

On Shifting “Windows” and “Security” from Less Antonymous to More Synonymous*

JUSTIN TROUTMAN[†]

October 1, 2007

*This is the original draft, which was adapted for the Security Watch column of Microsoft TechNet Magazine’s October 2007 issue; the adaptation is dubbed, “BitLocker and the Complexities of Trust.”

[†]Extorque, 114 Union Heights Blvd., Salisbury, North Carolina, 28146, USA. E-mail: justin@extorque.com

Contents

1	Traversing the credits before the show starts	3
2	The PGPotter and His Kiln of Wisdom	4
3	A Poor Man and His Elephant	5
4	The Figuratively-Two-Tusked Elephant	6
5	Musings Until Then	7

Windows Vista. I've not uttered either of those two words in an article before, which is why this article makes me feel like a five-year-old in FAO Schwarz with a blank check. My word budget is hefty and there's a considerable amount of real estate to cover when tackling the huge suburban expanse of peripheral discussion avenues radiating from the cynosure that Microsoft is. A while back, after perusing Bruce Schneier's security blog, I came across an entry discussing a creature that was new to me at the time – BitLocker. Apparently, this would provide Vista's cryptographic functionality. I know what you might be thinking. "Oh no. Vista is doing cryptography? It can't be any good. Even if it is, there's gotta be a back door somewhere."

Well, I'm not here to rant about historical reasons for not placing your bets on Windows' security, but I'd like to build a case on why I believe there are sound reasons for hearing BitLocker out, as well as pitch in a few cents on the kind of philosophy that ensures the vitality of cryptographic software and hardware. Make no mistake, though. By no means am I proposing BitLocker as an elixir for security as a whole; it is one of the many panes of the security window. Its goal is to provide resiliency under certain threat models. Imagine an adversarial version of the Great Bambino, calling a shot towards that window. If it hits the pane of cryptography, do you trust it to repel the shot, or shatter like the Royals' chances for a division title in 2006? That's what this article is about – with a little less antonomasia-induced baseball talk.

1 Traversing the credits before the show starts

I haven't even seen BitLocker yet, and – I know, I know – you're probably wondering, "How can you have an opinion on it, if you haven't seen it?" Let's say you rent a movie from Blockbuster, pop it into your VCR, and soon realize that the person who rented it before you didn't give a second thought to the "Be Kind. Rewind," policy. Perhaps you immediately press the rewind button on your remote, or perhaps you watch the credits scroll up, revealing the primary cast. I'm well aware that the inclusion of a good cast doesn't necessarily seal the deal for an inherently good movie, but it can speak a lot about what one can potentially expect from the structure of the film, and its personality.

A similar situation applies here. Upon reading Bruce Schneier's blog, I saw the title, "Microsoft's BitLocker." I thought to myself, "This is either worthy enough to garner itself an honorable mention, or ridiculously bad enough to deserve being called out." From left to right, my eyes receive the text with anticipation for the latter, but, surprisingly – the final verdict is positive. One particular sentence stood out. This could have been because it was partitioned into its own paragraph, despite being a single sentence, or perhaps it was the confident tone of the sentence. And I quote, "There aren't any back doors for the police, though." This statement packs a lot of punch.

I was curious as to where this confidence was coming from, so I followed the link. From Bruce's blog, my 6MB DSL connection promptly took me to the blog of Microsoft's System Integrity Team. I was impressed with the author's almost-martyr-like denunciation of the speculative-at-best rumors. He was straightforward in stating that back doors were unacceptable and he'd have no part in a project that supported them; it wasn't until I saw the author's name, however, that it all made sense. Oh, right – the name. "Niels Ferguson," it was signed. Not only did it explain Bruce's confidence-emitting statement, but it sent a little of that confidence my way, as well. This relates to the film credit analogy. By the way, you might know Bruce and Niels as the co-authors of *Practical Cryptography*, a seminal book about applying good cryptography, simplistically, correctly, and securely, and the co-designers of

the block cipher, Twofish, a 128-bit Feistel network, that earned its cryptanalytical bones by surviving as a finalist in the AES selection process.

Again, just because there's a seasoned cryptographer attached to a project, this is no guarantee that the final product will be secure. It is no secret that even the Michelangelos and Rembrandts of this art miss a few strokes here and there, and have their schools of design dismissed and broken. It's part of the artistic science (or scientific art, depending on your perspective) – a fixture of expectation, if you will. So, regardless of how BitLocker really turns out, we, at least, have a signal of hope that sound design strategies played some part in its evolution. A big, sarcastic “way-to-go” goes out to the Jerry-built cryptography that has unfortunately succeeded in its sometimes advertent, but sometimes inadvertent, bourgeoisification campaign in the software and hardware market that it has filled with more holes than Swiss cheese at an acupuncture appointment.

You have the bona fide attempts by developers without a lick of cryptographic sense, and then you have the mala fide attempts by likewise clueless consumivores who care more about the crisp Franklin than consistent fidelity. Both attempts, regardless of their intention, are security failures in the making. BitLocker's impact will be significant, and it makes exhaling quite a bit smoother and more relaxed, to know that at least one competent cryptographer is on deck. Obviously, this doesn't mean that BitLocker is exempt from failing just as miserably as the aforementioned attempts by the incompetent; it does, however, increase the potential for pumping out some quality cryptography.

2 The PGPotter and His Kiln of Wisdom

Not too long ago, Phil Zimmermann – remember, it's N Duo, not N Solo – generously shared some rules-of-thumb with me, that, should there ever be a published “Security Creed For Developers,” would grace the top of that magna carta. While these weren't intended to be specific to BitLocker, the design philosophies they promote can be applied to most any cryptographic instantiation. You might look at these and think, “Oh, now you're just being over-zealous. It's not *that* big of a deal.” Although there are some cases for which we wish it wasn't, it remains a given. Given that, I'd like to see a state of security where developers react to these proverbs with a Pavlovian demeanor; that is, they hear it and immediately respond with, “Thank you, Captain Obvious!”

“Chop-chop,” you say, “with the proverbs!” When designing a cryptographic infrastructure, you must be simple, correct, and secure. Although mistakes are inevitable, they should not be accepted as a natural occurrence and merely shrugged off with an apathetic “Oh well.” You must be a staunch advocate of nothing less than meticulous perfectionism. Phil asseverated, “Design as if making a mistake will cost someone's life.” Don't be too hasty to mumble, “Yeah, right. Like that really happens.” Quoting recently retired NSA cryptographer, Brian Snow: “I want functions *and* assurances in a security device. We do not 'beta-test' on the customer; if my product fails, someone might die.” This is from his address, as a “Distinguished Practitioner,” at ACSAC 2005, while he was still active at the NSA. Mistakes can have more than a monetary cost, folks. Designing by this “. . . or else” attitude isn't exceeding the call of duty; it's the belt that holds the pants up. And to think - some still wonder why we're constantly caught with them down?

Users deserve assurance; security should come packaged with it. The trust of the user is pivotal. “Publish source code,” professes Phil. “Don't trust crypto products that don't.” In *Practical Cryptography*, we're conditioned to look at cryptography as a mechanism for minimizing trust. Limit who you

trust and the extent to which you trust them. When users are left with developers offering the shallow reassurance of, “Trust me – it’s secure,” there’s really no incentive for trusting. Shift the trust from requiring the user to trust the word of the developer, to the user simply trusting the cryptography. Open the source. Analyze it mercilessly. Document it thoroughly. Present it impeccably. A 4-man San Diego band once rocked, “It’s in the suit that you wear.” Competent precision is a carrier of assurance, so show users that you’ve got a clue as to what you’re doing. Let them rest easy; let them rest assured.

You’re going to have to maintain that trust too. Phil makes this very clear when he says, “Earn the trust of the users. Once you shoulder that burden, you can never put it down.” The fruit of maintaining it is two-fold; you preserve the user’s assurance and you preserve your reputation as a source of secure software or hardware. Losing both is sometimes irrecoverable, and will, without a doubt, be difficult to pry off your name. I’m talkin’ leech-with-a-vengeance difficult. A 5-man Calabazas collective once melodized, “Did you ever meet a leech who was good at goodbyes?” To the developers: If security (i.e., in the form of privacy) is a dividend to you, you’ll probably pay no nevermind to this, but if it’s a natural right of humanity you’re protecting, you’ll probably concur.

Alright, so I’ve spent over 1500 words on the kind of attitude Microsoft should have, and now I’ll spend around the same on why I’m optimistic about Microsoft’s potential for exhibiting this attitude and rendering what is poised to be a piece of cryptography that should be taken seriously. I’m ready. How ’bout y’all? Right this way.

3 A Poor Man and His Elephant

Don’t worry. We’re still talking about cryptography here, but don’t feel bad if you had sudden thoughts of weathered behemoths with ivory headlights in Amboseli, or a fellow who’s a few bucks short. You’ll see where I’m going with this, in a little while. Until then, let’s take a look at BitLocker’s guts. This implies that Microsoft has done a great job at documenting the preliminary analytical summary of BitLocker, and thanks to Niels Ferguson, this is case. Thanks, Niels.

Okay, so what is it? At its core, BitLocker is how Vista (at least Enterprise and Ultimate editions, for the time being) encrypts all system volume data. This sounds like a commonplace application, until you consider the constraints. Since BitLocker encrypts data at the per-sector level, and the ciphertext cannot exceed the length of the plaintext, this leaves no breathing for anything else – nonce, IV, and MAC, to be exact. Y’all might know how I feel about MACs, and if you’ve skimmed through a copy of *Practical Cryptography*, you’ll see how Niels feels about them as well. Fortunately, I saw this coming. I realized the likelihood for such tight constraints and the conditions they would impose, and I knew that somehow, some way, message authentication would be addressed. I mean, c’mon, authentication is the Orville to encryption’s Wilbur; without both, it’s not going to fly.

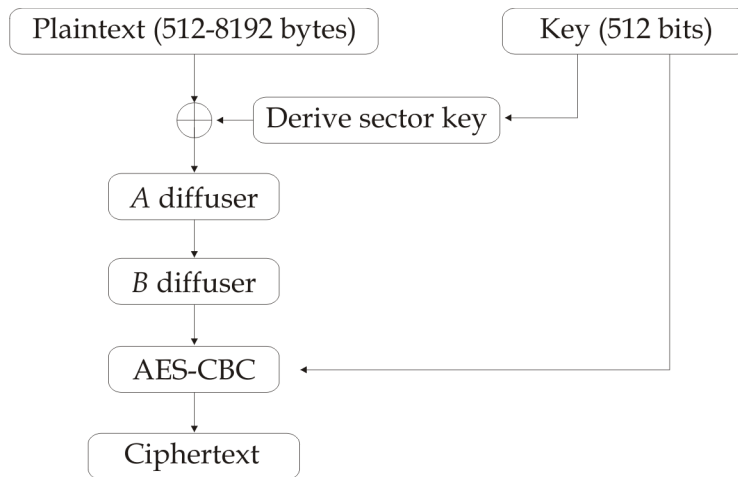
BitLocker invests in what is called *poor-man’s authentication*. The compromise isn’t as conservative as you’d usually want to be, since it assumes that manipulated ciphertext doesn’t result in meaningful plaintext; in other words, should ciphertext be manipulated, it should, for example, cause the system to crash, rather than be a controlled change that allows the adversary to carry out some desired function. Niels and Co. knew that an entirely new block cipher would require more time to analyze than is acceptable, but existing designs that were kosher enough, security wise, either hadn’t been analyzed well enough yet, or couldn’t pass the efficiency muster. So, for encryption, they

chose AES in CBC mode, which is IND-CPA secure (i.e., indistinguishability under chosen-plaintext attacks), but does not preserve integrity. That’s okay; we can use a MAC. . . uh oh. There’s no room for a MAC, and CBC is a mode of confidentiality, so there’s no integrity preservation at all. It’s time to call in the elephant.

4 The Figuratively-Two-Tusked Elephant

Several compromises had to be made, and in a situation such as this, custom tailoring was the only route, given the lack of pre-existing options. Remember this maxim: *Security is a trade-off*. And so it goes, practical environments rarely spawn a Homeric lotusland climate. Even though we have a treasure trove full of secure cryptographic primitives, there comes a time when none of them will fit. How well did Microsoft improvise? Impressively well. Consider the certainty that cryptography generates overhead, and this overhead must be acceptable; if it’s unacceptable, users will disable it. Why? For many, convenience is greater than security. I doubt Windows Vista (Molasses Edition) is on the company’s radar.

Meet Elephant. Elephant sports two diffusers, which are built to render much better poor-man’s authentication than vanilla AES in CBC mode (which shall hereby be dubbed “AES-CBC”); it should be noted, however, that AES-CBC mode, without Elephant, is an option, because there of those of you out there who must obey strict standards compliance policies. You’re going to lose integrity preservation this way, but this appears to be an inevitable caveat of the conditions that BitLocker is working under. Although not ideal, poor-man’s authentication is the most suitable solution for these constraints, and Elephant’s goal is to make the best of it. Until a more detailed analysis is published, I’ll laymanize the general voodoo that composes its structure. To help elucidate the flow of understanding, here’s a simple graphic.



When encryption takes place, four operations occur. The plaintext is combined, via XOR, with a sector key. At this point, it flows through two unkeyed diffusers. From there, it is encrypted using

AES-CBC. The sector key, and AES-CBC, are the two components that require key material; this leads to the crucial aspect of keying them independently. By doing so, it simplifies the formalization of a proof for reducing the security of the AES-CBC and Elephant construction to that of AES-CBC. This dexterous tactic balances the scale a little, since Elephant is a new primitive, and new primitives will be met with reluctance until they're rigorously analyzed. Having the ability to show that the AES-CBC and Elephant construction is no easier to attack than AES-CBC alone, is perhaps its flagship property.

The sector key and AES-CBC components both receive 256 bits of key material, which brings the key length to 512 bits, in its entirety. By default, however, the sector key and AES-CBC components use only 128 bits of key material, respectively; this means that some of the key material isn't used. The reasoning behind this is simple: It's easier to throw away bits you don't need, rather than change the key management infrastructure as key lengths vary. Speaking of variable lengths, the block length is allowed to be any power of two, within the range of 512-8192 bytes. In fact, to ensure that any alteration to the ciphertext results in all of a sector's plaintext being modified in a random way, the block cipher is designed to behave as a block cipher with such a variable block size; this is good poor-man's authentication at work. Furthermore, if the block cipher behaves like a "tweakable" block cipher, as described by Liskov, Rivest, and Wagner, at CRYPTO '02, with the algorithm changing slightly, from sector to sector, an adversary won't be able to successfully move one sector's ciphertext to another sector.

Pardon the Yoda-esque tone of this, but: Resourceful, BitLocker is. Secure, as well? Strides have obviously been taken, but analytical time will tell how well. BitLocker has its sight honing in on one particular threat model, specifically – the lost or stolen laptop. It's no secret that folks love to lose track of 'em. All sorts of intended-to-remain-confidential information is waiting to be opened, like a fortune cookie (and as easily as opening a fortune cookie, too, given the absence of security on a great many of those laptops). As long as we're still entrusting humans with the security of devices – let alone keeping up with them – the obvious is inevitable. It's not a matter of eliminating the sloppiness of human-nature, but, rather, isolating and mitigating its mess. After all, it's a whole lot cheaper to replace a laptop, than foot the cost of compromised data.

5 Musings Until Then

Like a rabid gnat, folks are buzzing with speculation about the cryptographic security of BitLocker, but much of this is short-sighted, and I'll explain why. For BitLocker to be secure isn't necessarily sufficient for it to be able to do its job. Keep in mind that BitLocker is one of many "Vistappendages." Microsoft mentions BitLocker as being tightly integrated into Windows Vista. I believe in modularity, and local failure. Is it possible for some other part of Vista to fail, and cause BitLocker to fail? If there is tight integration, is it plausible to imagine failure that is global? That is, if one part of the system fails, could BitLocker be one of the other parts that it takes down with it?

Tight integration with a lack of modularity leads to complexity. A particular Portuguese idiom paints a realistically surreal portrait of what complexity might look like, if it were a Herculean foe. That is, "bicho de sete cabeças," which literally translates to, "creature of seven heads." The underlying meaning it expresses is that of treacherous complexity. (My favorite Brazilian – a Mineira, to be exact – made sure my infantile Brazilian Portuguese was polished and pristine for the sake of this article, so it's with her help that I'm confident enough to speak with a foreign tongue!) All in all, this

may not even be the case with Vista, but only when an ample supply of analysis becomes available will we see what the situation looks like.

I was recently asked, “So, what do you think about BitLocker?” The answer is two-fold, really. Even if it turns out to be not-so-good, from a cryptographic standpoint, I still tip my hat to Microsoft’s System Integrity Team. Why? Because Microsoft ushered in the cryptographers who did things like they should be done – like a cryptographer. You’d be surprised at how many cryptographic systems are designed without the aid of anyone actually well-versed enough in cryptography to be designing such a system to begin with! For me to say, “BitLocker should be taken seriously,” is reasonable. This is setting aside the verdict of whether or not its security is sound. Many a cryptographic primitive and protocol have been broken, but at the same time, given us a platform of understanding for building something better. If anything, BitLocker is this platform; it could even be solid enough to hold its own.

Before you moan and wail about the stereotypical surface of things – that is, Microsoft doing security – pay attention to the cryptographic philosophies that begat BitLocker. There’s no better time to shift the paradigm of how folks perceive Microsoft’s attention to cryptographic detail, and in my humble opinion as a gradually maturing cryptographer, Microsoft is exhibiting a healthy helping of potential in this regard. My hat’s off to you, System Integrity Team.