a "Team B". Such a group would be second-tier in their country, perhaps a military agency as opposed to NSA (or the equivalent in other countries). It could be a contractor or loosely-organized group of hackers.

However, I think the final explanation is most likely. Whoever developed the code was probably in a hurry and decided using more advanced hiding techniques wasn't worth the development/testing cost. For future efforts, I'd like to suggest the authors invest in a few copies of Christian Collberg's book. It's excellent and could have bought them a few more months of obscurity.