Interview mit Renaud Deraison – Initiator des Nessus-Projekts

Marc Ruef, scip AG, maru-at-scip.ch
Renaud Deraison, deraison-at-nessus.org

scip AG: Hallo Renaud, vielen Dank für Deine Zeit. Da in Bälde der neue Major-Release 3.0 von Nessus erscheint, werdet Ihr Jungs wohl einiges zu tun haben. Wie waren Eure letzten paar Wochen? Seid Ihr nervös?

Renaud Deraison: Die Dinge entwickeln sich gut, danke. Wir haben soeben die Pakete für die jeweiligen Betriebssysteme, die wir allesamt mit dem jüngsten Release unterstützen wollen, fertiggestellt. Unser Team der Qualitätssicherung testet diese gerade durch.

Zur aktuellen Stunde bin ich in höchstem Masse zufrieden mit dem Status - Es gibt jedoch immer ein paar Dinge, die sich nach einer Herausgabe als total „schräg“ herausstellen. Wir sind trotzdem sehr zuversichtlich und freuen uns darauf, der Welt Nessus 3 vorstellen zu können.

Die Anpassung der Lizenzbedingungen der Nessus-Plugins im Dezember 2004 und die Ankündigung, dass Nessus 3.0 nicht mehr open-source sein wird, sind grosse Änderungen im Projekt. Diese bringen sowohl Vor- als auch Nachteile mit sich. Was waren Eure Gründe für eine solche Entwicklung? Welche Hoffnungen und Ängste bringt für Euch die neue Zukunft von Nessus?

Unser Ziel mit Nessus ist es, die beste Lösung für ein möglichst breites Publikum bereitzustellen. Um dieses Ziel zu erreichen, haben wir in den letzten zwölf Monaten einige Grundlagen geschaffen:

(1) Wir haben einen Dienst gestartet, der sich Direct Feed nennt. Dieser gewährt abonnierten Nutzern die Möglichkeit, neue Sicherheitsüberprüfungen sofort nach unserer Implementierung der Plugins umzusetzen. So schnell wie möglich, nachdem ein Plugin unser Qualitätsmanagement durchlaufen hat, wird es diesen Anwendern bereitgestellt. Alle User, die nicht bereit sind für diesen Service zu zahlen, können auf eine abgespeckte Version, bei der die Plugins jeweils sieben Tage später zugänglich gemacht werden, zurückgegriffen werden. Dies ist noch immer weitaus kürzer, als so manches andere konkurrierende Scanner-Produkt.

(2) Noch wichtiger war, dass wir eine neue Version von Nessus (Nessus 3) angekündigt haben. Diese wird unentgeldlich erhältlich sein, jedoch nicht mehr mit offenem Quelltext veröffentlicht. Der Grund dafür ist, dass open-source Software noch immer in vielen Firmen und behördlichen Organisationen verboten ist. Viele der Nutzer wollen sodann auf ein Projekt zurückgreifen, das nicht den Vorgaben entspricht. Durch den Wechsel zu einer proprietären Aufmachung können sie Nessus offiziell einsetzen und damit mehr erreichen. Wir haben zusätzlich entschieden, Nessus leicht zeitverzögert ebenfalls unter diesem Lizenzmodell für Windows- und MacOS X-Plattformen herauszugeben.

(3) Endlich haben wir eingeführt, dass Abonnenten des Direct Feed einen Email-Support für Nessus 3 erhalten. Dies gilt sowohl für Plugins zur Umsetzung lokaler Audits als auch für solche in Netzwerkumgebungen. Dies ist besonders komfortabel für Consultants, die einen Audit nach Sarbanes-Oxley/FISMA machen wollen oder für Teams, die einer Überprüfung unterzogen werden, um sich leichter mit der Security Policy zu argumentieren.

„Unser Ziel mit Nessus ist es, die beste Lösung für ein möglichst breites Publikum bereitzustellen.“

Der Grund für diese Anspassungen ist, dass wir Nessus als vollumfängliches kommerziell unterstütztes Produkt, das nicht nur von irgendwelchen Hobby-Programmierern entwickelt wurde, etablieren wollen. Wir haben eine solide Entwicklung Richtung Professionalität in den letzten Jahren an den Tag gelegt, wenn es um die Umsetzung neuer Plugins geht. Und entsprechend wünschen wir, dass dies von der Branche so aufgefasst wird.

Wir hoffen, dass diese Entscheidung die Anzahl Nutzer erhöhen wird. Alleine schon deswegen, weil die Version 3 auf einer Mehrzahl an Plattformen angeboten wird und die Installation noch einfacher ausfällt, da auf die plattformspezifischen Pakete zurückgegriffen wird. Da das neue Release nun kommerzielle Unterstützung findet, kann es neu auch in Bereichen eingesetzt werden, die zuvor unzugänglich waren, was zu einer grösseren Community führen wird.

Drehen wir das Rad der Zeit einige Jahre zurück, zur Geburtsstunde des Nessus Projekts. Was waren Deine Beweggründe für dieses? Und welche Lösungen waren für

Dich eine Inspiration?

Die Arbeit mit Nessus habe ich Ende 1997 begonnen und den Initial-Release im Jahr 1998 umgesetzt. Damals orientierte ich mich an SATAN, jedoch mehr aus Benutzer- denn aus Entwicklersicht. SATAN war schon zu dieser Zeit überholt und ich hatte erst kürzlich von der Programmierung auf MacOS 7-Systemen zu Unix gewechselt. Dabei habe ich die neue Einfachheit erst gerade kennengelernt und mit einer Software, die automatisch eine Umgebung auf Sicherheitslücken hin überprüfen können sollte, ein interessantes Projekt in der Richtung gefunden. Meine Arbeit stellte ich Mitte 1998 vor und erhielt prompt eine grosse Anzahl an Feedback, so dass ich meine Bestrebungen seitdem stets vorantrieb.

Hast Du Dir jemals kommerzielle Vulnerability Scanner wie den ISS Internet Scanner oder Symantec NetRecon angeschaut? In welchen Punkten wäre Nessus die bessere Wahl und in welchen Bereichen hätte Nessus mit Nachteilen zu kämpfen?

Ich bin nicht gerade die beste Person zur beantwortung dieser Frage (*lacht*). Die Stärke von Nessus liegt eindeutig in der skript-basierten Architektur. Durch das Implementieren der einzelnen Protokolle in eine proprietäre Skript-Sprache schaffen wir uns zwar zusäzlichen Aufwand, welcher das Endprodukt jedoch leichter zu warten und stabiler ausfällen lässt. Da wir alle Bibliotheken in interpretierten Sprachen vorliegen haben, können wir uns umfassend vor Pufferüberlauf-Zugriffen, Null Pointer Referenzierungen und anderen Programmierfehlern schützen, welche ansonsten zu Hauf in C/C++ anzutreffen sind. Dies erlaubt uns ein Mehr an Flexibilität, da wir nicht von dem abhängig sind, was das Betriebssystem zu senden gedenkt. Dies ist ein Vorteil, der nur die wenigsten Scanner aufzuweisen haben.

Es sind einige Forks der nach wie vor als open-source veröffentlichten Version 2.x von Nessus angekündigt. Was denkst Du über diese? Werden sie durch Euch als vollwertige Mitbewerber wahrgenommen? Und wie wollt ihr mit diesen konkurrieren, denn bekannte open-source Lösungen werden erfahrungsgemäss öfters aktualisiert weder Projekte mit geschlossener Entwicklergruppe?

Es freut mich wirklich zu sehen, dass es Leute gibt, die ihre Zeit und Energie in die Grundlage von Nessus investieren wollen - Obschon ich gestehen muss, dass ich es bereue, dass diese Unterstützung nicht schon zuvor angelaufen ist. In den letzten sieben Jahren waren wir stets um Mithilfe dankbar, wohl gerade deswegen, weil wir nicht viel von dieser erfahren haben.

Grundsätzlich bedeutet zusätzliche Konkurrenz, dass die Endanwender zusätzliche Auswahl haben. Dies ist eine grossartige Sache, solange die Mitbewerber geistiges Eigentum als solches wahrnehmen. Falls diese lediglich ihre Zeit nutzen, um bestehende Nessus-Elemente zu kopieren, den Namen zu ändern und ihre Copyright-Bestimmungen anzugeben, wäre ich sehr enttäuscht. Zur aktuellen Stunde hat noch kein Fork-Projekt eine ihrer überarbeiteten Versionen publiziert, so dass das Fällen einer Aussage eigentlich noch gar nicht möglich ist.

In Anbetracht der geringen Anzahl der Releases gilt es zu verstehen, dass die Nessus-Engine ganz klar von den Plugins getrennt ist. Nessus 2.x wurde beispielsweise jeweils mit stabilen Minor-Releases im Zyklus von drei Monaten umgesetzt. Zu bemerken ist, dass sämtliche Netzwerk-Operationen, die von Nessus umgesetzt werden, durch die jeweiligen Plugins gehandhabt werden - Diese implementieren verschiedene Protokolle (z.B. NFS, SNMP, SMB, FTP, usw.) und die Engine ansich ist nur dafür zuständig, die Zugriffe richtig zu koordinieren. Möchten wir zusätzliche Funktionalität einbringen, müssen wir lediglich einige Plugins abändern, anstatt den ganzen Core zu überarbeiten. Ich denke, dass eine niedrige Frequenz an neuen Releases eine feine Sache ist - Dies zeugt nämlich davon, dass die Engine sehr solid ist. Im Gegenzug sind wir sehr um Aktualität in Bezug auf die Plugins bedacht und werden dies auch in Zukunft so halten.

„Es freut mich wirklich zu sehen, dass es Leute gibt, die ihre Zeit und Energie in Nessus investieren wollen."

Mit anderen Worten erlaubt uns die gegenwärtige Architektur des Projekts ein umfassendes Update umzusetzen, ohne dass die Endanwender auf zeitintensive Upgrades der Engine zurückgreifen müssen.

Zudem haben wir eine sehr grosse Benutzerunterstützung - In Gegenüberstellung zu vergleichbaren Vulnerability Scannern ist unsere bei weitem die beste. Diese wird voraussichtlich mit Nessus 3 im kommenden Jahr noch mehr im

Bereich der Windows- und MacOS X-Anwender für sich gewinnen können. Wir haben eine gute Partnerschaft mit unseren Anwendern, wobei uns deren Rückmeldungen sehr wichtig sind. Und wir werden uns daran halten, ihre Bedürfnisse so gut als möglich zu befriedigen. Falls sie Kritiken vorbingen, hören wir auf diese und versuchen unsere Sache entsprechend besser zu machen.

Meine Hauptkritik an Nessus 2.x ist der nicht dedizierte Nutzen einzelner Daten der Plugins. Versteh mich nicht falsch: NASL ist eine grossartige Sache, jedoch sollten die jeweiligen Datenfelder (z.B. Schweregrad, Plugin Family, CVE-Nummer, SecurityFocus-ID, usw.) in wirklich separaten standardisierten Variablen gespeichert werden. Dies würde einen spezifischer Zugriff, wie man ihn aus dem Datenbank-Bereich schon länger kennt, viel einfacher gestalten. Was waren Eure Design-Kriterien in diesem Belang?

Eigentlich werden Dinge wie Plugin Family, Querverweise und Beschreibungen in separaten Feldern im NASL-Skript selbst gespeichert. Mit Nessus 3 werden wir ein Utility einbringen, die das kommandozeilenorientierte Parsen für den einfacheren Zugriff auf die jeweiligen Felder zukässt (nasl -V). Wir sind ebenfalls dabei die Beschreibungen eines jeden Plugins zu überarbeiten, um eine Standardisierung zu erreichen und damit das Parsing einfacher umsetzen zu können. Ebenfalls wird sich die Risiko-Metrik einheitlich an CVSS orientieren.

Nun, es ist jedoch wahr, dass diese Datenfelder jeweils in einzelnen Dateien abgelegt werden. Die Idee dahinter ist, dass die jeweiligen Plugins sehr autonom gehandhabt werden können - Sie lassen sich nämlich ohne Probleme einzeln kopieren. Hätten wir den Plugin-Code an einer und die Meta-Daten an einer anderen Stelle gespeichert, hätte dies zu Synchronisations-Problemen führen können. Und dies sowohl während der Entwicklung als auch während der Nutzung und des Updatings. Dies hätte zu einem weitaus unstabileren Produkt geführt.

Das Speichern aller Infos eines Plugins in einer Datei vereinfacht das Ganze ungemein. Zum Schluss bleibt es Nessus überlassen, die jeweiligen Tools aufzurufen, die Plugin-Daten zu parsen und untereinander auszutauschen sowie die Resultate in die Scan-Datenbank zu speichern.

Nessus läuft primär in Unix/Linux-Umgebungen. (Kommerzielle) Clients sind für Windows verfügbar, können jedoch nicht eigenständig eingesetzt werden. Hat dies einen technischen Grund?

Nessus 2.x war aufgrund einiger früher Design-Entscheidungen lediglich auf Unix-Derivaten lauffähig. Dies wurde deshalb forciert, da die Verfügbarkeit des Produkts eine sehr hohe Priorität genossen hat. Dedizierte Prozesse wurden eingesetzt, um die einzelnen Aufgaben zu erledigen, so dass ein Abbruch einer Komponente nicht den Verlust der ganzen Arbeit zu Folge hatte.

Für Windows haben wir den NeWT-Scanner entwickelt, welcher auf der gleichen Nessus NASL Engine basiert und noch einige andere Kleinigkeiten des Unix-Bruders teilt.

In Nessus 3 haben wir verschiedene Wege eingeschlagen, um die gleiche Robustheit gewährleisten zu können, ohne lediglich immer auf Unix-spezifische Prozessstrukturen zurückgreifen zu müssen. Das Resultat dieser Bemühungen ist, dass Nessus 3 portabler geworden ist und sich die Windows-Adaption nahezu gleich verhält. Deshalb werden wir NeWT für Windows in Zukunft mit dem Code von Nessus 3 ersetzen.

„Das Speichern aller Infos eines Plugins in einer Datei vereinfacht das Ganze ungemein.“

Denkst Du, dass echtes Security Testing lediglich mit Unix-Systemen umgesetzt werden kann?

Meine persönlich Einstellung ist, dass Security Testing nur mit jenen Werkzeugen umgesetzt werden kann, die man versteht. Stehen keine derartigen Lösungen zur Verfügung, kann auch keine Sicherheitsüberprüfung gemacht werden. Wird eine solche Arbeit auf einem System gemacht, das man nicht richtig nutzen kann, dann kann dies zur Verlangsamung des Testings oder gar zur absoluten Verhinderung der Erfüllung des Auftrags führen.

In Anbetracht dieser Postulierung kann ich sagen, dass Windows weder schlechter noch besser als Linux, FreeBSD oder OpenBSD ist. Es ist eine andere Umgebung, mit anderen Möglichkeiten. So lange man dies versteht und mit den Limitierungen entsprechend umgehen kann, kann man damit anstellen, was man möchte.

Wenn Du an Nessus in fünf oder zehn Jahren denkst, was wird das Projekt wohl noch bringen? Was sind Deine Voraussagen für die Zukunft des Vulnerability Scannings und Nessus 4.0? Oder gibt es da andere Projekte, die Du gerne in Angriff nehmen möchtest?

Ich denke, dass die Benutzer und Administratoren in fünf Jahren nicht nur ihr Netzwerk nach Schwachstellen absuchen werden. Stattdessen wird wohl jedes System und jeder Dienst auf Verstösse gegen die Policies überprüft. Es geht dann nicht mehr darum zu erkennen, ob Apache/3.2.45 gegen eine Pufferüberlauf-Schwachstelle verwundbar ist - Vielmehr geht es um die entscheidende Frage, ob auf dem besagten System ein solcher Dienst überhaupt angeboten werden darf. Oder warum zur aktuellen Stunde ein Apple-Laptop im internen Netzwerk vorhanden ist, obschon das Unternehmen den Einsatz von Dell-Rechnern vorsieht.

Ich denke, dass das eine gute und wichtige Sache ist - Eine Vielzahl der Leute würde nicht mit der Aufforderung zum Einspielen von Patches überflutet werden, würden sie sich Gedanken darüber machen, welche Dienste effektiv von welchen Hosts eingesetzt werden sollen. Werden sämtliche unnötigen Dienste in einem Netzwerk abgeschaltet, hätte man heutzutage wohl schon 90 % aller Schwachstellen gelöst, die man ansonsten mit Bugfixes adressieren müsste.

Du bist nun seit etwa zehn Jahren in der IT Security Branche tätig. Was hat sich Deiner Meinung nach seit der Explosion der Popularität des Internets Mitte der 90er Jahre verändert?

Das offensichtlichste ist, dass die Leute das Internet mittlerweile breitwillig nutzen, um damit Geld zu machen - Sowohl legal als auch illegal. Dies erzwingt eine bessere Kontrolle der Netzwerke, so dass der Missbrauch darüber minimiert werden kann.

Vulnerability Scanning und Penetration Testing wird immer populärer im Bereich der Computersicherheit - Auch in der Schweiz. Diskussionen über Cyberwar sind jedoch zur aktuellen Stunde nicht gegenwärtig. Denkst Du, dass diese die echten Gefahren des jungen Jahrtausends sein werden?

Der Begriff Cyberwar trifft es wohl weniger, weder der Begriff Cyber-Bürgerkrieg. Wobei ich ernsthaft daran zweifle, dass eine Regierung wirklich jemanden über das Internet "bekämpfen" will, oder dies erst plant, bekommen Missbrauch der Netzwerke zwecks eigenes Profits ein Mehr an Bedeutung. Dies ist die echte Gefahr.

Das Resultat davon: Würdest Du Deine Kinder während eines Bürgerkriegs nicht auf die Strasse lassen, lass sie auch nicht in der heutigen Zeit das Internet selbst erkunden. Der Grund ist der, dass irgendein Kerl in Florida entschieden hat, dass er Geld mit dem Versand von pornografischem Spam verdienen will und seine Emails wahllos an millionen von Mailadressen verschickt. Oder ein anderer entscheided sich, einen Wurm zu schreiben, der hunderte von Systemen in Mail-Relays umfunktioniert.

Die Lösung dieses Problems ist nicht einfach, sowohl aus juristischer als auch aus technischer Sicht. Juristisch gesehen werden viele Leute sagen, dass einem die Hände gebunden sind, da das Internet ein Gebilde ohne Grenzen ist. Dies ist wahr, aber mit der gleichen fadenscheinigen Argumentation könnte ich die Gesetzgebung zu sexuellen Übergriffen anfechten.

„Die Leute nutzen das Internet in der heutigen Zeit, um Geld zu machen: Sowohl legal als auch illegal."

Auf der technischen Seite greifen wir auf einfache Mittel zurück, um Missbräuche erkennen oder vermeiden zu können (Proxies, Intrusion Detection-Lösungen, Intrusion Prevention-Systeme, Spam-Filter, Antivirus-Produkte - Die Liste würde unendlich lange werden). Eine 100 %ige technische Lösung ist keine absolute Lösung - Spam-Filter versagen (in zweierlei Hinsicht: Legitime Mails werden manchmal abgewiesen und Spam wird oftmals durchgelassen), Antivirus-Produkte verlangsamen die Rechner, usw.

Tools wie Nessus erlauben das einfache Umsetzen von Vulnerability Scannings durch jedermann. Was denkst Du über Exploiting Frameworks wie MetaSploit von H.D. Moore? Sind diese eine Gefahr oder eine Chance für Administratoren?

Ich denke, dass derartige Lösungen viel besser sind, werde einen Haufen zusammengetragener Exploits, die allesamt verschieden arbeiten. MetaSploit macht vieles einfacher, wie zum Beispiel die Entwicklung eigener Exploits oder die Adaption eines Shellcodes zum Umsetzung von Re-

mote-Zugriffen. Und sowohl für Kunden als auch ihre Berater ist ein solches Produkt viel beruhigender, weder irgendwelche obskuren Exploits von anonymer Stelle.

Die Angst, dass Tools wie MetaSploit für den Einbruch in Systeme missbraucht werden, ist natürlich berechtigt. Obschon dies so ist, denke ich nicht, dass MetaSploit die Dinge grundsätzlich einfacher macht. Will jemand wirklich in ein System eindringen, wird er nämlich sicher auch ansonsten sehr viel Zeit investieren wollen, um einzelne Exploits zu kompilieren und auszuprobieren.

Für die meisten Leute sind Nessus und nmap von Fyodor die wichtigsten Security-Tools seit der Implementierung des Ping-Kommandos. Jedoch gibt es auch noch einige andere grossartige Produkte, die durch ehrgeizige Leute vorangetrieben werden. Hast Du viel Kontakt mit den Initiatoren vergleichbarer Projekte?

Ein Projekt, das ich zur aktuellen Stunde sehr schätze, nennt sich Scapy (http://www.secdev.org/projects/scapy/) und wird von Philippe Biondi betreut. Scapy ist ein interaktives Tool, mit dem Netzwerk-Pakete generiert und verarbeitet werden können. Ich hoffe, dass dieser Lösung in Zukunft mehr Beachtung geschenkt wird.

Vielen Dank für Deine Zeit und das interessante Interview. Ich wünsche Dir und dem Nessus Projekt natürlich alles Gute für die Zukunft!

Ich hoffe, dass ich mich den Fragen umfassend genug angenommen habe. Sollte es etwaige Fragen geben, so stehe ich der Leserschaft natürlich gerne zur Verfügung.

Herausgeber

)SCIP(scip ag
scip AG
Technoparkstrasse 1
CH-8005 Zürich
+41 44 445 1818
mailto:info@scip.ch
http://www.scip.ch

Zuständige Person:
Marc Ruef
Security Consultant
+41 44 445 1812
mailto:maru@scip.ch

scip AG ist eine unabhängige Aktiengesellschaft mit Sitz in Zürich. Seit der Gründung im September 2002 fokussiert sich die scip AG auf Dienstleistungen im Bereich IT-Security. Unsere Kernkompetenz liegt dabei in der Überprüfung der implementierten Sicherheitsmassnahmen mittels **Penetration Tests** und **Security Audits** und der Sicherstellung zur Nachvollziehbarkeit möglicher Eingriffsversuche und Attacken (**Log-Management** und **Forensische Analysen**). Vor dem Zusammenschluss unseres spezialisierten Teams waren die meisten Mitarbeiter mit der Implementierung von Sicherheitsinfrastrukturen beschäftigen. So verfügen wir über eine Reihe von Zertifizierungen (Solaris, Linux, Checkpoint, ISS, Cisco, Finjan, TrendMicro, Symantec etc.), welche den Grundstein für unsere Projekte bilden. Den kleinsten Teil des Wissens über Penetration Test und Log-Management lernt man jedoch an Schulen – nur jahrelange Erfahrung kann ein lückenloses Aufdecken von Schwachstellen und die Nachvollziehbarkeit von Angriffsversuchen garantieren.