# Entwicklung eines Systems zur Ermittlung von Bedrohungspotentialen in vernetzten Systemen

Thomas Hungenberg und Stephan Baum
{th|sb}@linast.de

Fachbereich Angewandte Informatik
Fachhochschule Bonn-Rhein-Sieg
D-53757 Sankt Augustin

Betreuer: Prof. Dr. Karl W. Neunast, Prof. Dr. Manfred Kaul

## Zusammenfassung

Diese Arbeit beschreibt den Entwurf und die prototypische Implementierung eines Systems zur Simulation und Visualisierung von Bedrohungspotentialen in vernetzten Umgebungen. Dabei handelt es sich um ein flexibel konfigurierbares und erweiterbares Werkzeug, welches die *symptomatische* Simulation und die Darstellung der *prinzipiellen* Wirkungsweise von Bedrohungen sowie möglicher Sicherungsmaßnahmen in beliebigen Netzwerkkonfigurationen ermöglicht. Der Einsatz dieses Systems erlaubt einer Person mit entsprechendem Fachwissen auf dem Gebiet der IT-Sicherheit, die Bedeutung der möglichen Gefahren für ein vernetztes System und die in ihm gespeicherten Daten einer anderen Person, die nicht über das nötige Know-how verfügt, auf verständliche Art und Weise zu vermitteln.

## 1 Motivation

Administratoren und technisch versierten Anwendern ist bekannt, daß innerhalb von Datennetzen Bedrohungen existieren, die sich in unerwünschter Art und Weise auf die vernetzten Systeme und die Netzwerkinfrastruktur auswirken können. Insbesondere sind neben prinzipiellen Software-Schwächen auch die kritischen Konfigurationen bekannt, die oft sogar als Voreinstellungen ausgeliefert werden und zum Beispiel die Verbreitung von Schadensroutinen per E-Mail erlauben bzw. begünstigen. Die Praxis zeigt jedoch, daß dieses Wissen um vorhandene Schwachstellen bisher nicht in ausreichender Weise in geeignete Sicherungsmaßnahmen umgesetzt wird.

Die Abschätzung möglicher Risiken für die Infrastruktur einer Firma oder Institution und die Ergreifung geeigneter Sicherungsmaßnahmen sind wichtige Aufgaben des Managements. Wenn also vernetzte Systeme durch Bedrohungen gefährdet sind, deren prinzipielle Funktionsweisen bekannt sind, läßt dies auf Versäumnisse seitens des Managements schließen. Ein wesentlicher Grund dafür kann unzureichendes Verständnis oder mangelndes Interesse für technische Problemstellungen dieser Art sein — es fehlt am notwendigen Problembewußtsein. Dies ist unter anderem darin begründet, daß oft umfangreiches technisches Fachwissen erforderlich ist, um potentielle Bedrohungen zu verstehen und die Auswirkungen auf die eigenen Systeme einschätzen zu können.

Eine Software, die es erlaubt, ein Bedrohungsszenario — also das Wirken einer bestimmten Bedrohung in einem Netzwerk — zu simulieren und zu visualisieren, kann ein geeignetes Werkzeug sein, um auch Personen mit geringerem technischen Vorwissen die Auswirkungen der simulierten Bedrohung zu verdeutlichen und sie bezüglich der Möglichkeiten zur Minimierung von Bedrohungspotentialen zu beraten (vgl. Use-Case-Diagramm in Abb. 1).

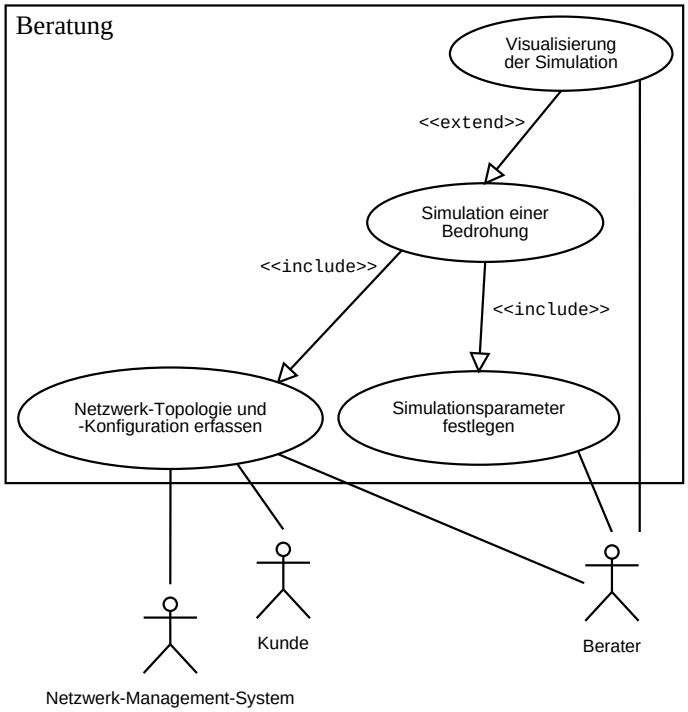


Abbildung 1: Beratung – der typische Anwendungsfall

# 2 Stand der Technik

Nach der Definition der Anforderungen an eine solche Software wurde der Stand der Technik analysiert, um zu prüfen, welche bereits existierenden Lösungen bei seiner Entwicklung eingesetzt werden könnten. Dabei wurden verschiedene Werkzeuge zur Simulation von Netzwerken sowie zur Visualisierung von Netzwerkkonfigurationen untersucht (u. a. wurden die in [Baj99], [Coh99], [Dup88], [Est96], [Est99], [Fal99] und [Kes88] beschriebenen Systeme betrachtet).

Es zeigte sich, daß die verfügbaren Programmpakete keine ausreichende Funktionalität bzw. Flexibilität boten, um darauf aufbauend ein den definierten Anforderungen entsprechendes Systems zu entwickeln. Um die Systemfunktionalität wie geplant realisieren zu können, wurde daher beschlossen, ein eigenes Simulations- und Visualisierungswerkzeug zu entwickeln.

# 3 Realisierung

Das entwickelte System zur Simulation von Bedrohungsszenarien ist in drei Schichten unterteilt: Kernsystem, Entkopplungsschicht und Ein-/Ausgabe-Schicht. Diese Strukturierung entspricht dem ’Facade’-Entwurfsmuster mit seinen Elementen ’subsystem’, ’facade’ und ’clients’ [Gam96].

Um eine weitgehende Plattformunabhängigkeit zu gewährleisten, wurde für die prototypische Implementierung des Systems die Programmiersprache Python und über die Schnittstelle Tkinter das GUI-Toolkit von Tcl/Tk verwendet.

## 3.1 Kernsystem

Im Kernsystem findet die eigentliche Simulation der Bedrohungsszenarien statt. Dazu wird zunächst der *statische* Zustand des zu betrachtenden Netzwerks — bestehend aus der Netzwerk-Topologie und der Konfiguration der Netzwerk-Knoten — erfaßt.

Ein Netzwerk wird in diesem System als bi-partiter Graph aufgefaßt, der sich aus Knoten- und Netz-Objekten zusammensetzt, die miteinander verbunden sein können. Knoten-Objekte repräsentieren dabei die an ein oder mehrere Netze angeschlossenen 'Geräte' und Netz-Objekte stellen die einzelnen Netzwerk-Segmente — also die physischen Verbindungen zwischen zwei oder mehr Knoten — dar.

Jeder Netzwerkkomponente können beliebige Eigenschaften (z. B. Informationen über die in einem Knoten aktive Software o. ä.) zugewiesen werden, die bei der Simulation einer bestimmten Bedrohung ggf. berücksichtigt werden können. Weiterhin können verschiedene netzwerkspezifische Merkmale der einzelnen Knoten konfiguriert werden, wie z. B. Routing-Tabellen oder Paketfilter-Regeln.

Die zu simulierenden *dynamischen* Vorgänge innerhalb des Netzwerks werden durch verschiedene Klassen modelliert, deren Instanzen als sogenannte Dienst- oder Bedrohungsmodule in einzelne Netzwerkknoten eingebracht werden können. Diese Module können Datenpaket-Objekte innerhalb des Netzwerks versenden bzw. von dem betreffenden Knoten empfangene Datenpakete verarbeiten. Beispielsweise kann einem bestimmten Knoten durch Hinzufügen eines entsprechenden Dienstmoduls innerhalb der Simulation die Funktionalität eines E-Mail-Servers oder -Clients gegeben werden. Auf gleiche Art und Weise kann ein Knoten aber auch mit schadhaften Fähigkeiten — z. B. mit der Routine zur Verbreitung eines Netzwerk-Wurms oder der Funktionalität eines Netzwerk-Sniffers — ausgestattet werden. So können also einerseits verschiedene Netzwerkdienste und andererseits die möglichen Auswirkungen von Bedrohungen auf diese Dienste simuliert werden.

## 3.2 Entkopplungsschicht

Die Entkopplungsschicht trennt die Implementierung des Kernsystems von der zur Verfügung gestellten Funktionalität, indem sie das Kernsystem kapselt und die Steuerung des getakteten Simulationsablaufs übernimmt.

Des weiteren findet in dieser Schicht eine Sammlung und Aufbereitung von Daten über eine ablaufende Simulation statt. Diese Daten können sowohl vom Kernsystem selbst als auch von außen (z. B. zu Visualisierungszwecken) abgerufen und genutzt werden. Außerdem sind in der Entkopplungsschicht Funktionen zur Anbindung von graphischer Bedienoberfläche und Visualisierungsmodulen implementiert.

## 3.3 Ein-/Ausgabe-Schicht

Diese Schicht umfaßt all die Komponenten, über die der Benutzer mit dem System interagieren kann. Dazu gehören insbesondere die graphische Bedienoberfläche (GUI) und die Funktionen zum Einlesen von Konfigurationsdaten in das Kernsystem.

Die GUI bietet dem Benutzer Bildschirmdialoge an, um ein zu simulierendes Szenario in das Kernsystem zu laden und die Simulation zu starten und bei Bedarf wieder zu stoppen. Die Geschwindigkeit des Simulationsablaufs läßt sich den Erfordernissen anpassen. Ein geladenes Szenario wird innerhalb der GUI zweidimensional visualisiert (vgl. Screenshot in Abb. 2). Während der Simulation werden Statusinformationen der einzelnen Netzwerkkomponenten und statistische Angaben permanent oder auf Abruf angezeigt. Die Visualisierung kann in verschiedener Hinsicht angepaßt werden, um spezielle Aspekte der Simulation einer Bedrohung besonders hervorzuheben.

## 3.4 Simulationstaktung

Die Simulation läuft getaktet in der Art und Weise ab, daß zu jedem Zeitpunkt immer nur eine einzelne Netzwerkkomponente aktiv ist. Dadurch ist gewährleistet, daß die Simulation schrittweise ausgeführt werden kann und es nicht zu zeitlich parallelen Ereignissen an verschiedenen Stellen der Topologie

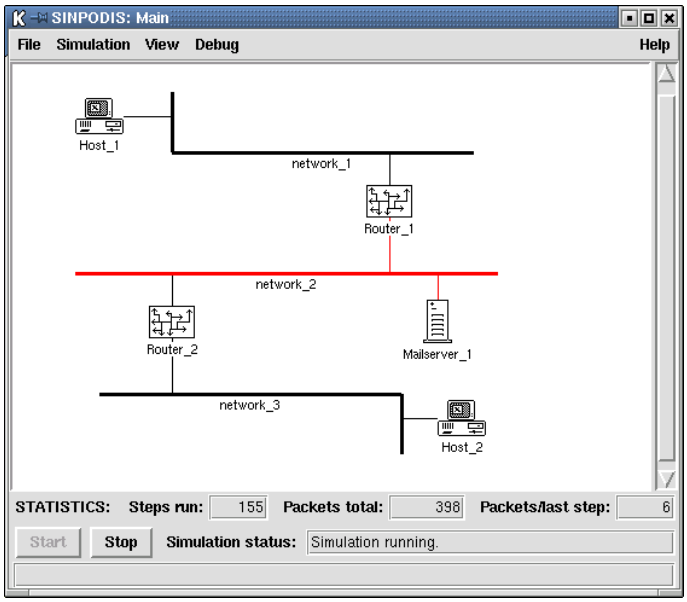


Abbildung 2: Screenshot einer Simulation von Auswirkungen eines E-Mail-Wurms

kommen kann. Im Vergleich zu einer ungetakteten bzw. parallelen Simulationsmethodik erleichtert dies die Beobachtung und Steuerung des Simulationsablaufs durch den Benutzer.

# 4 Ausblick

Bei der im Rahmen dieser Arbeit durchgeführten prototypischen Implementierung lag der Schwerpunkt auf der Bereitstellung eines Rahmensystems, welches vielfältige Möglichkeiten für zukünftige Erweiterungen des Funktionsumfangs bietet. Einige denkbare Erweiterungen werden im folgenden beschrieben.

## 4.1 Zusätzliche Bedrohungsarten und Netzwerkdienste

Aufbauend auf das vorhandene System können durch die Entwicklung entsprechender Module weitere Bedrohungsarten simuliert werden. Um die Auswirkungen bestimmter Bedrohungsarten zeigen zu können, ist die Implementierung von Dienstmodulen zur Simulation weiterer Netzwerkdienste wünschenswert.

## 4.2 Erweiterte Möglichkeiten zur Übernahme von Eingabedaten aus Fremdformaten

Bei der derzeitigen prototypischen Implementierung müssen die Topologiedaten für ein Bedrohungsszenario entweder manuell erstellt werden oder können mit Hilfe eines Dateikonverters aus den von dem Netzwerk-Editor 'Tkined' (Teil der Netzwerkmanagement-Plattform 'Scotty' [Scotty]) gespeicherten Daten gewonnen werden. Die Konfigurationsdaten für die Netzwerkkomponenten eines Bedrohungsszenarios müssen derzeit manuell erstellt werden.

Eine wünschenswerte Erweiterung wäre neben der Entwicklung weiterer Dateikonverter zur *indirekten* Übernahme von Daten anderer Netzwerk-Editoren auch eine Möglichkeit zur *direkten* Datenübernahme von Netzwerk-Managementsystemen, was eine flexible Erfassung der Struktur und Konfiguration realer Netzwerke ermöglichen würde.

### 4.3 Szenario-Editor

Eine Erweiterung, welche die Benutzung des Systems erleichtern würde, wäre die Integration eines vollständigen Szenario-Editors, der die Erstellung und Modifikation von Bedrohungsszenarien direkt über die graphische Bedienoberfläche des Systems ermöglicht. Eine konsequente Plausibilitätsprüfung könnte dabei helfen, mögliche Fehler in der Konfiguration zu vermeiden.

### 4.4 Erweiterte Visualisierung

Im Bereich der Visualisierung sind zahlreiche Erweiterungen möglich und zum produktiven Einsatz des Systems für Beratungszwecke teilweise auch erforderlich, um dem Beobachter einer Simulation die darin ablaufenden Vorgänge möglichst intuitiv veranschaulichen zu können.

Denkbare Ansätze für erweiterte Visualisierungsmöglichkeiten sind z. B. die detaillierte Anzeige der *innerhalb* einer Netzwerkkomponente simulierten Vorgänge, die dreidimensionale Anzeige des Bedrohungsszenarios (ggf. in verschiedenen Detaillierungsgraden), die Animation des Netzwerk-Verkehrs in verschiedenen Detaillierungsgraden (evtl. bis auf die Ebene einzelner Datenpakete) oder auch die realitätsnahe Darstellung des Netzwerk-Umfelds (also z. B. der Räume oder Gebäude, in denen die realen Netzwerkkomponenten untergebracht sind).

## Literatur


[Baj99] Bajaj, Sandeep et al.: „Improving Simulation for Network Research“, Technical Report 99-702, University of Southern California, März 1999
`http://www.isi.edu/~johnh/PAPERS/Bajaj99a.ps.gz`

[Coh99] Cohen, Fred: „Simulating Network Security“, Network Security, April 1999
`http://www.all.net/journal/netsec/9904.html`

[Dup88] Dupuy, Alexander, David F. Bacon, Jed Schwartz und Yechiam Yemini: „NEST: A Network Simulation and Prototyping Testbed“, Communications of the ACM, Vol. 33, No. 10, Seite 64-74, Oktober 1990
`http://www.cs.berkeley.edu/~dfb/papers.html`

[Est96] Estrin, Deborah et al: „Virtual InterNetwork Testbed (VINT): methods and system“, ISI Proposal 96-ISI-05, 1996
`http://www.isi.edu/nsnam/vint/vint.ps`

[Est99] Estrin, Deborah et al.: „Network Visualization with the VINT Network Animator Nam“, Technical Report 99-703b, University of Southern California, März 1999
`http://www.isi.edu/~johnh/PAPERS/Estrin99d.ps.gz`

[Fal99] Fall, Kevin: „Network Emulation in the Vint/NS Simulator“, Technical Report, University of California, Berkeley, Januar 1999.
Auch erschienen in: Proceedings of the Fourth IEEE Symposium on Computers and Communications (ISCC'99), Juli 1999
`http://www.cs.berkeley.edu/~kfall/papers/iscc99.ps`

[Gam96] Gamma, Erich et al.: „Design Patterns: elements of reusable object-oriented Software“, Addison-Wesley Publishing Company, 1995

[Kes88] Keshav, Srinivasan: „REAL: A Network Simulator“, Computer Science Department Technical Report 88/472, University of California, Berkeley, Dezember 1988
`http://www.cs.cornell.edu/skeshav/papers/real.ps`

[Scotty] Scotty – Tcl Extensions for Network Management,
`http://www.ibr.cs.tu-bs.de/projects/scotty/`