When Is a Product a Security Product?

I often find myself in mild arguments over whether this or that product is a security product. The competing alternative explanations are generally just spin: is such-and-such a service best called “business process outsourcing” or a “security service?” Is such-and-such a tool most evocatively described as an “intellectual property management toolkit,” or is it instead an “access control system?” The commercial abuse of language continues apace, but abuse of the language isn’t a good subject matter for Clear Text.

However, the question of “What is a security product?” does have meat on its bones. I suggest a working definition: a product is a security product when it has sentient opponents.

Let’s parse that definition in its contrapositive: if a product doesn’t have sentient opponents, then it isn’t a security product. This is best examined by looking at why products fail—if your product fails because of clueless users (“Hey, watch this!”), alpha particles, or discharged batteries, then it’s not a security product. If your product fails because some gleeful clown discovers that he can be a superuser by typing 5,000 lowercase a’s into a prompt, then said clown might not be all that sentient, but nevertheless, yours is a security product.

In other words, intent matters (just as the law would agree that intent can matter more than effect). In this case, if your product can, will, or does fail because someone can, will, or did try to make it fail, then even if you can’t bring yourself to agree that what you have is a security product, you at least have to agree that building your product as if it were a security product is something you must do.

This can’t be a completely bright line, but it is an instructive distinction. Security products are, almost by definition, designed with failure in mind—designed to resist failure even when that is what opponents devoutly wish for it, and designed for the failure case as much as or more than the success case. Security products envision opponents who can think and for the simple reason that they will.

Few important ideas are truly new, and this idea is no exception; as Bruce Schneier wrote in his foreword to Ross Anderson’s book, Security Engineering: A Guide to Building Dependable Distributed Systems (Wiley, 2001),

“Security involves making sure things work, not in the presence of random faults, but in the face of an intelligent and malicious adversary trying to ensure that things fail in the worst possible way at the worst possible time... again and again.”

When a sentient being attacks a product, we say such things as, “He used the 2-345y exploit.” In the now commonplace term exploit, we embody the idea of intent, the idea of starting with a small hole in some software that, through much effort, the attacker makes bigger and squeezes through. Put differently, only security products attract exploits in the same sense that only warm-blooded animals attract mosquitoes.

But then, perhaps we are yet again imagining that the digital world is more dissimilar to the physical world than it really is. Perhaps the better way to express this is to say that when the opponent you face will simply try again, perhaps harder now if his first approach is repulsed, then designing for failure should be the core of your concern. A street light drops a cover and misses hitting you, but the next street light won’t cough up its whole light arm to hit you; no security issue. You whack a dog with a stick, and it wants your arm even more; you’ve got a security issue. Trite as it sounds, that is the point about digital security—you can’t just take the easiest way out, or think along the lines of, “No one would ever do that,” or, worse still, “No one cares about our data enough to steal it.” And that, too, is not a new idea: as the US Army Ranger Handbook says,

“Two of the gravest general dangers to survival are the desire for comfort and a passive outlook.”

Design for the failure your sentient opponents hope you to have, and deny them their pleasure.

Daniel E. Geer Jr. is chief scientist for Verdasys and, like all chief scientists, thus does not do any real work. When he thinks of security, he thinks like Tallulah Bankhead: “Dahtling, do you know what makes cigarettes so wonderful? They never satisfy.” Contact him at dan@geer.org.