Conceptual Disclosure

Conceptual Disclosure

Transparent disclosure fundamentally requires both truth and honesty. Truth in the context of software security involves first and foremost informing consumers in plain English regarding the actual threats to security and privacy a given solution can actually thwart. Honest disclosure is a much higher hurdle, for it requires, the enumeration of the known threats that a given solution cannot effectively address. We have labeled this combination of truthful and honest disclosure, Transparent Disclosure. We would like to further the direction and discourse on disclosure by introducing what we have termed Conceptual Disclosure, which involves outlining all of the conceptually possible weaknesses of a given security solution irrespective of their practical risk or probability of success. **As such, the purpose of this document is to fully disclose the conceptual threats to FN's protection of User data.** Further, the reader should be aware that even after enumerating all of the conceptual issues that we currently understand could compromise FN's solutions, there is no guarantee that of certain tools or methods don't exist that we are simply unaware or of course the future tools could be invented capable of usurping FN's protection schemes. Said another way, 'we don't know what we don't know'.

However, in fairness to ourselves, it is important to note that most, albeit perhaps not all of the enumerated conceptual threats are applicable to all software (and software security vendors) alike. Further, most, albeit perhaps not all of the issues noted herein are solely conceptual in nature and thus not pragmatically relevant (at least currently, to the best our knowledge). We would further note in the context of full disclosure and its more liberal nomenclature (CYA) that cyber-security is a very broad topic and one which is dynamically evolving and as such, we fully expect to routinely update this disclosure, so please consider it as a continuous work in progress.

Weakness	Comment
Known Vulnerabilities, Tools or Attacks Capable of Compromising User data	**None. Zero. Zilch. Nada. There are no known methods that can compromise FN protected User data locally, on the server, or across the network** as corroborated by FN's assurance testing as well as the testing of those attack tools and methods known to and utilized by, by BT Counterpane. Further, we continue to work with various organizations and researchers and also utilize rewards and bounties as an additional means of more explicitly and fulsomely understanding the complete security properties of our solutions.
Hardware Compromised or the Hardware Manufacturer is the Attacker	If the User's hardware is compromised or if in fact the hardware manufacturer is the attacker, FN's security may be compromised for certain hardware components. Additionally, the use of certain of FN's current method(s) of protection, such as for example the use of a virtual keyboard to attenuate hardware key logging and/or attacks by the keyboard hardware manufacturer does protect user data, but creates a "challenged" user experience, and as such is not particularly practical for all but the most secure of user requirements or use cases. (Note: We believe this vulnerability can be mitigated by FN development plans focused on Agent Aware Hardware).
Operating System Compromised or OS Vendor is Attacker	If certain specific components of the User's Operating System (OS) are compromised or if in fact the Operating System manufacturer is the attacker, FN's security also may be compromised depending on the nature of attack. (Note: We also believe this vulnerability can be mitigated by FN development plans focused on Agent Aware Hardware).
Industry Standard Cryptography Compromised	If certain industry standard cryptography is compromised, such as RSA 2048, FN will not by technically be immediately compromised, for FN also utilizes several diversified super encryption mechanisms to further protect User data. That said, it is a reasonable conclusion that any attacker capable breaking industry standard cryptography is also capable of breaking FN's diversified algorithms. However, assuming that some form of secure cryptography exists and is available for FN's use, FN's software has been designed to quickly and easily adapt. Specifically, FN's software has been designed to 'fail gracefully' through the use of the software's update feature enabling the seamless replacement of any compromised algorithm(s) with other, stronger cryptographic algorithm(s) or for that matter any failed component. That said, the interval between FN's knowledge of the breaking of an industry standard algorithm (or any component) and the update of the User's software with a un-compromised replacement algorithm(or other component) is clearly a vulnerable period of User data protection.
Non-Randomness	Certain elements of FN's software, most specifically its cryptographic implementation utilize 'random numbers' or perhaps more accurately 'pseudo random numbers'. Conceptually this lack of true randomness creates cryptographic vulnerabilities.

No Quantum Cryptography

FN does not currently utilize quantum cryptographic algorithms. However, the FN framework has been designed to easily enhance the cryptographic implementation of our solution, which would include, but not be limited to, quantum cryptography.

Lucky Guess

It is generally understood that mathematicians can accurately assess the average number of iterations or guesses required to break/guess a crypto key, enabling the extrapolation, for a given hardware of certain robustness, of the time required to break any string (key) of given length and composition. However, there is no way to prove, mathematically or otherwise, that a lucky guess by an attacker, say on their first try, will not breach a system mathematically 'proven' to take years on average, to break. Further, we cannot prove that someone, somewhere, doesn't have a computer, running, hundreds, thousands, millions or even billions of times faster that today's hardware, rendering brute force key enumeration trivial.

No Mathematical Proof of Correctness

Mathematical proof of code operation is unfortunately infeasible for any large code base (such as FN's) given the scalability and cost challenges of current implementations.

Analog Hole

Although FN has made some progress related to some elements of the DRM problem for certain types of content such as text. However, FN has no method, nor foresees to have any method capable of addressing the "analog hole" issue. Conceptually, the challenge of stopping someone who has seen something from remembering it and later recreating it, or from digitally capturing content 'out of band' say via a digital camera (albeit perhaps at lower quality), that a User has rightful access to is at worst an intractable problem and at best one that will require much greater minds that our own to solve.

Malware Operational Impacts

FN's solutions primarily focus on protecting User's from data or content theft. However, as solely related to non-data theft oriented malware malfeasance, such as by way of example pop-up ads, local Denial of Service (DOS), use of the device as a botnet, etc. certain elements of FN's technologies have a role to play, but by no means are they nearly as complete or robust as FN's data protection solutions.

Compromising Emissions Attack

Sometime referred to as TEMPEST attacks and as defined per Wikipedia as "compromising emanations consisting of electrical, mechanical, or acoustical energy intentionally or by mishap unintentionally emitted by any number of sources are defined as unintentional intelligence-bearing signals which, if intercepted and analyzed, may disclose the information transmitted, received, handled, or otherwise processed by any information-processing equipment. FN has **not tested** its solution set related to this threat, but at least as related to the instance of this threat where plaintext could be recovered from an encrypted message, FN's use of diversified algorithms and the super-encryption of User data early in the creation of said data, as well as the use in certain instances of FN's floating keyboard, conceptually make these types of attacks much more difficult (we believe, **but without the benefit of empirical validation of any kind).**

Distributed Denial Of Service Attacks

Currently we believe we have achieved a partial solution that is partially effective. Specifically Fn has reduced application and server bottleneck performace vulnerabilities such that effectively a hackers sole option to effect a server DDOS attack would be bandwidth flooding, a marginally higher hurdle than exists today. However, we believe that this hurdle can be made significantly more difficult if the gate-keeping fabric of the internet (i.e. routing infrastructure or modems) was made identity aware, consistent with FN's approach to the rest of the IT stack, a requirement that we recognize suffers from daunting implementation/ instantiation challenges. Further, we believe that the best short term as well as the most feasible anti-DDOS solution is one of the few areas where regulation can play a value-added role. Specifically, ISP's can easily monitor offending machines and take whatever action is deemed appropriate: from User warning, to reducing quality of Service/bandwidth, denying internet access, or the provision of relevant User data to legal authorities, as a non-exhaustive set of examples.

No External Code Review – Coding Error's and/or the Software Vendor(FN) is the Attacker

There are a number of reasons that we at FN believe that 3rd party code reviews above and beyond the company's own internal practices will have negligible impact on the overall security of User data. First and least importantly, FN is by design and in practice particularly diligent in its coding. As noted, FN has gone to extreme lengths for commercial software in an effort to reduce the probability of coding errors by utilizing the code maintainability standards required of mission critical technologies such as the NASA space shuttle (full disclosure: we have not yet implemented 178B full graph traversal testing). Second, source code review is a blunt, non-scalable instrument and at the extreme, it is a practical impossibility that for every change of even one line of source code requires a full review, although I am sure this theoretical reality is much to the delight of source code review consultants and penetration testers whom inevitably strongly endorse these methods non-coincidentally due to their potential as perpetual full employment mechanisms. Even if we were more pragmatic about a more realistic implementation requirement, one would need to be equally pragmatic about the operational challenges and ultimate lack of robustness that source code review as a practice actually provides. Third, and particularly as related to FN as the attacker, FN utilizes both diversified templated code sets and binary obfuscation in the manufacture of its software, it would be very difficult to further ascertain that a given version of the source code wasn't modified prior to manufacture. Said another way, linking transformed binaries to a given instance of source code is not a trivial task, let alone a scalable one. Lastly and most importantly, one cannot guarantee that even with a full and continuous source code review of Fn's software, that say the compiler code, or that the hardware manufacturer code, or any supplier of hardware components and so on ad nausea, was not inserting malicious code. Further, it is unlikely that in the foreseeable future that the likes of Apple, Intel, Google, Microsoft et al will provide their source code for review and as such the partial checking, of part of the chain of custody, provides little actual protection, despite the pithy albeit naïve views and claims of the open source community's to the contrary. In fact, more generally, the 'root of trust issue' is de facto intractable as we would all need to build our own software for it not to be. And of course pragmatically there is more than enough of a challenge right now protecting users from 'the bad guys' that once this problem is solved, we are confident that the means and methods can developed which insure the integrity of all components utilized in the User data protection chain of trust.