Global Information Assurance Certification Paper

Stephen A Shepherd February 21, 2003

Vulnerability Disclosure

How do we define Responsible Disclosure?

Table of Contents

Vulnerability Disclosure

How do we define Responsible Disclosure?

Abstract

It is inevitable that vulnerabilities will be discovered in the production of information technology products, regardless of how much time and effort is placed into identifying and removing flaws during initial development. Based on this inevitability one would surmise that a logical structured procedure would be followed for disclosing newly discovered vulnerabilities. However the current process for disclosing vulnerabilities can range from a loosely organized effort to utter chaos.

This lack of structure has caused the eruption of a heated debate within the information security community. This debate has been going on for almost a decade. Yet to date there is no formal, accepted, and enforced standard of practice. Each side in this debate has expressed valid concerns both for and against the various concepts of disclosure. As a result this vigorous debate has given rise to the new term "Responsible Disclosure".

Within this document I will attempt to define "Responsible Disclosure". I will briefly explore some key events in vulnerability disclosure. I will also attempt to explain the conceptual differences between full disclosure, nondisclosure, limited disclosure and responsible disclosure. Finally I will examine some existing disclosure policies and proposed standards.

Key Events

Creation of Bugtraq

Before the creation of the Bugtraq vulnerability mailing list in the late 1990's vulnerabilities were passed around between a close circle hackers, software engineers, and security professionals. These "Trade Secrets" were used for both benign and malicious purposes and rarely captured enough publicity to draw the eye of the general public.

Once Bugtraq came on the scene this cloak of ambiguity was ripped away. Bugtraq offered an open forum for discussing and disclosing the vulnerability de jour. No longer were the weaknesses of the Internet shared by the "Elite" now anyone could read how to compromise a system and download the tools to launch the attack. Both administrator and hacker wannabe alike had complete access to the tools needed to protect or violate vulnerable systems.

As awareness of the existence of Bugtraq spread high-risk vulnerabilities received large amounts of attention in the press and among other members of the information

technology community. This public recognition offered an incentive to hackers and security firms to publicize newly discovered vulnerabilities. Their skills and professional services were validated by this recognition. While such recognition often benefited the discloser it also assisted black hats in exploiting vulnerable systems.

In order to help others reproduce the vulnerability a discloser would often post complete technical details usually including proof of concept code. Black hats would then use this information to develop a scripted attack or an attack tool to automate the exploitation of the vulnerability. Widespread distribution of these attack tools has led to the rise of uncounted script kiddies. These script kiddies lack the technical skills to understand an exploit or to create an attack tool. Instead they download attack tools and launch them blindly against the public Internet.

Microsoft "Information Anarchy"

Code Red, Lion, Sadmind, Nimda, and Ramen regardless of whether full disclosure enabled the spread of these worms to epidemic proportions they did prompt a pointed response from Scott Culp of Microsoft. In Oct 2001 Scott Culp the manager of Microsoft's Security Response Center released his paper entitled "It's Time to End Information Anarchy"[1]. In this paper Culp directly challenges the Full Disclosure ("Information Anarchy") supporters. He claims that Full Disclosure arms the enemy with detailed technical information that is then used to develop automated attack tools. Armed with these tools anyone can launch attacks on vulnerable systems.

Supporters of Full Disclosure claim that they motivate administrators to patch their systems by disclosing exploit code. However the wide spread success of the worms listed above provides further support for Culp's argument that Full Disclosure with exploit code does not provoke administrators to patch vulnerable systems. Vendor patches addressing the vulnerabilities used in each of these worms were in some cases available as much as a year before the worm was released in the wild. It is Culp's assertion that releasing exploit code as a part of Full Disclosure only makes it easier for an attack to be launched.

Further Culp states that it is not necessary for an administrator to have access to exploit code in order to defend vulnerable systems. This statement has led to a lot of controversy. While it may be true that most administrators do not need exploit code to secure their systems there are instances were exploit code would be used. For example an administrator might make use of exploit code to test for the existence of vulnerable systems. Exploit code may also be used to test the integrity of a patch that has been distributed to correct a vulnerability.

It would seem that over all Culp's recommendation is in line with other Responsible Disclosure supporters. The key Responsible Disclosure concepts present in his essay are notifying the vendor, delaying public disclosure until a patch is available, and finally public disclosure without exploit code.

Public Criticism of ISS

During the latter half of 2002 Internet Security Systems (ISS) received considerable criticism over it's handling of several vulnerabilities. ISS discovered flaws in Internet Software Consortium BIND software, Apache web server software, and Sun Microsystems Solaris X Windows font service. In all cases ISS notified the vendor and worked with them to coordinate a public disclosure with patch availability. Irrespective of whom the fault lays with the handling of each incident was not well received by the public. In the case of the ISC Bind vulnerability patches were not distributed quickly. The vendor patches for the Solaris Font problem were flawed and had to be recalled. Finally the Apache patch was not distributed until after public disclosure, even though black hats had been exploiting the vulnerability for over four months.

Because of these events ISS was motivated to release a public policy[2] describing each step it would take when disclosing a newly discovered vulnerability. The ISS disclosure policy contains several of the key Responsible Disclosure concepts with one notable exception. ISS declares that it will disclose the vulnerability to paying subscribers of its service one day after notifying the vendor. Further they may incorporate testing for the new vulnerability within their security products. I think it is interesting to note that one of the key goals of Responsible Disclosure is to keep knowledge of vulnerabilities within the smallest circle of people until a patch can be developed and made public. What is to prevent a black hat from subscribing to the ISS subscription service and receiving a notice of the vulnerability one day after the vendor is notified? Granted ISS will not be disclosing technical details about the vulnerability but the black hat will know what type of vulnerability exists and in which part of a vendor's product. That knowledge may assist the black hat in developing an exploit and using it on vulnerable systems before a vendor patch is available.

With greater frequency security research companies are being criticized for disclosing vulnerabilities for the sole purpose of generating favorable press coverage. The media coverage a security company receives can mean substantial revenue in the form of new or larger customer contracts. Because of this the public is starting to question the true motivation behind some of the vulnerability research and disclosure. In some cases the vulnerabilities being disclosed by security firms are the result of intense stress testing of products. The likelihood of these vulnerabilities being discovered outside of this manufactured lab environment is small. This poses the question as to whether these vulnerabilities should even be disclosed.

Terminology

In researching the topic of vulnerability disclosure I have encountered a wide variety of terms. Unfortunately existing policies, purposed standards, and articles on the subject tend to use different terminology. In an effort to present the following concepts I will us the definitions below in my writing.

Product: a software or hardware product.

Flaw: a flaw in the logical operation of a product. The behavior exhibited by the flaw is such that the product is left in an undesirable state.

Vulnerability: a flaw becomes a vulnerability if the exhibited behavior is such that it can be exploited to allow unauthorized access, elevation of privileges or denial of service.

Exploit: a tool or script developed for the sole purpose of exploiting a vulnerability.

Exploitation: the act of using an exploit against a vulnerable system for the purpose of gaining unauthorized access, elevating privileges, or denying services.

Discoverer: the first person to reveal a flaw and determine that it is a vulnerability. Depending on how the vulnerability is discovered the discoverer may or may not be known. For example if a vulnerability is released anonymously the identity of discoverer may not be apparent.

Originator: the person or organization that reports the vulnerability to the vendor.

Coordinator: a person or organization that acts as a liaison between the discoverer and the vendor. The coordinator may perform any of the following activities, initiate contact with the vendor, reproduce the vulnerability, or coordinate public disclosure. Possible coordinators might be CERT, SANS, or FIRST.

Vendor: the person or organization that is responsible for maintaining the vulnerable product.

Customer: persons or organizations that use the product and are exposed to the vulnerability.

Vulnerability Life Cycle

In the paper "Windows of Vulnerability: a Case Study Analysis"[3] William Arbaugh, William Fithen, and John McHugh define the life cycle of a vulnerability. I have summarized the seven stages below. Using this model we can analyze the objectives of each type of disclosure.

Birth: The birth stage denotes the creation of the vulnerability during the development process. If the vulnerability is created intentionally then the birth stage and the discovery stage occur simultaneously. Vulnerabilities that are detected and corrected before deployment are not considered.

Discovery: The life cycle changes to the discovery stage once anyone gains knowledge of the existence of the vulnerability.

Disclosure: The disclosure stage occurs once the discoverer reveals the vulnerability to someone else. This can be any disclosure, full and public via posting to Bugtraq or a secret traded among black hats.

Correction: The correction stage persists while the vendor analyzes the vulnerability, develops a fix, and releases it to the public.

Publicity: In the publicity stage the method of achieving publicity is not paramount but knowledge of vulnerability is spread to a much larger audience.

Scripting: Once the vulnerability is scripted or a tool is created that automates the exploitation of the vulnerability the scripting stage has been set in motion.

Death: When the number of systems vulnerable to an exploit is reduced to an insignificant amount then the death stage has occurred. This can happen by patching vulnerable systems, retiring old systems, or a lack of interest in the exploit by hackers.

Types of Disclosure

Nondisclosure

To have a policy of Nondisclosure means to keep the vulnerability information tightly contained so as the general public never learns of its existence. The black hat community practices a policy of Nondisclosure. When a vulnerability is discovered by a black hat the information is kept by that individual or judiciously distributed within a black hat group. These black hats then use the vulnerability to penetrate unprotected systems for whatever clandestine purpose they desire. Eventually the vulnerability information leaks out and is released in a public forum. However before this time systems and their administrators have no defense against exploitation.

Some vendors and security firms have tried to promote a policy of Nondisclosure. They feel that the vulnerability information can be controlled and only "trusted" individuals will be informed. In this way they can "protect" the vulnerable systems until a fix can be made available. The major flaw with this thinking is the belief that the information can be controlled. There is no way to assure that the selected individuals can be trusted not to use privileged vulnerability information for their own gains. Furthermore some of the individuals employed by vendors and security firms have questionable histories. Can we really trust "reformed" individuals with past careers as black hats or grey hats to act responsibly with privileged vulnerability information?

Obviously adopting a policy of Nondisclosure has several disadvantages and few advantages. On the plus side a Nondisclosure supporter might argue that controlling the disclosure of a vulnerability will help keep the information out of the hands of black hats. However there is no way to assure that the black hat community does

not already possess the vulnerability information or that they will not discover it on their own before a public disclosure is made. The only real advantage of Nondisclosure is to the vendor alone. If a vendor can keep a vulnerability secret while it is fixed the vendor can avoid any negative press that may be generated.

There are numerous disadvantages to a policy of Nondisclosure. First if vulnerability information is leaked or simultaneously discovered the black hat community has an opportunity to actively exploit the vulnerability. Systems will be left exposed during the time it takes for the software vendor to patch the product. Second since the vulnerability is not disclosed publicly administrators do not have the opportunity to protect vulnerable systems. Next because there is no negative press for the software vendor they are not motivated to repair the flaw in a timely manner. Finally it is impossible to define who is the "trusted" subset of individuals that should have access to sensitive vulnerability information. Because of these reasons a policy of Nondisclosure is obviously less than desirable.

Full Disclosure

At the other end of the disclosure spectrum is a policy of Full Disclosure. In his paper "Exposing Infosecurity Hype"[4] Jay Heiser defines Full Disclosure:

> The term 'full disclosure' is marvelously ambiguous, and therein lies much of the problem. It essentially means to 'widely disseminate as much information about system vulnerabilities and attack tools as possible so that potential victims are as knowledgeable as those who attack them.'

Supporters of Full Disclosure argue several advantages. Firstly a vendor is motivated to provide a timely patch or workaround to a new vulnerability. If the vendor fails to provide a timely fix and a vulnerability is disclosed fully and the resulting negative media will cause damage to the vendor's reputation and revenue. Further, in order to avoid future negative media a software vendor is motivated to create less vulnerable products.

Next, Full Disclosure advocates state that black hat hacker community is already aware of vulnerabilities. By fully disclosing vulnerability information administrators of vulnerable systems are armed with the information needed to take action. Full Disclosure supporters believe that it is imperative that administrators and programmers fully understand vulnerabilities in order to prevent and defend against them. They believe that because administrators and programmers have access to full technical details of the vulnerability they can take appropriate defensive action. Defensive action can be any or all of the following: developing and implementing an Intrusion Detection System (IDS) signature to allow detection of the exploit and implementing a temporary workaround such as shutting down a vulnerable service or blocking traffic at a firewall. In addition to these defensive actions a systems administrator might use exploit code to scan the network for vulnerable systems or to test the possible vulnerability of systems that have been patched. Also programmers

can review the structure of the flaw and attempt to avoid similar situations in future development.

Although the concept of Full Disclosure does not preclude vendor notification most opponents point to the lack of grace period during which the vendor can address the flaw as a major disadvantage. In Full Disclosure the vendor is notified at the same time as the vulnerability information is fully disclosed. Because of this, systems are vulnerable during the amount of time it takes the vendor to address the vulnerability.

While it is true that talented black hat community may likely have prior knowledge of an exploit, the hordes of script kiddies do not. Those against Full Disclosure argue that fully disclosing a vulnerability including exploit code, arms the script kiddies. Next oblivious of any technical knowledge but armed with the automated exploit the script kiddies proceed to launch attacks upon the Internet public. Finally if the talented black hats do not posses prior knowledge of a new vulnerability Full Disclosure makes it considerably easier for them to develop exploit code and automated tools.

Limited Disclosure

As with Nondisclosure the main concept behind Limited Disclosure is that the vulnerability information is shared among as few individuals as possible. During the initial phases of disclosure only a small group is allowed access to the full details of the vulnerability. This group consists of the discloser, the vendor and possibly a third party coordinator. The initial public disclosure only describes the flawed product and includes very few details about the vulnerability. In the Limited Disclosure model even the final disclosure does not contain full technical details and will only be released once the vendor has fixed the flaw. The concept for limiting the amount of technical information is founded in the belief that users, programmers, and administrators do not need detailed technical information in order to patch systems. Further the disclosure of full technical information only assists the black hat community.

There are several problems with the concept of Limited Disclosure. As with Nondisclosure we are faced with the dilemma of whom to trust with the initial vulnerability information. It will be very difficult to enforce ethical behavior amongst those that may stand to gain from the disclosure or exploitation of an unknown vulnerability.

Without mandatory public disclosure there is nothing to motivate the vendor to develop a timely fix. Since the vendor can delay the final disclosure until they have fixed the flaw, final public disclosure can be delayed indefinitely.

Next since the amount of technical information in the initial disclosure is greatly limited customers may not be able to take early defensive actions. For example without detailed technical information IDS signatures cannot be created. Also the development of tools to detect vulnerable systems and test vendor patches will be

impossible to develop. Finally without a complete understanding of the structure of the flaw, programmers will continue to make similar mistakes when coding future products.

The issue of simultaneous discovery or discovery of a vulnerability already being actively exploited is also overlooked. In these situations systems will be exposed or exploited while disclosure is delayed until the vendor is ready to release a patch. Even if a final disclosure were published before the vendor releases a patch administrators would still lack the information needed to deploy the countermeasures discussed above.

Responsible Disclosure

One of the main objectives of this paper is to define Responsible Disclosure. Because of this I will devote substantially more space to this disclosure approach. I will first look at the key concepts that comprise a Responsible Disclosure policy. Then I will look at various policies and proposals that have been fielded to date.

Discovery

During this stage of the vulnerability life cycle the method of discovery will determine how responsible disclosure will be proceed. There are two ways that a vulnerability can be discovered. First a responsible party such as a security firm, white hat, or vendor programmer can discover the vulnerability. Second evidence that a black hat has discovered the vulnerability can be uncovered. In the first situation access to vulnerability information can be controlled until a patch has been developed and a public disclosure can be made. In the second situation the vulnerability is already being actively exploited therefore a public disclosure must be made in order for customers to defend their resources.

Initial Contact

Initial contact signals the start of the disclosure stage in the vulnerability life cycle. First similar to Limited Disclosure the discloser should always notify the vendor before any public disclosure is made. This contact should be done in such a way as to confirm that the vendor has received the notification. The use of a third party coordinator may assist in facilitating initial and continued communication between the discoverer and the vendor. If possible all communication should be secured to avoid premature leakage of vulnerability information.

Next a reasonable deadline for vendor response should be purposed and agreed upon. The generally accepted deadline is 30 days. However there maybe mitigating factors that demand the deadline be shortened or extended. Regardless of mitigating factors the deadline should not be extended indefinitely.

Finally an effort should be made to keep the circle of trust small. It is important that knowledge of the vulnerability be kept secret until a patch can be developed. I think it

is important to note that the concept of private "subscription" notification is not a generally accepted component of Responsible Disclosure. Some security companies such as ISS disclose information to subscribing customers shortly after vendor notification. Doing this increases the risk that vulnerability information will be disclosed prematurely or can assist black hats in discovering the vulnerability on their own.

Continued Communication

During the time after initial contact and until public disclosure all communication lines between the vendor, discoverer, and originator should be kept open. Any miscommunication during the entire disclosure process could lead to premature disclosure and the exposure of customer resources.

It is important that the vendor attempt to reproduce the vulnerability in order to verify its existence. If involved a third party coordinator may attempt reproduction as well. The originator should provide the vendor and coordinator with all the necessary information and aid reproduction in any way.

If the vendor does not respond to initial contact or fails to continue communication the originator has no option but to proceed with public disclosure without a vendor supplied patch.

Patch Development

The vulnerability life cycle correction stage commences once patch development starts. Obviously it is imperative that a patch be developed to address the vulnerability. However it is equally important to thoroughly analyze the vulnerability. The flaw that creates the vulnerability may be present in other parts of the product or in other products that share similar development. Also similar products implemented by other vendors may also be vulnerable. This is especially true for products developed from a shared code base or accepted standards. The vulnerability in Simple Network Management Protocol (SNMP) is a prime example.

If the vulnerability is found to be present in other products the circle of trust will have to be widened. Additional vendors may need to be notified and allowed access to detailed vulnerability information in order to test their products.

The involvement of multiple vendors can lead to confusion and miscommunication. Every effort must be made to keep all actions coordinated. If a single vendor releases vulnerability information prematurely the customers of the remaining vendors can be left exposed.

Finally all patches should be tested completely by both the vendor and the originator. Differences in environments and system configuration my cause a patch to have negative side effects.

Public Disclosure

The vulnerability life cycle publicity stage begins after a patch has been developed and fully tested. During this stage the originator, coordinator, and vendor cooperatively develop the content of the public disclosure. The public disclosure will be similar to Full Disclosure with one exception; the public disclosure will not include any exploit code. However full technical details will be disclosed including a tested patch, potential workarounds, and possibly an IDS signature. The idea here is to give administrators and programmers enough information in order to defend against the vulnerability, but make it harder for script kiddies to launch an attack exploiting the vulnerability. Another similarity to Limited Disclosure is that the public disclosure will be timed to coincide with patch availability. However if the vulnerability is leaked or being actively exploited the public disclosure will preempt patch availability.

Exploit Released

Once an exploit is released we enter the scripting stage of the vulnerability life cycle. Although the scripting stage can occur at any time it is the prime goal of Responsible Disclosure to prevent or at minimum avoid assisting its occurrence. If Responsible Disclosure is followed a patch will be release before the Scripting stage occurs. This will allow responsible customers to protect their systems from exploitation. It is important to note that if the Scripting stage occurs before public disclosure is made then the timeline for public disclosure should be escalated. This will allow customers to take precautions against possible exploitation.

Existing Policies and Proposals

In this section I will take a brief look at existing disclosure policies and proposed disclosure policy standards. I will make an attempt to classify each disclosure policy into one of the four general types, Nondisclosure, Limited Disclosure, Full Disclosure, and Responsible Disclosure.

University of Oulu[5]

Before I discuss the various policies and proposals I would like to mention the work of the University of Oulu Secure Programming Group (OUSPG). Although they have not published any proposed disclosure standards they have done considerable work in the area of Responsible Disclosure. They have written two informative conference papers on the subject. The first "The Vulnerability Process: a tiger team approach to resolving vulnerability cases"[6] is an in depth analysis of the events within vulnerability disclosure. In the second paper "Introducing constructive vulnerability disclosures"[7] OUSPG first defines a disclosure process then applies it to a discovered vulnerability in the Wireless Access Protocol (WAP) while documenting the results of the disclosure process.

In addition OUSPG maintains a list of resources on their web site listing, conference papers, journal papers, speeches, book, thesis, reports, white papers, policies, and news articles on the subject of vulnerability disclosure.

NTBugTraq Disclosure Policy[8]

According to the timeline of disclosure policies documented on the University of Oulu Secure Programming Group web site the NTBugTraq Disclosure Policy created in July of 1999 is one of the first formal disclosure policies. Although it is probably best categorized as a Full Disclosure policy it does contain several elements of Responsible Disclosure.

First the moderator of NTBugTraq Russ Cooper takes it upon himself to reproduce the vulnerability. He will work with the discoverer to verify that the vulnerability can be reproduced before moving forward with public disclosure. In order to avoid false claims of vulnerabilities reproduction is an important step in any form of disclosure.

Next depending on the severity of the vulnerability, Russ will encourage vendor contact or if severity is low, post a public disclosure. I would say this is the policy's first deviation from Responsible Disclosure. The vendor should be notified in all situations. Without fully analyzing the vulnerability the likelihood that the vulnerability occurs elsewhere cannot be accurately determined.

Then by assuming the role of coordinator and initiating vendor contact, Russ supports another element of Responsible Disclosure "Communication". After initial contact the vendor has 48 hours to confirm reproduction of the vulnerability or respond with an explanation why they cannot reproduce the vulnerability. Similar to Responsible Disclosure an attempt is made to delay public disclosure until a vendor patch is available. However if the vendor is not responsive public disclosure will proceed.

If the decision is made to delay public disclosure then Russ will delay a maximum of 14 days before disclosing the vulnerability. Most Responsible disclosure recommendations purpose a 30-day delay before disclosure so 14 days could be considered too short. Also similar to responsible disclosure the public disclosure is coordinated between the vendor and the discoverer.

Another significant deviation form Responsible Disclosure is the lack of guidelines on the content of disclosure. Russ places no limitations on the inclusion of detailed technical explanations, exploit code, or proof of concept programs.

Rain Forest Puppy "RFPolicy"[9]

RFPolicy is a vulnerability disclosure policy created in late 2000 by Rain Forest Puppy. It is his attempt to formally document the procedure and actions he will follow when he discovers a vulnerability. In publishing the policy he does allow and encourage others to use it. Since it's creation RFPolicy has been widely referenced and used as a basis for other disclosure policies. In addition Russ Cooper along with

many others are credited for their contributions. Because RFPolicy lacks any definition of the content of the public disclosure it would be hard to categorize it as a Responsible Disclosure policy. Still as with the NTBugTraq policy it has several Responsible Disclosure components.

RFPolicy defines a unique method for setting a public disclosure deadline. After initial vendor contact five working days are given to allow for a vendor response. If that time elapses without vendor contact a public disclosure is made. If the vendor responds then another five days is granted for continued communication. This five day revolving deadline continues until a coordinated disclosure can be made. However if at any time more than five days elapses without vendor communication the discoverer has the option to make a public disclosure. Where as I find the revolving deadline idea beneficial to promoting communication and vendor response I feel five days maybe too short. There are too many things that could interrupt communication and allow more than five days to elapse.

RFPolicy attempts to encourage on going cooperation between the vendor and the discover leading ultimately to a coordinated public disclosure that includes a vendor patch. However as noted above RFPolicy makes no definition as to the content of the public disclosure. The decision to include detailed technical information, scripts, or exploit tools is left entirely up to the discoverer.

IETF draft-christey-wysopal-vuln-disclosure-00.txt[10]

The IETF draft proposal draft-christey-wysopal-vuln-disclosure-00.txt was filed in February of 2002. Authored by Steve Christey of MITRE and Chris Wysopal of @Stake this was the first documented attempt to develop some type of standard for vulnerability disclosure. However it did not make it past the draft stage and was expired in August of 2002.

The draft defines a standard method for a discoverer to contact a vendor. Within this definition are recommendations for standard e-mail address and web pages containing contact information. The draft also discusses the responsibilities of a coordinator, but it does not define what groups should fulfill this role or how to create a single coordinator group. These are important additions to the Responsible Disclosure idea. It is often difficult for a discoverer or coordinator to make initial contact. In order to assure reliable initial contact it is important a standard channel be defined and adopted.

The vendor is required to reproduce the vulnerability with in 7 days of initial contact. Similar to RFPolicy the vendor must remain in communication with the coordinator and discoverer every 7 days. Failure to maintain communication could result in early public disclosure. As with RFPolicy 7 days may be too short to allow reliable communication. Further the vendor is given 30 days to develop a fix for the vulnerability. The coordinator and the discoverer are required to give time extensions if the vendor is acting in good faith. If the vulnerability is found to effect products of

additional vendors, it is the responsibility of the first vendor or the coordinator to contact the affected vendors.

A coordinated public disclosure with proper credit is made once the vendor has completed patch development. In order to address the idea of Full Disclosure the draft defines the concept of a "Grace Period". This is a suggested period of time after the public disclosure before full technical information including proof of concept code is disclosed. Although the "Grace Period" does allow customers a window to defend their systems the posting of full disclosure information does nothing to address the arming of script kiddies.

"The Responsible Disclosure Forum" – Russ Cooper's Proposal[11]

Above we reviewed the disclosure policy for NTBugTraq. In November of 2001 Russ Cooper also posted a proposal on the NTBugTraq website for comment and debate. His proposal "The Responsible Disclosure Forum" is an attempt to reach a compromise between all sides in the disclosure debate.

In this proposal Russ starts by stating that the objective is to increase overall security of the net. Also individuals and companies either participate in the forum or are considered members of the black hat community. Although this is a unique statement when you consider other responsible disclosure proposals I think it is an important component. There needs to be a deterrent to irresponsible disclosure. Without civil or criminal laws to punish irresponsible disclosure public black listing may be the only option.

At the center of this forum is a single "Core Group" larger than any of the individual groups that exist today. All new vulnerabilities are initially disclosed to this group. It is the responsibility of the "Core Group" to follow a defined procedure that will garner the trust of the Internet community. The "Core Group" will act as a coordinator and will be responsible for reproduction, coordinating communication, and severity assessment. The main difference with this definition of the "Core Group" acting as a coordinator is the ability of the "Core Group" to place pressure upon a vendor in order to respond to a vulnerability. It is important that a vendor be motivated by forces other than their own internal needs. Remember having any type of "Trusted" group will present a substantial challenge. How will we assure the "Trust" of those within the group, and how will the vulnerability information be handled to assure that it is secured.

Another unique concept is the definition of a "Second Group". This group would receive early warnings with enough information to prepare defenses and detect exploits. However considerable effort would be made to not publish detailed technical information, scripts, or tools that would aid untalented script kiddies in launching attacks. I think this idea of the "Second Group" is the biggest challenge in this proposal. It will require considerable effort to assure that vulnerability information is not disclosed prematurely. In addition black hats may covertly infiltrate the "Second Group" in order to gain early knowledge of new vulnerabilities. Any

enlargement of the circle of trust before a patch has been developed will increase the risk of information leakage.

Finally Russ describes a coordinated public disclosure between the vendor, coordinator, and discoverer. However there is no discussion of the amount of detail that will be included in the final public disclosure. The content of the final public disclosure is an area that requires further definition.

Fisher Plan[12]

In December of 2002 Dennis Fisher with the help of the SANS organization requested input on The Fisher Plan. According to the SANS News Bites e-mail list,

> The plan arose in the days following October 2, 2002, when Richard Clarke told two hundred people attending the SANS/FBI Top Twenty Vulnerabilities briefing in Washington, "Look for vulnerabilities. If you find one, tell the vendors and if they are not responsive, tell the government." Dennis rightfully pointed out that the government is a large organization and connecting with the right person would be nearly impossible.

The Fisher Plan proposes a reporting center that would be responsible for vulnerability reproduction, vendor coordination, determining a deadline for repair based on the severity of the vulnerability, exerting pressure upon vendors to fix vulnerabilities within the set timeline, coordinating a public disclosure, and possibly issuing financial compensation to the discoverer.

Similar to Russ Cooper's "Core Group" the group proposed in the Fisher Plan will face many challenges. As of this writing, work on the Fisher Plan is not yet underway. If you wish to contribute to the Fisher Plan please contact info@sans.org with the subject "Fisher Plan".

Conclusion

Less Vulnerable Software in the Future

It is agreed that vulnerabilities are going to occur but this inevitability is no reason to continue developing products without adequate efforts to eliminate as many flaws as possible. When a customer detects a flaw it costs them money. Customers need to demand better quality products. Vendors need to listen to this demand and take steps to improve quality. Products should ship with only basic features enabled. Internal development staff should be trained on secure programming techniques. Vendors should be committed to quicker and higher quality patch development. Products should do a better job of updating themselves.

To some extent the major players are taking some steps in this direction, but it is up to the customers to demand these steps be taken. In the long run vendors only listen

to one thing, the bottom line. If customers stop buying products because the vendor is not producing quality products the vendor will be forced to change.

Compromise

Vulnerabilities are going to be discovered. The good guys will discover some and the bad guys will discover some. It is in the best interest of those companies and individuals that do not want to be associated with the black hat community to follow a Responsible Disclosure policy. Developing an accepted standard for Responsible Disclosure is going to take a coordinated effort. All parties interested in improving the state of information security are going to have to come together and compromise. We must find a way to address the issues. Vendors must be notified and held to timely patch development. The customer must be given the information they need to defend their systems. Credit and possibly compensation needs to be given to the discoverer. Finally every effort must be made to keep automated attack tools out of the hands of script kiddies. Only by addressing these key issues can we make the Internet more secure.

Resources

Rauch, Jeremy. "The Future of Vulnerability Disclosure?" 8 December 1999. URL: http://www.usenix.org/publications/login/1999-11/features/disclosure.html (January 2003).
Kabay, Mich. "FULL DISCLOSURE." Information Security Magazine. May 2000. URL: http://www.infosecuritymag.com/articles/may00/columns3_logoff.shtml (January 2003).
Schneier, Bruce. "Full Disclosure and the Window of Exposure." Counterpane Crypto-Gram Newsletter. 15 September 2000. URL: http://www.counterpane.com/crypto-gram-0009.html (January 2003).
Davis, Noel. "Sniping at OpenBSD." 9 October 2000. URL: http://rootprompt.org/article.php3?article=1061 (January 2003).
"A Tour of the Microsoft Security Response Center." 2000. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/columns/security/essays/sectour.asp (January 2003).
McClure, Stuart, and Scambray Joel. "Anti-hacking method of full disclosure under attack from a part of the security industry." InfoWorld. 2000. URL: http://iwsun4.infoworld.com/articles/op/xml/00/08/14/000814opswatch.xml (January 2003).
Lasser, Jon. "Keep Security Censorship Away From Linux." 6 November 2001. URL: http://online.securityfocus.com/columnists/35 (January 2003).
Martin, Brian. "Microsoft's Responsible Vulnerability Disclosure, The New Non-Issue." 10 November 2001. URL: http://www.attrition.org/security/rant/z/ms-disclose.html (January 2003).
Graham, Robert. "Full-Disclosure Debate." 2001. URL: http://www.robertgraham.com/diary/disclosure.html (January 2003).
Morgenstern, Michael, Parker, Tom, and Hardy, Scott. "It's Time to be Responsible." 1 March 2002. URL: http://online.securityfocus.com/guest/10711 (January 2003).
Schneier, Bruce. "'Responsible Disclosure' IETF Document." Counterpane Crypto-Gram Newsletter. 15 March 2002. URL: http://www.counterpane.com/crypto-gram-0203.html#2 (January 2003).
Morgenstern, Michael, and Parker, Tom. "The Realities of Disclosure." 12 July 2002. URL: http://online.securityfocus.com/guest/14155 (January 2003).
"@stake Security Vulnerability Reporting Policy." 5 June 2002. URL: http://www.atstake.com/research/policy/index.html (January 2003).
"The CERT Coordination Center Vulnerability Disclosure Policy." 9 October 2000. URL: http://www.kb.cert.org/vuls/html/disclosure (January 2003).
"Microsoft Corporation Product and Service Security Policy." 21 January 2000. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/policy/policy.asp (January 2003).

References

[1] Culp, Scott. "It's Time to End Information Anarchy." October 2001. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/columns/security/essays/noarch.asp (January 2003).
[2] "Internet Security Systems X-Force™ Vulnerability Guidelines." 18 November 2002. URL: http://documents.iss.net/literature/vulnerability_guidelines.pdf (January 2003).
[3] Arbaugh, William, Fithen, William, and McHugh, John. "Windows of Vulnerability: A Case Study Analysis." Published in (IEEE) Computer. December 2000. URL: http://www.cs.umd.edu/~waa/pubs/Windows_of_Vulnerability.pdf (January 2003).
[4] Heiser, Jay. "FULL DISCLOSURE? FULL COMPLICITY! Deconstructing the myths behind the full-disclosure debate." Information Security Magazine. January 2001. URL: http://www.infosecuritymag.com/articles/january01/columns_curmudgeons_corner.shtml (January 2003).
[5] "University of Oulu Secure Programming Group (OUSPG)." Ver. 1.21. 16 July 2001. URL: http://www.ee.oulu.fi/research/ouspg/index.html (January 2003).
[6] Laakso, Marko, Takanen, Ari, and Röning, Juha. "The Vulnerability Process: a tiger team approach to resolving vulnerability cases." Presented at FIRST'1999. 18 June 1999. URL: http://www.ee.oulu.fi/research/ouspg/protos/sota/FIRST1999-process/paper.pdf (January 2003).
[7] Laakso, Marko, Takanen, Ari, and Röning, Juha. "Introducing constructive vulnerability disclosures." Presented at FIRST'2001. 22 June 2001. URL: http://www.ee.oulu.fi/research/ouspg/protos/sota/FIRST2001-disclosures/paper.pdf (January 2003).
[8] Cooper, Russ. "NTBugtraq Disclosure Policy." 26 July 1999. URL: http://www.ntbugtraq.com/default.asp?sid=1&pid=47&aid=48 (January 2003).
[9] Puppy, Rain Forest. "Full Disclosure Policy (RFPolicy)." Ver. 2.0. 1999. URL: http://www.wiretrip.net/rfp/policy.html (January 2003).
[10] Christey, Steve, and Wysopal, Chris. "Responsible Vulnerability Disclosure Process." IETF Draft. February 2002. URL: IETF draft expired in August 2002 and is only available via e-mail request to the authors (January 2003).
[11] Cooper, Russ. "The Responsible Disclosure Forum." 8 November 2001. URL: http://www.ntbugtraq.com/default.asp?sid=1&pid=47&aid=66 (January 2003).
[12] Fisher, Dennis. "The Fisher Plan". SANS NewsBites 12 December 2002 Vol. 4, Num. 502. URL: http://www.sans.org/newsletters/newsbites/vol4_50.php (January 2003).