Seminar Organic and Ubiquitous Computing

Computer Immunologie

Peter Sommer

9. Januar 2005

Betreuer: Wolfgang Trumler
Lehrstuhlinhaber: Prof. Dr. Theo Ungerer

Zusammenfassung

Diese Arbeit soll aufzeigen, warum es wünschenswert ist, biologische Immunsysteme auf Computersysteme und Netzwerke zu übertragen. Es wird angedeutet, wie dies geschehen kann, ohne jedoch auf konkrete Implementierungen einzugehen. Es wird dabei vorwiegend der Ansatz von S. Forrest und S. Hofmeyr erläutert. Dieser wendet Anomalie-Erkennung auf Systemaufrufe einzelner Programme an. Es werden außerdem Beispiele für andere mögliche Vorgehensweisen genannt. Es wird auch darauf eingegangen, welche Probleme durch ein Computer-Immunsysteme entstehen können, und was wir noch tun müssen, bevor einsetzbare Implementierungen in Angriff genommen werden können.

1 Einleitung

Aus unserem heutigen Gesellschaftsleben sind Computer-Systeme nicht mehr wegzudenken. Damit ist aber in den letzten Jahren auch mehr und mehr die Gefahr erwachsen, dass Viren, Würmer und Hackerangriffe diese Systeme lahm legen und einen immensen wirtschaftlichen Schaden verursachen können. Wie aktuelle Fälle gezeigt haben, sind heutige Schutzmechanismen nicht ausreichend. Die momentane Sicherheit von Computersystemen wird vorwiegend durch Firewalls und Virenscanner-Software realisiert. Diese sind Signatur-basiert, d.h. Angriffe und schädliche Programme können nur dann erkannt werden, wenn diese Signaturen/Regeln zuvor entweder in Form einer Sicherheitsrichtlinie von menschlicher Hand in entsprechender Form kodiert wurden, oder entsprechende

„Virendefinitionen“ vorliegen. Es handelt sich um statische, d.h. nicht-adaptive Schutzmechanismen, die nicht in der Lage sind, bisher unbekannte Angriffe und Schädlinge zu erkennen und abzuwehren. Um Sicherheitslücken (d.h. Angriffspunkte) in Software schließen zu können, muss zumeist auf ein Patch des Herstellers gewartet werden. Selbst wenn ein Patch zeitnah zur Verfügung steht, steht dessen Verbreitung im Wettbewerb mit der Verbreitung der Angriffe, die diese Sicherheitslücken ausnutzen. Dieses Wettrennen entschied schon oft der Schädling für sich (vgl. Sasser). Aus den genannten Gründen wäre es wünschenswert, einen adaptiven Sicherheitsmechanismus zu etablieren, der im Stande ist, bisher unbekannte Angriffe zu erkennen und abzuwehren. Dies ist möglich über sogenannte Anomalie-Erkennung, wie sie das menschliche Immunsystem in ähnlicher Form verwendet.

Zunächst wird das biologische Immunsystem, soweit relevant, erklärt. Der darauf folgende Abschnitt hebt die Eigenschaften eines biologischen Immunsystems hervor, die für die Informatik interessant sind. Nachdem eine Möglichkeit der Implementierung beschrieben wurde und Alternativen genannt wurden, wird ein Vergleich mit bestehenden Systemen dargestellt. Weiter wird auf die Nachteile und möglicherweise entstehende Gefahren eingegangen und ein Ausblick auf noch anstehende Aufgaben gegeben.

2 Immunologie

Um die zum Verständnis notwenigen Eigenschaften des biologischen Immunsystems von Wirbeltieren zu vermitteln, werden nachfolgend die grundlegenden Mechanismen erklärt, die in [7] detailliert beschrieben werden.

Das Immunsystem (im Folgenden IS) basiert auf zwei Aspekten: dem Erkennen von Pathogenen (Schädlingen) und der Eliminierung derselben. Man geht davon aus, dass die Erkennung auf dem Unterscheiden zwischen „körpereigenen“ Proteinen (in der Literatur „self“) und „körperfremden“ Proteinen bzw. Antigenen (in der Literatur „nonself“) basiert. Genauer gesagt müssen gefährliche Fremdkörper erkannt werden, denn es gibt im Körper Mikroorganismen, ohne die unser Köper nicht weiter funktionieren würde. Deswegen sollte das IS nicht gegen diese vorgehen. Nach der Erkennung werden Pathogene durch Antikörper (Abwehrstoffe) eliminiert.

Das menschliche IS basiert auf dem Prinzip der Vielschichtigkeit. Die erste Barriere ist die Haut. Sie verhindert ein physisches Eindringen vieler Pathogene. Dieser folgt ein physiologisches Hindernis, das die Lebensbedingungen der Pathogene verschlechtert (Temperatur, pH-Wert). Die nächste Barriere ist das eigentliche (reaktive) Immunsystem. Dieses wird in zwei Teilsysteme unterteilt: das angeborene IS und das adaptive IS.

Das angeborene IS ist zuständig für eine schnelle erste Reaktion, die innerhalb der ersten Stunden anspringt. Seine Aufgabe ist es, die anfängliche Infektion einzudämmen und dem adaptiven System Zeit zu geben, eine entsprechende passgenauere Reaktion auszubilden. Das angeborene IS ist statisch und kann sich nicht an spezielle Pathogene anpassen. Im Prinzip könnte man bestehende Sicherheitsmechanismen von Computer-/Netzwerksystemen als die Schichten bis inklusive des angeborenen IS ansehen. Deswe-

gen interessieren wir uns mehr für das adaptive Immunsystem.

2.1 Das adaptive Immunsystem

Das adaptive IS zeichnet sich dadurch aus, dass es lernt, spezielle Pathogene zu erkennen und sich ein Gedächtnis über bereits gesehene Pathogene zu behalten, um gegen zukünftige Infektionen schneller vorgehen zu können. Die Reaktion des adaptiven IS wird unterteilt in die primäre und die sekundäre Reaktion. Die primäre Reaktion stellt die eigentliche Lernphase dar. Sie ist relativ langsam und benötigt bis zu drei Wochen, um eine Infektion zu beseitigen. Das IS stellt sich dabei auf neue Pathogene ein und bildet passende Abwehrmaßnahmen aus. Diese merkt sich das IS und kann damit auf zukünftige Infektionen durch den gleichen Erreger mit der viel schnelleren und effektiveren sekundären Reaktion abwehren. Dieses Gedächtnis kann für manche Erreger über die gesamte Lebensdauer des Organismus bewahrt werden. Außerdem kann eine Infektion damit in vielen Fällen so schnell bekämpft werden, dass nicht einmal Symptome der Infektion sichtbar werden. Die sekundäre Reaktion kann dann außerdem gegen ähnliche, aber dennoch leicht andere Pathogene vorgehen. Dieser Tatsache liegt das Prinzip der Impfung zugrunde. Wurde das Immunsystem z.B. einmal mit einer gutartigen Version des Pocken-Virus konfrontiert („cowpox“ in Abbildung 1), kann es in Zukunft eine sekundäre Reaktion auf die bösartige Variante ausbilden („smallpox“).

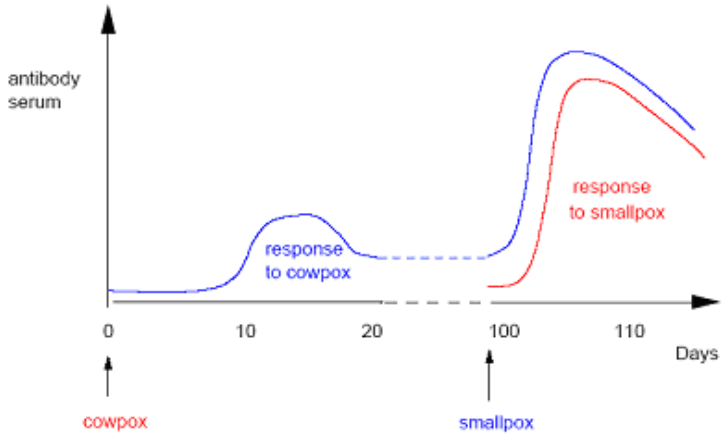

Abbildung 1: Prinzip der Immunisierung [7]: blaue Linie bis 20 Tage: primäre Reaktion, blaue Linie danach: sekundäre Reaktion auf identischen Erreger; rote Linie: sekundäre Reaktion auf ähnlichen Erreger

Hauptbestandteil des adaptiven IS sind die weißen Blutkörperchen (Lymphozyten). Diese stellen die Detektoren des Systems dar. Sie funktionieren vollständig unabhängig voneinander und sind quasi über den gesamten Organismus verteilt. Damit kommt das System ohne eine zentrale Kontrolle aus.

Das Erkennen von Pathogenen geschieht durch chemische Bindung. Jede erkennende Zelle des Immunsystems hat an ihrer Oberfläche viele, gleichartige Rezeptoren. Diese

können sich an bestimmte Stellen an der Oberfläche eines Erregers binden (Fragmente von Proteinen, sogenannte Peptide) . Je komplementärer und je unterschiedlicher elektrisch geladen die Bindungspartner sind, desto höher ist die Wahrscheinlichkeit, dass tatsächlich eine Bindung eingegangen wird (Affinität). Durch die Gleichartigkeit der Rezeptoren ist ein Lymphozyt spezifisch auf die Erkennung eines Erregers ausgelegt. Die Anzahl der zustande gekommenen Bindungen kann als Maß für die Affinität dienen. Damit ist es ihr möglich, einzuschätzen, wie viele Erreger sich tatsächlich in ihrer näheren Umgebung befinden. Tatsächlich wird ein Lymphozyt sogar erst bei Überschreitung eines gewissen Schwellwerts aktiviert.

2.2 Wie Rezeptoren-Vielfalt erzeugt wird

Um über Lymphozyten mit passenden Rezeptoren für jeden möglichen Erreger zu verfügen, müsste das Immunsystem 10^{16} [7] unterschiedliche Arten bereithalten. Dies kann der Organismus nicht leisten. Deswegen behilft sich die Biologie damit, dass sie Detektoren in einem pseudo-zufälligen Klon- und Mutationsprozess erzeugt. Das heißt, es entsteht „zufälligerweise“ ein Lymphozyt mit passenden Rezeptoren.

Mit dieser Vorgehensweise hält der Körper permanent etwa 10^8 [10] unterschiedliche Arten von Lymphozyten bereit. Um dennoch das breite Spektrum der möglichen Pathogen-Proteine abdecken zu können, werden die Detektoren permanent durch neu (zufällig) generierte ersetzt. Auf diese Weise frischt das IS sein Repertoire innerhalb von zehn Tagen komplett auf. Damit schafft es der Körper, innerhalb ein paar Wochen alle möglichen Proteine abzudecken (vergleiche primäre Reaktion).

2.3 Tolerierung der Detektoren

Nun kann aber ein Detektor mit Rezeptoren entstehen, der mit Proteinen des eigenen Organismus eine Bindung eingeht, diese also „erkennt“ (Autoimmunität). Um dies zu verhindern, wird ein Detektor, wenn er an körpereigene Proteine bindet, durch programmierten Zelltod vernichtet. Dieses Vorgehen bezeichnet man als „negative Selektion“ und Tolerierung, da die Detektoren körpereigene Zellen tolerieren. Für die gewöhnlichen Detektoren, die B-Zellen, passiert dies im Knochenmark. Da dort aber nicht alle körpereigenen Proteine vorkommen, reicht das nicht aus.

2.4 Kostimulation

Weil damit auch autoimmune Detektoren in Umlauf geraten könnten, bedient sich das IS zusätzlich dem Prinzip der „Kostimulation“. B-Zellen können nur aktiviert werden, wenn sie zwei „Signale“ empfangen: einmal, dass genug Bindungen stattgefunden haben (Schwellenwert), und ein anderes, das von Th-Zellen („T-helper-cells“) bereitgestellt wird. Diese werden, bevor sie in den Kreislauf des Organismus entlassen werden, durch eine Zensierungsstelle geschickt. Diese stellt der Thymus dar (ein Organ hinter dem Brustbein). In ihm hält der Körper fast alle Proteine bereit, die körpereigen (also „self“) sind.

Da aber auch im Thymus evtl. ein paar körpereigene Proteine nicht vorkommen, kann es trotzdem passieren, dass Autoimmunität auftritt. Hier spielt wieder Kostimulation eine Rolle. Das erste Signal ist wieder eine Überschreitung des Schwellenwertes der Bindungen, das zweite wird vom angeborenen IS erbracht in Gebieten, in denen das Zellgewebe geschädigt ist (viele Zelltode). Dadurch könnte eine autoreaktive Th-Zelle in solchen Bereichen überleben, das ist aber nicht so schlimm, da davon ausgegangen wird, dass gesundes „Körpereigenes“ häufiger auftritt als ungesundes.

3 Warum ein Immunsystem für Computer?

Es ist nicht schwer zu sehen, dass ein solches Immunsystem einige Eigenschaften hat, die wir uns für die Sicherung unserer Computersysteme und Netzwerke wünschen. Interessant sind vor allem:

1. Vielzahl an Einzelkomponenten
   Die enorme Vielzahl der Einzelkomponenten des IS macht jede einzelne entbehrlich. So führen Ausfälle von Teilkomponenten des Systems nicht dazu, dass Schutzmechanismen ausfallen.

2. Dezentrale Organisation
   Durch das hohe Maß an Verteilung der Abwehrmechanismen über den gesamten Organismus, hat dieser keinen Punkt, an dem, wenn er unterlaufen wird, das gesamte Immunsystem zusammenbricht.

3. Vielfältigkeit
   Es ist von essentieller Wichtigkeit, dass ein Angreifer, falls er die Sicherheitsmechanismen für ein einzelnes System unterlaufen hat, nicht auf gleiche Weise in (viele) andere eindringen kann. Dies bewerkstelligt ein biologisches IS dadurch, dass jeder Organismus sein eigenes, nahezu einzigartiges IS ausbildet. Dies wäre auch sehr wünschenswert für Computersysteme.

4. Implizite Spezifikation einer Sicherheitsrichtlinie
   Die Unterscheidung zwischen körpereigen und Antigen erübrigt die explizite Definition einer Sicherheitsrichtlinie per Hand.

4 Realisierung

Das Hauptproblem besteht in der Frage, wie man „körpereigen“ bzw. „systemeigen“ definieren möchte. S. Forrest und S. Hofmeyr zeigen in [4] folgende Möglichkeiten auf: Speicherzugriffe auf einem System, Systemaufrufe auf dem Kernel oder Befehlssequenzen während der Ausführung eines Programms, Verhaltensweisen von Benutzern, Muster in den Tastatureingaben, TCP/IP-Pakete die einen Einzelplatzrechner erreichen oder verlassen, den Netzwerk-Verkehr durch Netzwerkkomponenten (z.B. Router, Gateway) oder das kollektive Verhalten aller Komponenten in einem Netzwerk.

An diese Definition von „systemeigen“ wird die Forderung gestellt, dass sie legitime Veränderungen am System akzeptieren muss: Benutzer bearbeiten Dateien, neue Software wird installiert, neue Benutzer werden angelegt oder Benutzer verändern ihr Verhalten (z.B. ein Benutzer wechselt die Abteilung), sowie Routinearbeiten (des Systemadministrators).

4.1 Schutz für Einzelplatzsysteme

S. Forrest, S.Hofmeyr et. al. haben in [4] herausgearbeitet, dass sich Systemaufrufe auf dem Systemkernel für die Definition des eigenen Einzelplatzsystems als sehr geeignet erweisen. Veränderungen hierbei stellen quasi einen Seiteneffekt eines Angriffs dar [11]. In einer Trainingsphase werden Systemaufrufe eines Prozesses (Programms) mitgelesen und kurze Sequenzen davon werden in einer Datenbank gesammelt. [4] beschränkt sich auf privilegierte Prozesse, die z.B. unter Unix als root laufen. Damit können diese Programme ohne Einschränkungen das System manipulieren und sind deswegen für Angreifer sehr interessant. Während dieser Trainingsphase nehmen wir an, dass keine Angriffe vorkommen. Neue Sequenzen des normalen Verhaltens („systemeigen“) werden hinzugefügt, bis die Datenbank über längere Zeit nicht mehr weiter wächst. Dann haben wir wohl so ziemlich alle Sequenzen erfasst. So verfahren wir für jedes (kritische) Programm. Wenn das Programm in Zukunft läuft, werden die Systemaufrufe überwacht und ständig mit den in der Datenbank gespeicherten Sequenzen abgeglichen. Wenn keine Entsprechung gefunden wird, handelt es sich höchstwahrscheinlich um eine Anomalie (systemfremde Elemente). Deswegen wird dieses Verfahren auch Anomalie-Erkennung genannt.

Eine andere Variante ist, Detektoren ähnlich dem pseudo-zufälligen Prozess zu generieren. Statt genetischer Information könnten wir zufällig Strings erzeugen. Darauf können wir einen negativen Selektions-Prozess anwenden, wie sie das biologische Immunsystem anwendet. Zufällig erzeugte Detektoren werden erst nach einer Reifephase aktiviert, in der sie nicht auf die Definition von „systemeigen“ ansprechen dürfen. Tun sie das dennoch, werden sie verworfen und durch neu erzeugte ersetzt. Alle, die die Reifephase überstehen, sollten somit systemeigene Elemente tolerieren.

Dann überwachen wir die zu schützenden Programme/Daten mit den so gewonnenen Detektoren. Die Bindung des Detektors an Proteine eines Erregers im biologischen System wird im Computer als Vergleich von zwei Strings dargestellt. Ein perfekter Treffer wären zwei zeichenweise identische Strings. Jedoch verwendet das biologische IS unvollständige Vergleiche, um einige seiner Ziele zu erreichen (z.B. wird so die Impfung ermöglicht, siehe „Immunsystem“). Also definieren wir uns eine Anzahl an kontinuierlichen Bits, die mindestens übereinstimmen müssen, damit ein Treffer registriert wird (so wäre z.B. für die Zahl 3 Treffer(„abcde“, „cdefg“)=true). D.h. ein bestimmter Detektor kann nicht nur genau eine Anomalie feststellen, sondern einen kleinen Teilbereich, wie das lebende System.

Wenn wir die Datenbank aufbauen oder später die Systemaufrufe überwachen, schieben wir immer ein „Fenster“ über die Abfolge der Aufrufe („Sliding-Window-Algorithmus“). D.h. wir betrachten immer eine fixe Anzahl unmittelbar aufeinander folgender Aufrufe.

Diese Vorgehensweise ist in Abbildung 2 dargestellt; dort hat das Fenster die Größe drei. In der Grafik wäre bei der Überwachung eine Anomalie festgestellt worden (rot), wenn man den Schwellenwert auf zwei festgesetzt hätte. Der von „read“ auf „mmap“ geänderte Systemaufruf verursacht im Beispiel zwei Erkennungstreffer. Wäre die Sequenz der Aufrufe länger, könnten es sogar drei sein (maximal natürlich die Länge des Fensters, ein „perfekter“ Treffer).

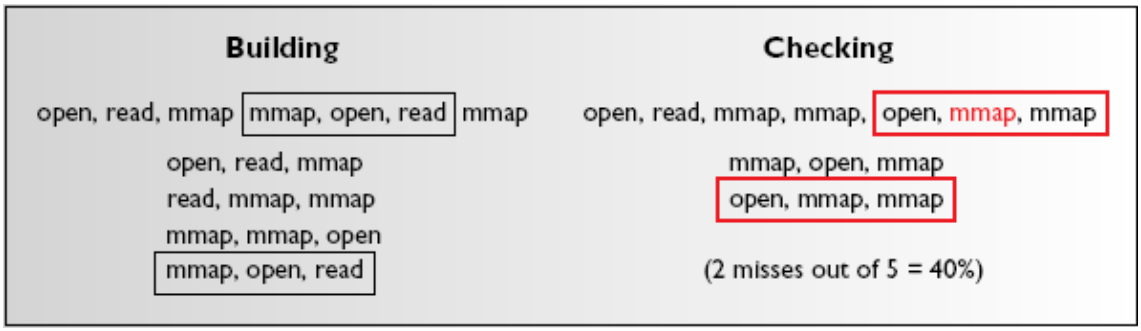

Abbildung 2: aus [4]: Der Sliding-Window-Algorithmus

Damit ist es aber, wie im biologischen System, nicht möglich, alle Fälle abzudecken, weil die dafür notwendigen Zahlen unsere Ressourcen übersteigen würden. Deswegen gibt es zu jedem einzelnen Zeitpunkt „Löcher“ im Schutz durch die Detektoren. Man möchte deswegen die Tatsache modellieren, dass im biologischen System ein ständiger Austausch der Detektoren stattfindet, um so mit der Zeit fast alle möglichen Anomalien feststellen zu können. Dies verhindert, dass Angreifer bekannt gewordene Löcher (wiederholt) ausnutzen. Dazu versieht man die Detektoren mit einer endlichen Lebensdauer. Danach werden sie durch neue, zufällig generierte, ersetzt. Jedoch muss man hier abwägen, wie hoch man diese Zeitspanne festsetzt: wenn man diese sehr niedrig ansetzt, verringert man die Gefahr, dass Löcher ausgenutzt werden. Allerdings befindet sich dann ein relativ hoher Anteil der Detektoren in der Reifephase, wo sie nicht zum Schutz beitragen können [3]. Wenn aber Detektoren in der Vergangenheit auf Anomalien reagiert haben, so werden diese mit einer erheblich längeren Lebensdauer versehen, außerdem wird der Schwellenwert für diese herabgesetzt. Damit setzen wir die sekundäre Reaktion um. Der Vorteil an diesem Vorgehen ist, dass wir uns quasi automatisch Signaturen von Angriffen merken können (im Gegensatz zu klassischen Systemen, wo diese von Hand codiert werden müssen).

Jedoch werden wir vielleicht nicht ganz ohne Handarbeit auskommen. Ein Entscheidendes Merkmal eines künstlichen IS ist die Rate der „false positives“, also der legitimen Aktionen, die vom IS verhindert werden, weil Detektoren darauf ansprechen. Um diese zu senken, können wir das Prinzip der Kostimulation umsetzen. Wenn der Schwellenwert überschritten wird und ein Erkennungstreffer registriert wird (welcher zu einer Reaktion des IS führen würde), so wird ein menschlicher Administrator informiert. Wenn dieser bestätigt, dass es sich tatsächlich um einen Angriff handelt, dann stellt er sozusagen das entsprechende Kostimulations-Signal bereit. Das Problem hierbei könnte sein, dass ein

Administrator aufgrund anderer Tätigkeiten oder Abwesenheit nicht sofort auf solche Anfragen reagieren kann. Es ist die Frage, ob wir dann die Aktivität vorsorglich unterbinden und als falsch positiv in Kauf nehmen wollen (d.h. es tritt eine autoimmune Reaktion auf).

4.2 Schutz für Computer-Netzwerke

Selbstverständlich möchte man nicht nur Einzelplatzsysteme besser schützen, sondern die verteilten Mechanismen eines Immunsystems auch für Netzwerke ausnutzen [6]. Denn alle Angriffe, die nicht über die Konsole des Einzelsystems erfolgen, können nur über das Netzwerk erfolgen. Dabei wird sogar nur eine Implementierung benötigt, unabhängig von der Plattform, auf der die Einzelplatzrechner laufen, da Netzwerkverkehr immer über standardisierte Protokolle abläuft (meistens TCP/IP). Es gibt zwei Arten von Angriffen über eine Netzwerkverbindung: Angriffe auf einen einzelnen Rechner, die Sicherheitslücken in Serversoftware oder Betriebssystemen ausnutzen, und Angriffe, die auf das Netzwerk insgesamt zielen (z.B. indem sie Schwachstellen der Protokolle ausnutzen). Angriffe auf einzelne Computer sind bereits durch das vorhergehende Kapitel abgedeckt. Deswegen reicht es, sich nun auf Angriffe auf das Netzwerk an sich zu konzentrieren. Typischerweise weiß ein Angreifer nicht sehr viel über das Zielnetzwerk und muss durch gewisse Tests mehr in Erfahrung bringen. Damit ändert er aber den Netzwerk-Verkehr.

Der TCP/IP-Verkehr im Netzwerk wird der Einfachheit halber durch sogenannte Datenpfad-Tripel repräsentiert. Diese bestehen aus einem drei-Tupel der Form (IP-Adresse des Quellcomputers, IP-Adresse des Zielcomputers, Netzwerkdienst), siehe Abb. 3. Dieses Tripel wird nur einmal pro Verbindung erzeugt (d.h. während der Kommunikation über diese Verbindung entstehen keine neuen Tripel). Anhand dieser Tripel können wir wieder eine Definition des normalen Verhaltens des Netzwerks konstruieren, um dann Anomalie-Erkennung darauf anzuwenden.

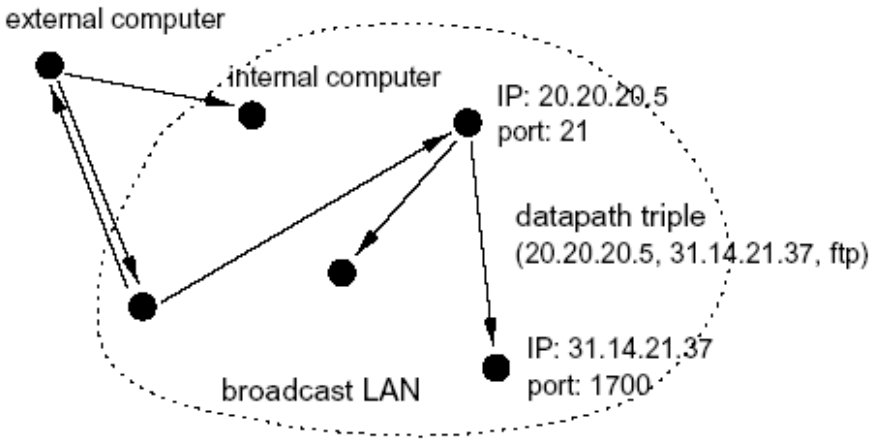

Abbildung 3: aus [6]: Charakterisierung des Netzwerk-Verkehrs über Datenpfad-Tripel

Dazu erzeugt man wieder, mit dem gleichen Verfahren wie für unsere Einzelplatzsysteme, Detektoren. Diese überwachen nun aber statt Systemaufrufen die ein- und ausgehenden Datenpfade auf einem Rechner oder einer Netzwerkkomponente (z.B. Router). Die Detektoren könnten in einem broadcast-fähigen LAN-Netzwerk (z.B. Ethernet) zentral über einen Server generiert werden, um die Clients nicht mit Rechenaufwand zu belasten. Außerdem könnte hier zentral zensiert/toleriert werden (negative Selektion). Die ausgereiften Detektoren werden dann auf die Einzelplatzrechner verteilt. Damit kann jeder einzelne Rechner verschiedenartige Angriffe auf das Netzwerk erkennen. Außerdem ist das System robust, da der Ausfall von einzelnen Detektoren oder Rechnern den Schutz nicht erheblich beeinträchtigt. Die Erzeugung von Detektoren könnte man im Notfall auch von den Client-Rechnern aus bewerkstelligen, sollte der Server ausfallen. Wieder werden die Sicherheitsrichtlinien implizit erstellt, da sich diese aus dem tatsächlichen Verhalten des Netzwerks zusammensetzen, und nicht aus dem, was sie eigentlich sein sollten. Um die kontinuierliche Erneuerung des Detektorenvorrats zu beschleunigen, könnten Leerlaufphasen des Netzwerks genutzt werden, um mehr Detektoren zu erzeugen und damit mehr unterschiedliche Angriffe innerhalb kürzerer Zeit erkennen zu können.

Dieses System ist gut skalierbar, da die bisherige Funktion ohne Kommunikation unter den Einzelkomponenten möglich ist. Diese arbeiten vollkommen unabhängig.

4.3 Andere Ansätze

Es existieren einige andere oder erweiternde Ansätze. U.A. sind dies:

- **Autonomous Agents** [12], [5], [2]. Hier sind Agenten jeweils unabhängig für eine Teilkomponente (Prozesse, Eingabegeräte, Netzwerkkomponenten) des Systems verantwortlich und schützen diese.
- **Data Mining und Machine Learning** [9], [5]. Hier wird das (normale/anomale) Verhalten über Data-Mining-Techniken aus den von heutigen Betriebssystemen bereits bereitgestellten Protokoll-Informationen (log-Dateien) extrahiert.
- **Danger Model** [1]. Auch hier werden die Protokoll-Dateien ausgewertet und Zuständen zugeordnet. Dann wird entschieden, ob sich das System in einem gefährlichen Zustand befindet. Das heißt, gefährlich ist, was einen negativen Effekt (z.B. viel Ressourcenverbrauch) auf das System hat.
- **Time Series Prediction, Advanced State Detection**, [1] Aus dem Verlauf der Protokoll-Dateien wird versucht, mit Regeln daraus abzuleiten und damit das Verhalten des Systems zu beschreiben. Dann kann man mit gewissen Methoden vorhersagen, in welchen Zustand sich der aktuelle Systemzustand entwickeln wird. Mit dieser Vorgehensweise wäre es somit möglich, gegenzusteuern, bevor es zu einer kritischen Situation kommt.

4.4 Auswirkungen der Umsetzung auf Computersysteme

Experimente haben gezeigt [6], dass nach dem obigen Verfahren zur Anomalie-Erkennung erzeugte Datenbanken mit systemeigenen Sequenzen sich auf unterschiedlichen Systemen zu mindestens 40% voneinander unterschieden, selbst wenn es sich um die gleichen Prozesse handelte (z.B. sendmail). Damit erfüllt die Definition von „systemeigen“ das Prinzip der Verschiedenheit.

Das Prinzip der Kostimulation (im Fall von Netzwerken z.B. gegenseitig durch Detektoren) sorgt dafür, dass ein leicht geändertes Benutzerverhalten einer Person nicht in einer (autoimmunen) Abwehrreaktion des Immunsystems resultiert. Lediglich wenn mehrere Benutzer plötzlich und auf die gleiche Art ihr Verhalten auf erhebliche Weise ändern (was sehr unwahrscheinlich ist), würde eine Reaktion hervorgerufen werden.

5 Unterschiede

Wir können aber nicht 1:1 die Natur nachbilden. Es ist die Frage, ob einige Eigenschaften des biologischen Immunsystems überhaupt adäquat umgesetzt werden können. Außerdem stellen Computersysteme zusätzliche Anforderungen an ein Immunsystem, wie z.B. Zugriffsrechte.

5.1 Was nicht modelliert werden kann

Mit Computersystemen können wir teilweise nicht mit so großen Stückzahlen hantieren, wie biologische Systeme. Es ist deswegen die Frage, ob alle der dargestellten Verfahren im Rechner überhaupt angewendet werden können, da biologische Systeme einige der Ziele nur durch eine überwältigende Anzahl an z.B. Detektoren erreichen ([1], S.10f). Diese permanent in Umlauf zu bringen, könnte die Performance des Systems im laufenden Betrieb evtl. stark beeinträchtigen.

5.2 Zusätzliche Anforderungen

In einem Informationssystem muss verhindert werden, dass Benutzer unberechtigter Weise an vertrauliche Daten gelangen, zu denen Sie keine Zugangsberechtigung haben. Biologische Systeme kennen diese Problematik nicht - sie verteilen sogar permanent ihre genetische Information (vergleiche Verbrechensaufklärung mittels Hautschuppen etc.).

Zusätzlich zur biologischen Variante des Immunsystems können wir ein Computersystem vor böswilligen Benutzern schützen, die sich als legitim ausgeben (z.B. durch Ausspionieren des Passwortes). Dies ist auch notwendig, da dieses Vorgehen gängige Praxis zum Einsteigen in ein Computersystem ist. Da Angreifer normalerweise ein anderes Benutzerverhalten an den Tag legen (denn Sie sind vermutlich in das System eingestiegen, um Schaden anzurichten oder Daten auszuspionieren), wird eine Reaktion des Immunsystems ausgelöst.

6 Vergleich mit herkömmlichen Systemen

Herkömmliche, Signatur-basierte Systeme erkannten bei Tests am MIT laut [6] zwischen 50% und 70% der Attacken. Dagegen können nach [4],[6] Anomalie-Erkennungs-basierte Systeme mit bis zu 90-100% Wahrscheinlichkeit neue Attacken erkennen. Der entscheidende Faktor ist hier die Rate der als falsch positiv eingestuften Aktivitäten. Näher experimentell untersucht hat dies [8], Table 4, S. 461. Dabei konnten sogar alle Attacken erkannt werden (7 von 7). Allerdings wurde dafür eine recht hohe Zahl an falschen positiven Einstufungen in Kauf genommen (1,76, d.h. etwa 25%). Beachtet werden muss aber, dass Signatur-basierte, herkömmliche Systeme manche Attacken (die fehlenden 30-50%) gar nicht erkennen können. Anomalie-Erkennung kann aber im Prinzip alle mögliche Angriffe zu einer gewissen Wahrscheinlichkeit erkennen. Damit können Würmer wie Sasser bei herkömmlichen Systemen auf vielen Systemen gleichzeitig Schaden anrichten. Jedoch könnten mit einem Immunsystem ausgestattete Systeme zumindest zu einem gewissen Bruchteil geschützt werden.

7 Nachteile und mögliche Gefahren

Trotz aller Vorteile, die ein Immunsystem für Computer bringen kann, gibt es noch Probleme, über die man sich Gedanken machen sollte, bevor solch ein System zum Einsatz kommen kann.

1. Im Prinzip könnte ein Angreifer das System unterlaufen, indem er über einen längeren Zeitraum verteilt kleine Eingriffe am System vornimmt, die vom IS als Schwankung des systemeigenen Verhaltens interpretiert werden.

2. Computer/Netzwerke sind dynamische Systeme. Es müssen ständig neue Komponenten hinzugefügt und neue Softwareversionen aufgespielt werden. Dies ist in der Natur jedoch nicht vorgesehen. Deswegen müssten wir bei einem Computer-IS, das IS ähnlich wie bei einer Transplantation „außer Kraft“ setzen. Dann stehen jedoch Angreifern Tür und Tor offen (das System akzeptiert plötzlich „systemfremd“ als „systemeigen“). Eine Möglichkeit wäre, nur Detektoren für gewisse Einzelkomponenten (z.B. den „sense of self“ für sendmail) auszuschalten. Damit wäre nur die betroffene Komponente potentiell gefährdet. Auf jeden Fall aber muss sich das System an die neuen Komponenten gewöhnen, d.h. es muss eine Definition des normalen Verhaltens dieser Komponente erstellen, was über eine gewisse Zeitspanne die Leistung bei der Benutzung des Systems beeinträchtigt (bis sich an der Definition für „systemeigen“ nichts mehr ändert). Zudem sollte in dieser Zeitspanne kein Angriff stattfinden.

3. Angenommen, ein böswilliger Benutzer hat sich den Zugang (d.h. Benutzernamen und Passwort) des legitimen Benutzers beschafft. Das IS reagiert, während er auffällige Änderungen am System vornimmt, und sperrt den Zugang. Dadurch stellt sich das Problem, wie der legitime Benutzer seinen Zugang wieder erlangen /

den Computer entsperren kann. Sein Zugang kann dazu offensichtlicherweise nicht mehr dienen.

4. Man sollte aufpassen, dass das System sich nicht irgendwann (zu oft) gegen den legitimen Benutzer wendet (wie in etlichen Science-Fiction-Werken dargestellt), und damit seinen Sinn und Zeck nicht mehr erfüllt.

8 Zusammenfassung und Ausblick

Das Immunsystem aus der Natur zu analysieren und in entsprechender Form auf Computersysteme zu übertragen könnte sicherlich einen enormen Zuwachs an Sicherheit bringen. Es wurde gezeigt, dass Anomalie-Erkennung auf Computersysteme anwendbar ist. Die vorgestellten Realisierungen sind nicht zu komplex und handlich.

Allerdings gibt es noch einige Detailfragen zu klären. Vor allem ist fraglich, ob der Idealzustand, ohne eine menschliche letzte Entscheidungsinstanz (in Form eines Administrators) auszukommen, überhaupt erreicht werden kann, die entscheidet, ob es sich tatsächlich um Angriffe oder „false positives“ handelt. Ich bezweifle dies. Zu klären wird auch sein, ob ein dargestellter Schutz auf ein gesamtes System (d.h. auf alle Systemprozesse) ausgeweitet werden kann, oder ob es tatsächlich reicht, nur gewisse Prozesse zu schützen. Da bisher entsprechende Implementierungen nur für einzelne Prozesse existieren (bzw. dokumentiert sind), können über das tatsächliche Verhalten eines gesamten Systems nur begrenzt Aussagen gemacht werden.

Literatur

[1] M. Burgess. Computer immunology. *Proceedings of the Twelth Systems Administration Conference (LISA XII) (USENIX Association: Berkeley, CA)*, page 283, 1998.

[2] M. Crosbie and G. Spafford. Defending a computer system using autonomous agents. In *8th National Information Systems Security Conference*, 1995.

[3] S. Forrest and S. Hofmeyr. Engineering an immune system. *Graft*, 4(5):5–9, 2001.

[4] Stephanie Forrest, Steven A. Hofmeyr, and Anil Somayaji. Computer immunology. *Communications of the ACM*, 40(10):88–96, 1997.

[5] G. Helmer, J. Wong, V. Honavar, and L. Miller. Intelligent agents for intrusion detection. In *IEEE Information Technology Conference*, pages 121–124, 1998.

[6] S. Hofmeyr. A immunological model of distributed detection and its application to computer security, 1999.

[7] Steven A. Hofmeyr. An Interpretative Introduction to the Immune System. Technical report, University of New Mexico. To Appear in Design Principles for the

Immune System and other Distributed Autonomous Systems. Oxford University Press, Eds, I. Cohen and L. Segel. 2000.

[8] Steven A. Hofmeyr and Stephanie Forrest. Architecture for an artificial immune system. *Evolutionary Computation*, 8(4):443–473, 2000.

[9] Wenke Lee and Salvatore Stolfo. Data mining approaches for intrusion detection. In *Proceedings of the 7th USENIX Security Symposium*, San Antonio, TX, 1998.

[10] Anil Somayaji, Steven Hofmeyr, and Stephanie Forrest. Principles of a computer immune system. In *Meeting on New Security Paradigms, 23-26 Sept. 1997, Langdale, UK*, pages 75–82. New York, NY, USA : ACM, 1998.

[11] M. Stillerman, C. Marceau, and M. Stillman. Intrusion Detection for Distributed Applications. *Communications of the ACM*, 42(7):62–69, July 1999.

[12] A. Watkins. An immunological approach to intrusion detection. In *Proceedings of the 12th Annual Canadian Information Technology Security Symposium.*, 2000.