nCircle Vulnerability Scoring System

1. Vulnerability and Risk Analysis

Measuring and managing the security risk associated with information and information technology remains one of the most challenging and elusive problems faced by all levels of an organization. The challenge of how to measure, and therefore how to manage, risk is ever-present and top of mind for information security professionals. Unfortunately, most of the tools for vulnerability and risk management that exist today do not provide a suitable metric and consequently do not improve an organization's ability to cost effectively manage risk. Traditional vulnerability management tools deliver a list of detected conditions, ranked in a coarse and subjective manner (e.g. low, medium, high or 1 through 5), and from this list information security decisions must be made. Such subjective ranking simply does not provide a means to effectively prioritize across thousands, let alone tens of thousands of hosts. The growing number of vulnerabilities that exist means that, as the list increases in length, so does the importance of prioritization. The ability to effectively score vulnerability conditions in a meaningful, repeatable, and objective manner allows organizations to understand their infrastructure and focus on areas where their limited resources can be most effective in lowering the overall risks to the business. A viable risk metric is a required foundation for any information security practice.

1.1. Background

nCircle developed an objective scoring formula in early 2001. The formula used in the actual risk calculation has not changed since then, and has been utilized as a vulnerability scoring metric within a number of large organizations that have adopted nCircle IP360™ as a vulnerability and risk management standard. It's worth noting that although the product itself has changed considerably over the years, the nCircle IP360 score has remained the most relevant, usable, and functional risk metric in the industry.

The IP360 score provides a viable and tested model of risk assessment. This paper explains in detail the thought behind the metric, as well as the actual formula used for calculating IP360 vulnerability scores. The model is presented as an "open-source" method of risk analysis; anyone who finds the method useful is encouraged to utilize it. There are still many other methods of risk analysis available. This paper argues that other common industry models of "risk assessment" are inherently ambiguous and ultimately not functional in real world situations.

This paper:

1. Details the most commonly used method of assessing risks and vulnerabilities.
2. Discusses the shortcomings of many common vulnerability "scoring" models.
3. Presents the IP360 vulnerability score: an alternative methodology that addresses the limitations in the vulnerability scoring techniques in other software and service offerings.

1.2. Details and Definitions

Vulnerability management and vulnerability scoring do not exist in a vacuum. In order to provide a cohesive and comprehensible paper, this section aims to set a common vocabulary and context.

Vulnerability Management: Vulnerability management is the process of assessing the existence and severity of vulnerability conditions within an organization, including the workflow and process for making mitigation decisions about the vulnerabilities.

Vulnerability Scoring: Vulnerability scoring is the process or method for describing the risk that a specific vulnerability presents.

Vulnerability management is a component of risk management, which encompasses the areas of disaster recovery, business continuity, policy, and physical security. Vulnerability scoring, in turn, is a tool used to make vulnerability management more effective. In the domain of information security, with which this paper is most concerned, all of these practices deal explicitly with the protection of data.

Data refers to the information stored on and passed between the hosts within an organization. *Data*, so understood, encompasses not only the information passed from machine to machine, but also any information *about* the network's structure, composition, or configuration.

From an information security standpoint, security objectives are applied to information and information systems that have been categorized. (800-53 - FIPS 199). These security objectives are most commonly discussed as *confidentiality*, *integrity*, and *availability*. The loss of any one of these characteristics constitutes an incident. With this in mind, threats to network security fall into three general classes:

- Threats to data *confidentiality*
- Threats to data *integrity*
- Threats to data *availability*

1.2.1. Definition of Risk and Vulnerability

A vulnerability is some aspect of a network resource's functioning, configuration, or architecture that makes the resource a target of potential misuse, exploitation, or denial of service, e.g. the realization of a threat to confidentiality, integrity, or availability. In other words, a vulnerability is an *opportunity for threat to be realized*. Vulnerabilities in a system can be attributed to many factors, which include, but are not limited to:

- Software bugs
- System architecture flaws
- Weaknesses in user access control
- System configuration
- Information the network resources make available to users
- Physical organization of a network

Risk, then, is the *potential* that the threat will be realized for a particular vulnerability. The relationship of vulnerability, threat, risk, and exploit is important to understand. These are terms that often get misused, and whose definitions have changed over time.

1.2.2. Vulnerability Analysis

Vulnerability analysis involves the systematic detection of vulnerabilities in network resources. This is distinct from the process of vulnerability research, which involves the discovery and documentation of vulnerability conditions. It is also distinct from the process of vulnerability management, which addresses the larger process of reacting to vulnerability analysis. Vulnerabilities can exist at multiple layers of the network infrastructure. It's important to keep in mind the common levels and their differences, as the analytics apply in different ways at different levels.

Endpoint Conditions: Analysis of endpoint vulnerability conditions involves determining whether settings on the hardware, the configuration of an operating system, or flaws or limitations in the software produce vulnerabilities on a specific network resource. Buffer overflows in FTP services and weaknesses in user authentication services are common examples of endpoint conditions.

Network Conditions: The context in which an endpoint functions in a network environment further defines its risk level. Improperly configured access control, routing conditions, and network points of failure constitute network vulnerability conditions.

Often, these two types of conditions are analyzed separately, though the analysis of either one clearly has bearing on the analysis of the other. A major reason for this analysis gap is the lack of a common, usable metric for measuring risk.

Any organization's resources are limited, yet the number of vulnerabilities discovered is generally too numerous for complete remediation to be achieved. Given a network of ten thousand hosts, with a conservative average of ten to twenty vulnerabilities each, the full list of conditions would easily reach into the hundreds of thousands. Clearly, it is not sufficient merely to catalogue the risks to the network. The organization must have a set of criteria by which to categorize the discovered conditions and aid in making effective risk mitigation decisions. These requirements implicitly involve a model of risk analysis specific to the condition of network resources, of which several have been developed. The vulnerability score, then, is a risk metric or categorization applied to a specific vulnerability.

1.3. Current Models of Vulnerability Scoring

The process of assessing a host for vulnerabilities and reporting on the data found is not new. There are a number of established means of ranking vulnerabilities, both prior to and after discovery on a host:

Method 1: High/Medium/Low

In 1999, the National Infrastructure Protection Center (NIPC) began publishing CyberNotes, a monthly security update that includes a framework for "risk analysis." The NIPC's method for classifying risks is noteworthy because prior to 1999 there has been little effort to establish uniform industry standards, or "best practices" for risk diagnostics.

According to their classification scheme, all attacks against a network fall into one of three risk categories: Low, Medium, or High.

Low Risks: The NIPC defines a low risk as "A vulnerability that provides information to an intruder that could lead to further compromise attempts or a Denial-of-Service attack." Generally, low risk conditions do not compromise the system beyond a Denial-of-Service. This type of condition is often inherent in running a particular service.

The definition, however, does include the following caveat: "... while the DoS attack is deemed low from a threat potential, the frequency of this type of attack is very high. DoS attacks against mission-critical nodes are not included in this rating, and any attack of this nature should instead be considered as a 'High' Threat." Terry Escamillia, in *Intrusion Detection: Network Security Beyond the Firewall* categorizes both local and remote Denial-of- Service attacks as "annoying."

Moderate Risks: Vulnerabilities that allow local or remote users to increase their privileges on a system or access confidential information such as company financial records or user passwords are usually considered moderate risks. According to the NIPC, "Any vulnerability that will allow an intruder immediate access to the system that is not privileged access" is a medium risk.

High Risks: Any vulnerability that could potentially allow a user to gain privileged access to a system is almost always regarded as a "high risk." According to the NIPC, a high risk is "A vulnerability that will allow an intruder to immediately gain privileged access (e.g. Administrator, root) to the system.

This model from NIPC provided the genesis of many vulnerability scoring methodologies still seen in products today. There are variations on the terms, of course (critical, informational, severe, etc), but the concepts are largely similar. All of these variations rely on a **depth of access principle**. Vulnerabilities are ranked based on the depth of access produced by exploit. The implicit rule is that greater depth of access means greater risk.

Method 2: Numerical Ranking

There are some clear disadvantages to the "High/Medium/Low" method of ranking vulnerabilities, and some products have made changes to adopt a numerical system of ranking conditions. There are two primary reasons for making this change:

- Meaningful aggregation of vulnerability rankings
- Use of *n* number of vulnerability rankings

The previous method, being a text-based category, does not allow for aggregation of vulnerability rankings. By using numbers instead of text categories, tools can provide aggregation functionality.

The aggregation is secondary, however, to the ability to utilize a greater quantity of rankings. It became apparent fairly quickly that three categories (high/medium/low) don't adequately cover the existing vulnerabilities. Rather than attempt to find more words to insert into the rankings, using numbers allows for easy expansion to, for example, five categories instead of three.

A current example of this model appears in the Payment Card Industry standard. PCI is a standard for information security developed by a consortium of credit card companies. It is not used to secure those companies internally, but designed to provide an external standard that can be applied to merchants and issuing banks. The PCI standard defines five levels of vulnerability:

Level 5 (Urgent): Level 5 vulnerabilities provide remote intruders with remote root or remote administrator capabilities. With this level of vulnerability, hackers can compromise the entire host. Level 5 includes vulnerabilities that provide remote hackers full file-system read and write capabilities, remote execute of commands as a root or administrator user. The presence of backdoors and Trojans qualify as level 5 vulnerabilities.

Level 4 (Critical): Level 4 vulnerabilities provide intruders with remote user, but not remote administrator or root user capabilities. Level 4 vulnerabilities give hackers partial access to file-systems (for example, full read access without full write access). Vulnerabilities that expose highly sensitive information also qualify as level 4 vulnerabilities.

Level 3 (High): Level 3 vulnerabilities provide hackers with access to specific information stored on the host, including security settings. This level of vulnerabilities could result in potential misuse of the host by intruders. Examples of level 3 vulnerabilities include partial disclosure of file contents, access to certain files on the host, directory browsing, disclosure of filtering rules and security mechanisms, susceptibility to denial of service (DoS) attacks, and unauthorized user of services such as mail relaying.

Level 2 (Medium): Level 2 vulnerabilities expose some sensitive information from the host, such as precise versions of services. With this information, hackers could research potential attacks against the host.

Level 1 (Low): Level 1 vulnerabilities expose information such as open ports.

The PCI standard expands upon the **depth of access** methodology of the high/medium/low model. The descriptions of each level are verbose, and strive to leave little room for interpretation. In this model, one finds a few additional concepts beyond simple **depth of access**, including **quality of information**, and **class of system/information**. The **quality of information** concept is included in the varying levels of information disclosure described. The difference between levels 1, 2, and 3 is described partially be the quality of the information exposed, e.g. “filtering rules” vs. “open ports.” The concept of **class of system/information** is surfaced by the references to specific systems, e.g. “mail relaying,” but is still not a complete component of the ranking.

Method 3: Common Vulnerability Scoring System

In 2004 the National Infrastructure Advisory Council introduced the Common Vulnerability Scoring system in a paper. CVSS represents the most extensive effort to date aimed at providing a common method of scoring vulnerability conditions. CVSS is a well-documented standard, and this paper will not address the intricacies of its implementation, but the model proposed instead. CVSS scores vulnerability conditions on a 1-10 scale, using a combination of three scores which are, in turn, generated from a set of selectable criteria. Exploring the individual criteria is not necessary in order to understand the model. CVSS employs a *base score, temporal score,* and *environmental score* in order to generate the overall CVSS score for a vulnerability.

Base Score: These metrics describe inherent characteristics of the vulnerability, such as access vector, access complexity, and authentication requirements.

Temporal Score: These metrics describe elements about the vulnerability that change over time, such as availability of exploit and type of fix available.

Environmental Score: These metrics describe the effect of a vulnerability within an organization's environment, such as damage potential and target distribution.

CVSS makes up for a lot of the disadvantages of the previous two models, incorporating a significant number of additional criteria and defining them clearly. Additionally, CVSS has the benefit of being sponsored by the US government and developed by a working group of industry professionals. Adoption of a common method of scoring vulnerabilities is imperative if organizations as whole are going to address risk in a sustainable way. That being said, CVSS is not without its disadvantages.

1.4. Limitations of Current Models

All of the above models have limitations that make them less than perfect for addressing the risk management needs of the enterprise. These limitations are often common across the models and can be explained as such. The three major limitations of the current models can be described as subjectivity, ambiguity, and inaccuracy.

Limitation One: Subjectivity

In order for a risk analysis metric or ranking to be used in any common way, the methodology for applying the rankings to vulnerability conditions must be objective. The high/medium/low model clearly fails to provide an objective methodology. There is nothing inherent in the term ‘high’ to define to which conditions that category might apply. An external definition must be applied.

The numeric model described previously suffers from the same limitation, though it is somewhat masked by the use of a number instead of a text-based label. A number as category, however, contains no inherent definition of its contents either.

Subjectivity creates a significant problem in implementation. The value of these risk assessment models increases in proportion to the number of vulnerabilities that have been classified by the method in a customer's environment. As customers try to make sense of the vulnerability information reported to them, the requirement to use the vulnerability rankings systematically increases in importance. The subjective nature of these models makes that difficult.

Limitation Two: Ranking vs. Metric

There is an important distinction between a ranking and a true metric. A ranking provides only a relative distinction between conditions. Its accuracy decreases in proportion to the number of conditions being ranked, or alternatively, the number of available rankings must be increased. A metric provides an atomic measurement of a condition, regardless of the other conditions that exist. In other words, a ranking is meaningful only relative to the ranking of other conditions, whereas a metric is meaningful in isolation or amidst *n* number of conditions.

The importance of this distinction is directly related to the number of vulnerabilities that exist and to the number of organizations performing vulnerability assessments. If an assessment of ten hosts is performed, the pool of possible vulnerabilities is fairly small, so distinguishing between hosts is possible with rankings. If, however, an assessment contains thousands of hosts, each with tens of vulnerability conditions, then the ability to distinguish conditions and their risk based on a ranking becomes challenging without a dramatic increase in the number of available rankings.

The move from text-based categories to numeric rankings may appear to address this limitation through aggregation, but it does not. There is no more repeatable or objective logic behind the numeric ranking than the text-based categories. The input data that produces the numbers suffers from the same underlying limitation of subjectivity, making the result of any aggregation ultimately subjective as well.

This is a limitation that applies to the first two methods, but not to CVSS. CVSS makes the move to deliver a metric, rather than a ranking. Ultimately, this metric has its own limitations. The components of the metric still suffer from some subjectivity problems. More importantly, however, the metric delivers a finite scale of 1-10, which inherits some of the problems with relative values that traditional rankings also have. These issues have surfaced when different organizations produce different CVSS scores for the same condition.

Limitation Three: Contextual Relevance

There are actually two distinct limitations in the sense of contextual relevance for the example methods.

Both the high/medium/low method and the numeric ranking method suffer from the fact that they do not account for the context of time. A condition, once labeled, does not change. This means that as a vulnerability ages and the exploit techniques become more widely distributed, the vulnerability ranking remains the same, although the risk that vulnerability presents does not.

CVSS suffers from a contextual relevance issue as well, though in a completed different sense. CVSS addresses the risk over time context with the *Temporal Score*. This score is produced by combining rankings for exploit availability, fix status and confidence. As a vulnerability progresses through its lifecycle from theoretical w/ no patch to confirmed with an official patch, the risk that vulnerability presents decreases. To be clear, this is a limitation only in the usage of CVSS. CVSS has been designed as a common metric to measure the risk that a vulnerability presents to

the overall user base, rather than to measure the risk that an instance of a vulnerability condition presents to a specific organization. It is true that after an official patch is available, the overall risk of a vulnerability decreases, but if a specific organization fails to apply that patch, the risk *that* vulnerability *instance* presents to that organization *increases*. The contextual relevance of CVSS differs from that of an operational metric.

Where the ranking methods ignore the time parameter, CVSS and nCircle view the time parameter in different relationships to the measurement target, i.e. a vulnerability instance versus a vulnerability itself.

2. nCircle Vulnerability Scoring System

2.1. Context of Vulnerability Scoring

Vulnerability scoring does not equate completely with security. In fact, there is no “formula” for security, no *a priori* method for determining if a company's security is tight enough to keep criminals out. The vulnerability score itself does not account for the context in which that vulnerability has been discovered. nCircle IP360 also uses host “Asset Values” to provide business context to the vulnerability scores. Asset Values are provided by the customer and are integers (typically dollar values) that denote the value of a particular host in the enterprise. Representing the value of the asset is an important component of prioritization; if a host has a high vulnerability score but a low asset value, the security administrator may choose not to focus on it, whereas if its asset value was high, it would clearly be a priority.

Additionally, nCircle solutions such as the nCircle Topology Risk Analyzer™ and nCircle nTellect™ build on the foundation of vulnerability management by adding further context to the vulnerability score. These solutions are able to provide meaningful metrics because of the method used to calculate the IP360 vulnerability score. A valid metric can be extended in this way.

The IP360 vulnerability score is, ultimately, a mathematical abstraction based on the results of an assessment. The results of IP360 assessments are thoroughly described in the reports available through the VnE Manager, including recommendations for how site security can be improved and risks to the network minimized. Having acknowledged these considerations, it is now important to discuss the details of how vulnerabilities in the customer network are identified and scored.

2.2. Heuristic Approach to Estimating the Penetrability of a Network

Each vulnerability in a system or network is associated with a specific “risk” value, but this value should not be thought of as an absolute measurement of the threat, which the vulnerability, if left unchecked, poses to the network. This “risk value” changes over time based on factors that are entirely independent of the system or network that exhibits the vulnerability. When interpreting the vulnerability score of a network, there are two very important considerations to keep in mind. Both considerations have to do with the vulnerability score being a heuristic measurement rather than an absolute metric that is not subject to change.

2.3. The Vulnerability Scoring Equation

The IP360 vulnerability score has been developed to address concerns inherent in existing vulnerability rating systems. The model, its mathematical structure and variables, were developed over several years using data collected from thousands of security audits. The primary components of the vulnerability score for a condition (*n*) are:

- $\mathbf{t}_n$: The number of days that have elapsed since information concerning vulnerability *n* was first made available via major security sources.
- $\mathbf{r}_n$: The "class risk" factor, which represents the threat inherent in having vulnerability *n* on a system *s*
- $\mathbf{s}_n$: A measurement of the "skill set" required to successfully carry out an attack, which exploits vulnerability *n*.

Let V_n represent the vulnerability score, which is calculated in the following manner:

$$V_n = \sqrt{t_n} \times \frac{r_n!}{s_n^2}$$

2.4. Analysis of the Vulnerability Score

This section examines how numerical values are assigned to each of the variables employed in the vulnerability score formula. Lastly, a few comments will be offered concerning the formula itself.

2.4.1. A Time-based Approach to Vulnerabilities

The variable *t* in the "vulnerability score" formula represents the amount of time that information concerning a vulnerability has been available to the public from major security sources. These sources may change over time, but the concept of public availability remains consistent. One can consider such sources as CERT Advisories, vendor alerts, mailing lists or news feeds as current examples of such sources. To calculate the value for *t* for a given vulnerability, simply determine how many days have elapsed since news of the vulnerability was first published in an advisory or posted to a discussion group.

CVE ID	**Date Posted**	**Current date**	**t**	$\sqrt{t}$
CVE-2005-1983	8/9/2005	4/1/2006	235days	15.3

2.4.2. A Risk-based Approach to Vulnerabilities

Based on the nCircle scoring system, it is very easy to determine the "risk" for an vulnerability. First, the "class" of the vulnerability must be determined. The nCircle risk model uses a system of 6 "risk classes" to categorize vulnerabilities (in the order of increasing severity):

1) Local attacks against resource availability (e.g. various local DoS attacks)
2) Local methods for increasing user privileges
3) Local methods for obtaining complete administrative privileges
4) Remote attacks against resource availability
5) Remote methods for increasing user privileges
6) Remote methods for obtaining complete administrative privileges

Calculating the "base risk" for the exploit involves inserting the highest applicable "risk class" value for a vulnerability into the formula above. Here is an example using CVE-2005-1983 from MS05-039 :

Exploit	**Type\Strategy\Impact**	**Class**	**Risk (r)** *r!*
CVE-2005-1983	Remote\BufferOverflow\Root	6	720

2.4.3. Understanding Skill and the Vulnerability Score

At first glance, measuring or determining the "skill" prerequisites for performing various kinds of attacks presents a number of difficulties. Even the very idea of numerically quantifying skill levels is nebulous at best, and almost any numerical scheme one could use to represent the degree of difficulty associated with effectively exploiting a vulnerability can be criticized as being uninformative and arbitrary.

The nCircle model avoids these difficulties by using a "tool oriented" method of quantifying how difficult it is to perform certain attacks. The vulnerabilities that require the least skill to exploit are those for which there exist sophisticated applications that do all of the hard work for the user: the user is able to install the program, pull up a graphical-user interface, then point, click and root! On the opposite side of the skill-spectrum, vulnerabilities that require the greatest skill are those that are highly "theoretical." Occasionally, an exploit is referenced in a public newsgroup or advisory but there is no publicly available source code, scripts, or binaries that could be used to automate or facilitate an effective attack on the vulnerability. To effectively exploit this vulnerability requires advanced knowledge, patience, research, and genuine innovation.

The following table describes the nCircle IP360 classification scheme for skill in greater detail:

Skill Label	Description of the Tools Required	Skill Level (S)	S^2
Automated Exploit	A graphical application that includes an installer, or no human interaction required, e.g. a network worm	1	1
Easy	A non-UNIX binary application, typically containing an installation script, batch file, or other simple installation mechanism. A binary is a precompiled exploit that does not require specific operating system or networking knowledge.	2	4
Moderate	A non-Windows binary application, typically a binary containing an installation script, batch file, or other simple installation mechanism. A binary is a precompiled exploit that does require operating system or networking knowledge.	3	9
Difficult	A non-Windows shell, perl, or interpreted script program that requires limited knowledge of operating systems, shell code, interpreters or networking.	4	16
Extremely Difficult	An un-compiled set of source files, typically compressed in some way that requires specific knowledge of operating systems, compilers, and advanced system experience.	5	25
No Known Exploit	Typically, this category describes an exploit or that has been referenced in a public forum or advisory and does not include source code, an exploit script, or a reference to predefined exploit source.	6	36

For the MS05-039 example from above, there is exploit source code easily available. The value of S^2, therefore, is calculated as follows:

Type of Tool Available	Class	s^2
Automated Exploit	1	1

2.4.4. Calculating a Sample Vulnerability Score

By fitting the values derived in sections 2.4.1, 2.4.2, and 2.4.3 into the vulnerability score formula, the result is a vulnerability value for the vulnerability under consideration as of the date 4/1/2006.

$$11016 = 15.3 \times \frac{720}{1}$$

It is important to consider that the nCircle Vulnerability Score provides a metric for a vulnerability at a point in time. In practice, this point in time is almost always *now*, but the metric can be used to predict risk increases. For example, in an additional 365 days, the MS05-039 condition will score like this:

$$17640 = 24.5 \times \frac{720}{1}$$

3. Extending Vulnerability Scoring

3.1. Calculating the Vulnerability Score of a Host

To this point, the vulnerability score has been applied to a condition. While the condition may be the atomic unit in vulnerability management, enterprises do not think of their environments as collections of conditions. Hosts and their applications are also valid targets for vulnerability assessment and scoring.

The vulnerability score for a host on the network (e.g. a firewall, an individual computer, a router, etc.) is the sum of the risk values (i.e. vulnerability scores) for each of the vulnerabilities discovered on that host. MS05-039 preceded MS05-041. A host exhibiting CVE-2005-1983 is also likely to exhibit CVE-2005-2303. Calculating the vulnerability score for that host, then, is a matter of summing up all of the individual vulnerability scores for the conditions discovered on that host. To calculate the vulnerability score for the host *S*, the formula presented below is used, where V_1 is the first vulnerability discovered in *S* and V_n is the last:

$$V_1 + \ldots V_n = V_s$$

It is useful to calculate the combined vulnerability score for a single host for multiple reasons. First, examining the vulnerability scores of various network resources provides a basis for making effective remediation decisions based on host, rather than vulnerability. The ability to prioritize hosts is important in the enterprise. Additionally, the metric can be used as a means of comparison and for change detection. Hosts that have the same configuration and function should produce the same score. IP360 displays several views of vulnerability information, including per resource and consolidated.

3.2. Calculating the Vulnerability Score of a Network

IP360 represents IP space as collections of user defined networks. It is trivial, given the explanation for a host vulnerability score, to produce a network vulnerability score in the same manner. The network score is the sum of all of the vulnerability scores from each of the network systems, where S_n is the value of the combined scores of the individual vulnerabilities discovered in a resource S, as discussed in section 3.1.

The network score is calculated in the following way:

$$S_{n1} + S_{n2} + S_{n3} \ldots = V_n$$

The network score can be used, further, for trending risk posture over time. In a large enterprise, the ability to effectively track and trend risk posture is invaluable.

4. Logical Consequences of the nCircle Scoring System

Any metric applied consistently to an environment has consequences. The act of measuring in a consistent manner produces changes in behavior. The behavioral changes vary based on what that metric actually measures, of course. Thus, what you measure directly affects what change eventually occurs. An inappropriate method of measuring risk will result in equally inappropriate actions to reduce risk. Only by applying a valid metric can reasonable risk reduction actions be taken.

This paper demonstrates that the nCircle IP360 vulnerability score provides a valid and relevant base metric for measuring IT risk in the corporate environment. Adoption of the vulnerability score as a foundational metric for measuring risk in the enterprise, in conjunction with host Asset Values and other nCircle solutions such as the nCircle Topology Risk Analyzer, enables organizations to more effectively:

- **Measure** network security risk using objective metrics
- **Manage** network security risk through dashboard reporting and integration with existing enterprise systems
- **Reduce** network security risk by focusing IT resources on the highest priority risks

About nCircle

nCircle is the leading provider of enterprise-class vulnerability and risk management solutions. Global enterprises and government agencies rely on nCircle's proactive security solutions to identify, measure, manage, and reduce security risk on their worldwide networks. nCircle has won numerous industry awards for innovation and technology leadership and has been named to the top 100 Best Places to Work in the San Francisco Bay Area. nCircle is headquartered in San Francisco, CA, with regional offices throughout the USA and in London, Toronto and Tokyo. Additional information about nCircle is available at www.ncircle.com.

nCircle Network Security
101 2nd St. Suite 400
San Francisco, CA 94105
(888) 464 2900
www.ncircle.com