Norbert Pohlmann

Die Vertrauenswürdigkeit von Software

Eine Analyse und Diskussion über die Beurteilung und den Aufbau von Vertrauenswürdigkeit von Software

Die Software-Qualität der Betriebssysteme und Anwendungen ist für die heutige Bedrohungslage der IT und des Internets nicht mehr ausreichend. Die Frage, die in diesem Artikel behandelt wird, ist: Wie schaffen wir es mit einer höheren Vertrauenswürdigkeit von Software eine höher IT- und Internet-Sicherheit zu erreichen?

1 Einleitung

Software stellt in allen Branchen einen immer größeren Wertschöpfungsanteil dar. Wir nutzen Software in PCs, Notebooks, Smartphones, in sehr großen Rechenzentren, aber auch immer mehr in Autos, in Industrieanlagen, im Haus und vielen weiteren Lebensbereichen. Doch in der aktuell genutzten Software sind zu viele Fehler – ein großes Sicherheitsproblem. Die Software-Qualität der Betriebssysteme, Anwendungen und Dienste reicht bei der heutigen Bedrohungslage nicht mehr aus. So liegt die Fehlerdichte, also die Anzahl an Softwarefehlern pro 1.000 Zeilen Code, in qualitativ hochwertiger Software heute im Schnitt bei 0,3. Da gängige Betriebssysteme und große Anwendungen zehn Millionen Zeilen Code und mehr haben, sind demnach durchschnittlich 3.000 Softwarefehler zu finden. Teile dieser Softwarefehler sind Ziele für professionelle und erfolgreiche Angriffe von kriminellen Organisationen und Nachrichtendienste, wie die NSA. Bei den großen Betriebssystemen, Anwendungen und Diensten ist in den nächsten zehn Jahren auch mit keiner sprunghaften Verbesserung der Software-Qualität zu rechnen und selbst wenn: Auch bei verbesserter Software-Qualität werden die Angreifer noch vorhandene Software-Schwachstellen professioneller ausnutzen.

In diesem Artikel soll der Aspekt Vertrauenswürdigkeit von Software analysiert und diskutiert werden.

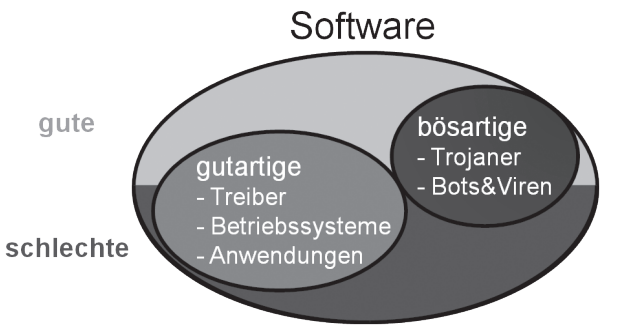
2 Klassifizierung von Software bezüglich der Qualität und Vertrauenswürdigkeit

Im Folgenden wird die Software in unterschiedliche Klassen aufgeteilt, um das Problem der Qualität und Vertrauenswürdigkeit von Software besser diskutieren zu können [Pohl11].

Norbert Pohlmann

Professor für Informationssicherheit und geschäftsführender Direktor des Instituts für Internet-Sicherheit an der Westfälische Hochschule Gelsenkirchen sowie Vorstandsvorsitzender des TeleTrusT – Bundesverband IT-Sicherheit.
E-Mail: pohlmann@internet-sicherheit.de

Abb. 1 | Gute und schlechte sowie gutartige und bösartige Software



2.1 Gute Software

Eine gute Software erreicht ein hohes Maß an Qualität, das heißt, die Anzahl der Softwarefehler ist minimal. Eine gute Software ist gegeben, wenn sie eine hohe Funktionalität aufweist und alle gewünschten Funktionen korrekt und zuverlässig arbeiten. Außerdem stellt eine gute Software eine einfache und verständliche Benutzerschnittstelle zur Verfügung. Eine gute Software kann in einem hohen Maße Datensicherheits- und Datenschutzansprüche erfüllen, weist so wenig wie nur möglich Softwarefehler und damit auch Schwachstellen auf und ist somit besser geschützt gegen Malware-Angriffe. Je weniger Zeilen Code eine Software hat, desto höhere Qualität kann prinzipiell erreicht werden.

2.2 Gutartige Software

Zu gutartiger Software zählen in der Regel normale Software-Produkte, wie Treiber, Betriebssysteme und kleine und große Anwendungen. Gutartige Software sind Lösungen, die im privaten und betrieblichen Umfeld eingesetzt werden, um gewünschte Aufgaben zu bewältigen. Anwender von gutartiger Software nutzen diese wissentlich und aktiv. Bei gutartiger Software bestehen bei den vertrauenswürdigen Herstellern der Software keine kriminellen Hintergedanken. Die Software bietet die Funktionen und Dienste, die für die definierte Aufgabenstellung notwendig sind. Nach der Snowden Aufklärung müssen wir allerdings davon ausgehen, dass diese Software-Produkte, z.B. aus den

USA, bewusst nicht entfernte Schwachstellen enthalten, also nicht gutartig sein müssen.

Software, die nach dem Geschäftsmodell „Bezahlen mit persönlichen Daten“ angeboten wird, stellt bei der Beurteilung der „Gutartigkeit“ eine besondere Herausforderung dar. Dem Kunden werden Features angeboten, die er gerne haben möchte, aber die Software macht auch Dinge, die in der Regel nicht vom Nutzer gewünscht sind, wie das Auslesen von Kontaktdaten, Position, … und das Erstellen von Profilen, um diese direkt oder indirekt über Werbung verkaufen zu können.

Wichtig zu verstehen ist, dass gutartige Software sowohl „gut“ als auch „schlecht“ sein kann!

Die Vertrauenswürdigkeit eines Software-Herstellers und die Evaluierung der Eigenschaften und Funktionen der Software sind ein gutes Maß für die Beurteilung der Gutartigkeit einer Software. Ein Attribut für eine gutartige Software kann z.B. ein Siegel, wie das „IT Security made in Germany“, sein (https://www.teletrust.de/itsmig/qualitaetszeichen/).

2.3 Schlechte Software

Eine schlechte Software hat viele Softwarefehler (Schwachstellen, Bugs, etc.) und ist damit grundlegende Ursache für erfolgreiche Remote-Angriffe auf unsere Computer. Das Risiko für die Ausnutzung der Schwachstellen und damit für Schäden ist entsprechend groß, da sich kriminelle Organisation und Nachrichtendienste zunehmend auf dieses Problem konzentrieren. Die Erfolgsaussichten eines positiven Remote-Angriffes auf unsere Computer und die gespeicherten Werte sind sehr groß. Aus diesem Grund ist eine schlechte Software das Übel unserer modernen IT und sorgt dafür, dass so viele Computer mit Malware (Schadsoftware) infiziert sind! Die Ursachen für schlechte Software sind immer noch: Steigende Komplexität der Software, kein Sicherheitsbewusstsein der Softwareentwickler, fehlende Expertisen der Softwareentwickler (schlechter Programmierstil, mangelnde Informationen über eingesetzte Bibliotheken und Komponenten), fehlendes Wissen über aktuelle Sicherheitsbedrohungen, der Zeitdruck für die Fertigstellung der Software (Time-to-Market) und damit verbunden unzureichendes Testen und kurze Anforderungsphasen und ein daraus resultierender unsystematischer Entwurf. Diese Liste der Gründe für schlechte Software ist nicht vollständig und daher noch erweiterbar. Der Softwareentwicklungsprozess verläuft häufig unsystematisch und für heutige Anforderungen an die Software nicht professionell genug.

Tab. 1 | http://scan.coverity.com

Code-Basis	Code-Zeilen	Anzahl der Fehler	Analysedauer (Min.)	Fehlerdichte
Apache	127,839	32	10	0.250
Ethereal	1,157,801	143	108	0.124
Firebird	239,701	163	13	0.680
Firefox	303,908	108	24	0.355
FreeBSD	1,582,166	635	257	0.401
Linux	3,171,631	1062	254	0.335
Mysql	607,639	136	68	0.224
NetSNMP	173,138	148	16	0.855
OpenLDAP	254,004	158	20	0.622
OpenSSL	194,751	66	19	0.339
OpenVPN	69,610	7	4	0.101
Perl	479,759	89	25	0.186
PHP	430,817	204	36	0.474
PostGres	815,562	295	38	0.362
ProFTPD	89,834	26	4	0.289
Python	258,272	96	16	0.372
Samba	310,592	216	34	0.695
Snort	82,919	48	4	0.579

Leider sind sehr viele Softwareentwickler heute immer noch nicht genug ausgebildet, um gute Software zu schreiben. Sehr negativ ist auch, dass viele Hersteller von Software viel zu wenig Verantwortung übernehmen, um dieses Problem nachhaltig zu lösen!

Die Tabelle (siehe Tab. 1) zeigt eine Übersicht über die Fehlerdichte von Open Source Softwareprogrammen. Eine Fehlerdichte von 0.25 bedeutet z.B., dass auf 1000 Zeilen Code im Schnitt viele Softwarefehler vorhanden sind.

Die Anzahl der Softwarefehler bei Open und Closed Source Software ist im Prinzip gleich (siehe auch [Cove12]).

Außerdem müssen wir erkennen, dass sich diese Situation der Softwarequalität auch nicht kurzfristig ändern wird, d.h. die Fehlerdichte von Software wird zwar kleiner, Fehlerfreiheit ist zurzeit aber für große Software nicht erreichbar. Trotz kleiner werdender Anzahl der Software-Schwachstellen wächst, wegen ihrer professionellen Nutzung durch die NSA und kriminelle Organisationen, die Bedrohung durch gezielte Angriffe weiter.

Ein besonderes Risiko sind sogenannte Zero-Day-Attacks. Entdeckt jemand eine Sicherheitslücke und meldet diese nicht dem Software-Hersteller, so wird die Schwachstelle der Software erst nach dem ersten Angriff bekannt. Zero-Day-Attacken sind daher sehr effizient, weil sie großflächig neue Sicherheitslücken ausnutzen können, bevor Sicherheitsprodukte, wie z.B. Virenscanner, die benötigten Signaturen zum Erkennen der Angriffscodes bereitstellen können oder die Softwarehersteller in der Lage sind, Software-Upgrades zur Verfügung zu stellen.

Die NSA kauft im Jahr für 25 Mio. Dollar Zero-Day-Attacks und schafft damit einen attraktiven Markt, der die Software insgesamt sehr viel angreifbarer macht.

Wichtig zu verstehen ist, dass eine schlechte Software die Basis für bösartige Software – für Malware – ist!

Wir gehen heute davon aus, dass die Geheimdienste keine Backdoors (Hintertüren) im Produkt einbauen lassen müssen, weil die Nutzung von Schwachstellen immer einen unerlaubten Zugriff ermöglicht.

2.4 Bösartige Software: Malware

Malware ist der Oberbegriff für „Schadsoftware“ wie Viren, Würmer, Trojanische Pferde und andere. Angreifer wie kriminelle Organisationen, Spione oder Terroristen - nutzen Software-Schwachstellen von schlechter Software aus, um Malware auf Computern zu installieren. Hauptsächlich über E-Mail-Anhänge oder unsichere Webseiten mit Hilfe von sogenannten Drive-by Downloads wird Malware in Computer unbemerkt eingeschleust. Das Institut für Internet-Sicherheit geht zurzeit davon aus, dass auf jedem 20. IT-Endgerät in Deutschland ungewollte intelligente Malware vorhanden ist, die über ein Botnetz gesteuert wird [Pohl13]. Ein Botnetz ist eine Gruppe von Computern, die unter zentraler Kontrolle eines Angreifers steht und von ihm für Angriffe genutzt wird.

Mit Hilfe von intelligenter Malware auf fremden Computern können Angreifer diese vielfältig manipulieren und nutzen:

Eine Keylogger-Funktion in Malware speichert alle Informationen, die z.B. über die Tastatur vom Nutzer in das eigene Computersystem eingegeben werden. Diese Informationen, hauptsächlich Identitäten, Passwörter und Kreditkarteninformationen, werden dann von der Malware regelmäßig in sogenannte Drop-Zonen im Internet gesendet. Drop-Zonen sind Speicherbereiche von beliebigen Servern im Internet, von denen sich die Angreifer die wertvollen Informationen unentdeckt holen können und damit Angriffe auf die Internet-Dienste der Opfer durchführen. Außerdem hat Malware Funktionen, mit der sie in der Lage ist, das identifizierte Rechnersystem beliebig zu steuern und z.B. vertrauliche Dateien auszulesen. Weitere typische und nutzbare Mal-Funktionen von Malware sind: Spam-Verteilung, Beteiligung an DDoS-Angriffe und Click Fraud.

Intelligente Malware von heute ist sehr flexibel. Kriminelle Organisationen sind in der Lage, Malware aus der Ferne zu steuern und auch neue Funktionen nachzuladen, sowie weitere Computer zu infizieren.

Eine weitere Herausforderung in diesem Bereich ist, dass zunehmend Malware gezielt für Personen und deren Computer geschrieben und genutzt wird. Mit solchen „Targeted Attacks" sollen typischerweise Informationen, wie Strategiepläne von Vorständen/Politikern/Generälen, Entwicklungsdaten von neuen Produkten, usw. entwendet werden. Die Motivation liegt in den Bereichen wirtschaftliche, politische oder militärische Spionage. Da bei solchen gezielten Angriffen die Anti-Malware-Produkte konzeptionell schlechter wirken, muss hier nach neuen wirkungsvollen Sicherheitskonzepten gesucht werden.

Advanced Persistent Threat (APT) ist die Begrifflichkeit, die sich für intelligente Malware wie Stuxnet und Flame international etabliert hat. Advanced Persistent Threat (APT) wird in der Regel als ein gezielter Angriff mit komplexen Angriffstechnologien und -taktiken sowie aufwendigen Hintergrundinformationen eines Opfer-IT-Systems und dessen Umgebung verstanden. Dabei nimmt der Angreifer einen großen Aufwand auf sich (Advanced), um erfolgreich auf ein Opfer-IT-System zuzugreifen und möglichst lange (Persistent) unentdeckt zu bleiben. So kann es über einen längeren Zeitraum Informationen ausspähen oder Schaden anrichten.

Vor einigen Wochen hat sich Symantec als größter Hersteller von Anti-Malware- Lösungen zu Wort gemeldet und mitgeteilt, dass sie nur noch 45 % der Malware erkennen. Diese Zahl spiegelt sicherlich das neue Verhältnis zwischen gezielten und Massen-Angriffen wider.

Bedauerlicherweise ist Malware sehr oft eine sehr „Gute Software" im Sinne der Klassifizierung von Software!

2.5 Open versus Closed Source Software

Welchen Einfluss hat Open Source Software auf die Klassifizierung der Software?

Grundsätzlich sind bei Open Source Software natürlich sehr viele Personen in der Lage, die Qualität der Software zu verbessern und Schwachstellen zu finden, damit diese reduziert werden können. Die Realität zeigt aber, dass leider nicht so viele helfen oder helfen können, die Software-Qualität von Open Source Software zu verbessern.

Zurzeit ist die Software-Qualität von Open und Closed Source Software die gleiche.

Ein besonderer Vorteil von Open Source Software ist die Möglichkeit, einen höheren Grad an Vertrauenswürdigkeit erreichen zu können. Hier liegt ein sehr großes Potenzial von Open Source Software, das zurzeit leider nicht ausreichend ausgeschöpft wird, weil dies sicherlich auch sehr viel Aufwand bedeutet und daher auch finanziert werden müsste.

Außerdem ist es bei Open Source Software sehr einfach möglich, eigene IT-Sicherheitslösungen einzubinden, was auch die Vertrauenswürdigkeit erhöht.

Auf der anderen Seite können Angreifer bei einer Open Source Software deutlich einfacher und besser Schwachstellen finden und diese für sich ausnutzen, was einen prinzipiell höheren Einsatz von böswilliger Software möglich macht.

Im Bereich der Open Source Software ist die Herausforderung das Potenzial der Open Source Gemeinde, mehr Verantwortung zu übernehmen.

Die Schwachstelle in Open Source Software OpenSSL, die unter den Namen Heartbleed berühmt wurde, zeigt deutlich, dass sehr viele große Unternehmen zwar eine Open Source Software nutzen, aber nicht helfen, diese auch sicher zu bekommen und zu halten. Hier wäre eine größere Verantwortung der Banken und Großunternehmen sicherlich hilfreich.

2.6 Bewertung

Zu viel schlechte Software erlaubt den Angreifern, böswillige Software zu platzieren und remote-mäßig auf fremde Computer zuzugreifen, um Werte auszulesen oder von diesen Computern aus Schäden anzurichten. Dies ist zurzeit das größte IT-Sicherheitsproblem, das wir sehr schnell überwinden müssen.

3 Wie kann die Vertrauenswürdigkeit von Software erlangt werden?

Vertrauenswürdigkeit hat etwas mit Vertrauen zu tun. Aber welche Kriterien zur Beurteilung der Vertrauenswürdigkeit sollten bei Software angewendet werden?

Im Folgenden werden ein paar Aspekte aufgezeigt, wie eine höhere Vertrauenswürdigkeit von Software erzielt werden kann.

3.1 Nutzung und Darstellung eines Security Development Lifecycle (SDL)

Ein wichtiger Punkt ist die Nutzung und Darstellung eines Security Development Lifecycle (SDL). Der SDL beschreibt Richtlinien, wie eine Projektplanung und -durchführung unter der Berücksichtigung der Sicherheit realisiert werden kann.

Aspekte sind z.B.:

- Eine strikte Trennung von Entwicklungs- und Testabteilung. Die Entwickler sollten nicht ihren eigenen Programmcode verifizieren.
- Tests von Release Versionen, nicht nur Versionen in der Entwicklung.
- (Grafische) Übersicht über das Gesamtsystem
- Klare Definition der Zuständigkeiten

- Modulare Gestaltung der Software. Somit kann jeder Teil unabhängig getestet werden. Als sicher geltende Teile könnten in Kombination getestet werden.
- Einschätzung des Risikos, geeignet wären Klassifizierungsschemen wie DREAD und STRIDE des „Threat Risk Modeling" des Open Web Application Security Project [owaps]
- Sicherer Betrieb der Software, nach Release der Software sollte diese auch möglichst sicher betrieben werden. Anwender/Nutzer schulen möglichst sichere Standardkonfiguration
- Abhärtung gegenüber Angriffen von Innen: Logs, Berechtigungsmodell, usw..
- Nutzen von bewährten Tools zum Test der Anwendung
- (GUTE) Dokumentation, von Zuständigkeiten, Tests, ...
- …

Diese Punkte werden jedoch von vielen Firmen nicht richtig umgesetzt, können aber in Form einer Zertifizierung nach außen getragen werden, um eine höhere Vertrauenswürdigkeit zu erzielen.

3.2 Evaluierung/Zertifizierung

In der Praxis ist es so, dass die Software-Sicherheit durch eine Evaluierung/Zertifizierung erzielt werden kann.

Bei der Evaluierung und Zertifizierung müssen unabhängige und qualifizierte Organisationen die Qualität und Vertrauenswürdigkeit von IT und IT-Sicherheit in Produkten und Lösungen prüfen. Als Ergebnis haben wir ein höheres Vertrauen, das durch eine vertrauenswürdige Organisation bestätigt wird. Diese Vorgehensweise ist notwendig, da wir Nutzer dies nicht selbst tuen können.

3.3 Qualitätszeichen

Ein weiterer und nach der NSA-Affäre immer wichtigerer Aspekt für das Erlangen von Vertrauenswürdigkeit von Software im Bereich der IT-Sicherheit ist z.B. das TeleTrusT-Qualitätszeichens „IT Security made in Germany".

Die Verwendung des markenrechtlich geschützten TeleTrusT-Qualitätszeichens „IT Security made in Germany" wird interessierten Anbietern durch TeleTrusT auf Antrag und bei Erfüllung der nachstehenden Kriterien zeitlich befristet gestattet.

Kriterien, die erfüllt sein müssen:

1. Der Unternehmenshauptsitz muss in Deutschland sein.
2. Das Unternehmen muss vertrauenswürdige IT-Sicherheitslösungen anbieten.
3. Die angebotenen Produkte dürfen keine versteckten Zugänge enthalten (keine „Backdoors").
4. Die IT-Sicherheitsforschung und -entwicklung des Unternehmens muss in Deutschland stattfinden.
5. Das Unternehmen muss sich verpflichten, den Anforderungen des deutschen Datenschutzrechtes zu genügen.

Bei nachträglicher Nichterfüllung eines oder mehrerer Kriterien kann die Zeichennutzung durch TeleTrusT untersagt werden.

3.4 Produkthaftung

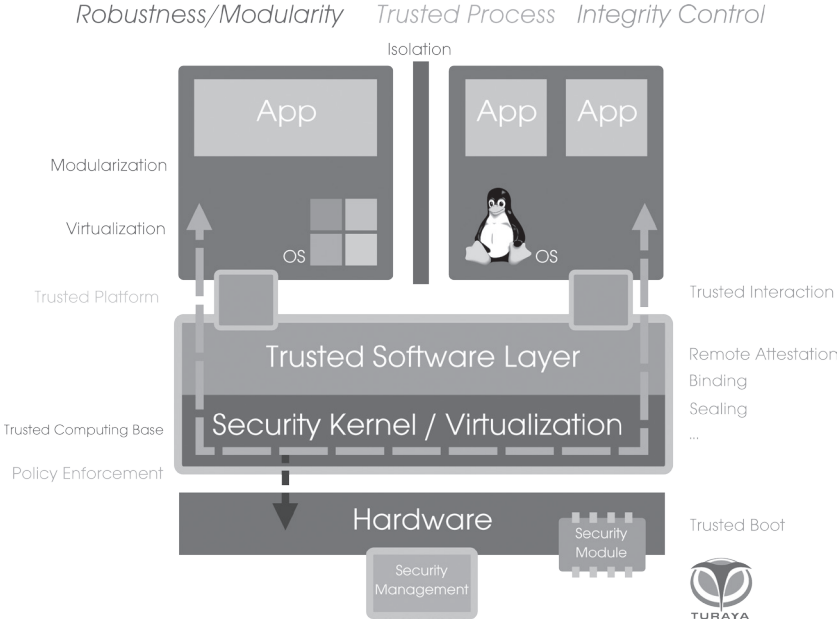
Zurzeit bestimmen die großen Technologiehersteller und Dienste-Anbieter, wie Google, Apple, Facebook und Microsoft, was wir als Nutzer brauchen. Doch die Verantwortung für ihre Lösungen übernehmen sie nicht. Was wir allerdings dringend benötigen, ist eine Herstellerverantwortung wie in der Automobilbranche! Wenn wir heute ein Auto kaufen, übernimmt der Hersteller, bei dem wir das Auto kaufen, gegenüber uns die volle Verantwortung. Aber auch die Automobilhersteller arbeiten mit mehreren hundert Zulieferern zusammen. Und doch gibt es für uns immer nur einen Ansprechpartner. Die Hersteller lassen die Autos überprüfen und wenn sie einen Fehler erkennen, werden große Rückrufaktionen gestartet, um Fehler zu beheben, bevor die eigentlichen Probleme auftreten. Dies hat ein sehr großes Vertrauen zu den Herstellern aufgebaut. Wer aber übernimmt die Verantwortung für IT-Systeme? Am Ende keiner! Wenn die IT-Hersteller beginnen würden, die Verantwortung zu übernehmen, dann würden die heutigen IT-Sicherheitsprobleme deutlich geringer. Alle Softwareprogramme und die Hardware wären besser aufeinander abgestimmt und Fehler würden einfacher gefunden und behoben.

4 Moderne Sicherheitsarchitektur für vertrauenswürdige IT-Systeme

Kern aller Bemühungen einer modernen Sicherheitsarchitektur ist die Bildung einer „Trusted Platform", einer sicheren und vertrauenswürdigen Basis für Anwendungen. Die Trusted Platform, in der Form einer Sicherheitsplattform, auf der Basis von intelligenten kryptographischen Verfahren, stellt eine sichere Umgebung zum Schutz von sicherheitskritischen Daten für sicherheitskritische Operationen bereit, die sehr viele heutige IT-Sicherheitsprobleme reduzieren [LiPo07].

In der Abb. 2 und im folgenden Text wird dargestellt, mit welcher Sicherheitsarchitektur, mit welchen Sicherheitsprinzi-

Abb. 2 | Moderne Sicherheitsarchitektur



pien und IT-Sicherheitsmechanismen ein vertrauenswürdiges IT-System aufgebaut werden kann. Beginnen wir mit den Sicherheitsaspekten, die die Robustheit und die Modularität betreffen [PoSp13].

Trusted Computing Base (TCB): Per Definition ist die „Trusted Computing Base" der kritische Teil eines IT-Systems. Wenn im TCB eine Schwachstelle vorhanden ist, dann ist das ganze IT-System betroffen. Wenn außerhalb der TCB eine Schwachstelle vorhanden ist, dann kann anhand einer Sicherheitspolicy der potenzielle Schaden sehr eingeschränkt und beschrieben werden. Aus diesem Grund ist eine TCB sehr sorgfältig designed und implementiert. Eine auf einem Microkernel basierende TCB hat ca. 20.000 Line of Codes und 80.000 Line of Codes für die Virtualisierungstechnologie und ist von daher eine sehr vertrauenswürdige Basis, die in der Regel auch schon semiformal verifiziert werden kann. Es gibt aber auch TCBs, die z.B. aus einem sehr abgespeckten und speziell gehärteten Linux bestehen, das sehr viel vertrauenswürdiger als übliche Betriebssysteme ist!

Ein weiterer Sicherheitsaspekt ist die Virtualisierung. Hier kann der Robustheitsvorteil genutzt werden, der schon seit langem bei Servern verwendet wird. Es ist sehr einfach möglich, die verschiedenen virtuellen Maschinen, wieder in einen stabilen Urzustand zu versetzen und von da aus neu zu starten.

Der Sicherheitsaspekt Isolierung sorgt dafür, dass die virtuellen Maschinen isoliert und sicher getrennt voneinander laufen und sich so nicht gegenseitig beeinflussen können. Daher haben Schwachstellen in einer isolierten virtuellen Maschine keinen Einfluss auf die anderen virtuellen Maschinen. Eine solche stark isolierte virtuelle Maschine wird im Bereich von TC auch Compartment genannt.

Der Sicherheitsaspekt der Modularisierung gibt uns eine interessante und flexible Möglichkeit, Anwendungen, die aus Sicherheitsgründen zusammen gehören, in einer virtuellen Maschine laufen zu lassen und Anwendungen, die aus Sicherheitsgründen getrennt sein sollten, in verschiedenen virtuellen Maschinen zu positionieren. Hier haben wir einen interessanten Gestaltungsspielraum, mit dem sehr hohe IT-Sicherheit erzielt werden kann, weil für verschiedene Sicherheitsprobleme unterschiedliche virtuelle Maschinen genutzt werden können.

Mit dem generellen Sicherheitsaspekt „Integritätsüberprüfung" kann die Integrität und damit der vertrauenswürdige Zustand eines IT-Systems überprüft werden. Die Trusted Software Layer stellt dazu vertrauenswürdige Sicherheitsdienste zur Verfügung, die helfen, IT-Systeme (Hardware, Software und Konfigurationen) vertrauenswürdig zu gestalten und messbar zu machen. Das Security Modul ist z.B. ein TPM mit intelligenten kryptographischen Verfahren auf dem Level von SmartCard-Sicherheit, aber auch weiteren Sicherheitsdiensten, wie die Platform Configuration Register (PCR), die die sichere Speicherung und Überprüfung von Messdaten garantiert. Das TPM ist ein kleiner passiver Sicherheitschip, der fest mit dem Mainboard verbunden ist (siehe auch den Artikel „Hardware-Sicherheitsmodule zum Schutz von sicherheitsrelevanten Informationen" im gleichen Heft).

Es war schon immer ein Traum der IT-Sicherheitsspezialisten ein Sicherheits-Modul auf den IT-Systemen zur Verfügung zu haben!

Trusted Boot sorgt dafür, dass das IT-System nur in einem definierten vertrauenswürdigen Zustand aktiv wird. Remote Attestation gibt die Möglichkeit, die Vertrauenswürdigkeit von anderen, auch fremden IT-Systemen zu messen, bevor eine Interaktion mit diesem IT-System begonnen wird. Binding und Sealing sind weitere Trusted Computing Funktionen, mit denen moderne IT-Sicherheitssysteme intelligent umgesetzt werden können. Unter dem Begriff Trusted Interaction werden Sicherheitsdienste in der Trusted Software Layer zusammengefasst, die dafür sorgen, dass Informationen vertrauenswürdig eingegeben, gespeichert, übertragen und dargestellt werden können. Trusted Process vereint die Sicherheitsaspekte, die die Abläufe in den und mit den verschiedenen IT-Systemen betreffen.

Security Management fasst die wichtigen Funktionen zusammen, die notwendig sind, um das proaktive Sicherheitssystem vertrauenswürdig nutzbar zu machen. Mit Hilfe des Policy Enforcement sind wir in der Lage, die definierten Regeln auf unserem eigenen, aber auch auf fremden IT-Systemen vertrauenswürdig umzusetzen.

Trusted Virtual Domains ist ein Umsetzungskonzept, das es uns ermöglicht, übergreifende Sicherheitskonzepte vertrauenswürdig umzusetzen. Dies ist insbesondere bei Cloud- und Verteilten-Anwendungen" erforderlich.

All diese Sicherheitsaspekte zusammen, stellen die vertrauenswürdige Basis, einer Trusted Platform, dar!

Eine Trusted Platform ist ideal dazu geeignet, das Problem der Vertrauenswürdigkeit von Software auf eine angemessene Ebene zu führen und unsere moderne IT in der Zukunft deutlich sicherer zu gestallten.

5 Zusammenfassung

Unsere IT-Technologie und insbesondere Software haben zurzeit Grenzen, sich gegen intelligente, moderne Angriffe angemessen schützen zu können.

Beim Design von IT-Systemen hilft es sehr, wenn wir neue Wege gehen und z.B. durch die Nutzung einer Trusted Platform mit schlankem Mirokernel und Virtualisierung eine deutliche bessere Vertrauenswürdigkeit erzielen können.

Wir brauchen zusätzlich eine klare und umfangreiche Produkthaftung für IT und IT-Sicherheit, um eine deutlich höhere IT-Sicherheit und Vertrauenswürdigkeit zu motivieren. Dies soll aktiv umgesetzt und gefördert werden. Die Hersteller müssen Verantwortung übernehmen, um wieder Vertrauen zu schaffen, damit wir mit einem sehr guten Gefühl die innovativen Möglichkeiten auch in Zukunft gerne nutzen werden.

Literatur

[LiPo07] M. Linnemann, N. Pohlmann: „Turaya – Die offene Trusted-Computing-Sicherheitsplattform", in „Open Source Jahrbuch 2007", Hrsg.: B. Lutterbeck, M. Bärwolff, R. Gehring, Lehmanns Media, Berlin, 2007

[Cove12] Coverity Scan: 2012 Open Source Report https://scan.coverity.com/

[Pohl13] N. Pohlmann: „Daten gegen Diebstahl sichern", Wirtschaftsspiegel, IHK Münster, 2/2013

[Pohl11] N. Pohlmann: „Bugs, die Nahrung für Malware – Von guter, schlechter und böser Software", IT-Sicherheit – Management und Praxis, DATAKONTEXT-Fachverlag, 4/2011

[PoSp13] N. Pohlmann, A. Speier: „Eine Diskussion über Trusted Computing – Sicherheitsgewinn durch vertrauenswürdige IT-Systeme". IT-Sicherheit – Management und Praxis, DATAKONTEXT-Fachverlag, 5/2013

[owaps] https://www.owasp.org/index.php/Threat_Risk_Modeling