Ulrich Flegel, Michael Meier (Eds.)

Detection of Intrusions and Malware & Vulnerability Assessment

GI Special Interest Group SIDAR Workshop, DIMVA 2004
Dortmund, Germany, July 6-7, 2004
Proceedings

Gesellschaft für Informatik 2004

Lecture Notes in Informatics (LNI) - Proceedings
Series of the Gesellschaft für Informatik (GI)

Volume P-46

ISBN 3-88579-375-X
ISSN 1617-5468

Volume Editors
Ulrich Flegel
University of Dortmund,
Computer Science Department, Chair VI, ISSI
D-44221 Dortmund, Germany
ulrich.flegel@udo.edu
Michael Meier
Brandenburg University of Technology Cottbus,
Computer Science Department, Chair Computer Networks
P.O. Box 10 13 44, D-03013 Cottbus, Germany
mm@informatik.tu-cottbus.de

printed by Köllen Druck+Verlag GmbH, Bonn

Komponenten für kooperative Intrusion-Detection in dynamischen Koalitionsumgebungen

Marko Jahnke, Martin Lies
Sven Henkel, Michael Bussmann
Forschungsgesellschaft für Angewandte
Naturwissenschaften e.V. (FGAN)
Neuenahrer Str. 20, 53347 Wachtberg
{jahnke|lies|henkel|bus}@fgan.de

Jens Tölle
Universität Bonn
Institut für Informatik, Abt. IV
Römerstr. 164, 53117 D-Bonn
toelle@cs.uni-bonn.de

Abstract:
Koalitionsumgebungen sollen für alle miteinander kooperierenden Mitglieder einen Vorteil bei der Verfolgung eines gemeinsamen Ziels erbringen. Dies gilt für die verschiedensten Anwendungsbereiche, etwa bei kooperierenden Strafverfolgungsbehörden, Wirtschaftsunternehmen oder Streitkräfte. Auch bei der Erkennung von sicherheitsrelevanten Vorgängen in vernetzten Computersystemen erhofft man sich von der Zusammenarbeit eine verbesserte Erkennungsfähigkeit sowie eine schnelle und koordinierte Reaktion auf Einbruchsversuche.

Dieser Beitrag stellt verschiedene praxisorientierte Werkzeuge für die koalitionsweite Vernetzung von Ereignismeldungs-produzierenden Sicherheitswerkzeugen vor, die wesentliche Probleme des Anwendungsszenarios lösen helfen:
Frühzeitige Anomaliewarnung – ein graphbasierter Anomaliedetektor wird als adaptives Frühwarnmodul für großflächige und koordinierte Angriffe, z.B. Internet-Würmer, eingesetzt.
Informationsfilterung – Meldungen werden beim Verlassen der lokalen Domäne entsprechend der domänenspezifischen Richtlinien zur Informationsweitergabe modifiziert (d.h. insbesondere anonymisiert bzw. pseudonymisiert).
Datenreduktion – zusätzliche Filter zur Datenreduzierung auf der Basis von vordefinierten Abhängigkeitsregeln steigern die Handhabbarkeit des Datenflusses.
Die Funktionsfähigkeit der genannten Komponenten wird derzeit in Form einer prototypischen Implementierung eines *Meta-IDS* für dynamische Koalitionsumgebungen nachgewiesen.

1 Einführung

Koalitionsumgebungen (*Coalition Environments, CEs*) finden sich in den verschiedensten Bereichen der Gesellschaft, so z. B. bei internationalen Kooperationen von Strafverfolgungsbehörden, bei Allianzen von Wirtschaftsunternehmen oder zusammenarbeitenden Streitkräften (z.B. NATO). Kennzeichnend sind oft gemeinsame Interessen oder ein gemeinsam durchzuführender Auftrag der Koalitionsteilnehmer. Gleichzeitig existiert gegenüber den Partnern in einer Koalition oft nur ein beschränktes Vertrauen, etwa wenn ein Konkurrenzverhältnis besteht.

Aus offensichtlichen Gründen können großflächige sicherheitsrelevante Aktivitäten in Computernetzen schneller aufgedeckt werden, wenn Zugriff auf eine größere Menge von angriffsabhängigen Ereignismeldungen möglich ist. Andererseits muss die Informationsweitergabe über kritische Aktivitäten und Tendenzen an andere Zusändigkeitsbereiche (Domänen) einer Koalitionsumgebung schnell und in einem interoperablen Format erfolgen, damit ein entsprechender Vorteil aus der Kooperation entstehen kann. Das Ziel besteht also darin, alle verfügbaren Angriffsinformationen zusammenzuführen, sie effizient auszuwerten und die Analyseergebnisse weiterzuleiten.

Die technische Realisierung einer derartigen Kooperation stellt sich in Form der Vernetzung verschiedener Sicherheitswerkzeuge (z.B. IDS, Paketfilter oder Integritätsprüfprogramme) dar, die Informationen über an anderer Stelle im Koalitionsnetzwerk registrierte relevante Aktivitäten erhalten, auswerten und weitergeben. Die besondere Herausforderung in dynamischen Koalitionsnetzwerken ensteht zum Einen durch Fluktuation bei den Koalitionsmitgliedern und durch sich ändernde Vertrauensverhältnisse. Diese schlagen sich in der Bereinigung von Informationen, die eine Domäne verlassen, nieder (z. B. durch Anonymisierung bzw. Pseudonymisierung von Ereignismeldunge). Zum Anderen stellt die Heterogenität der zugrundeliegenden Sicherheitswerkzeuge sowie deren durchzusetzenden Sicherheitsrichtlinien ein Hindernis für die effiziente Erkennung von Angriffen dar.

Dieser Beitrag stellt Komponenten für die Verarbeitung und Auswertung koalitionsweiter Ereignismeldungsvolumina vor, die zur Lösung dieser Fragestellungen beitragen. Der Nachweis ihrer Funktionsfähigkeit wird durch die prototypische Realisierung eines *Meta-IDS* geführt. Der Rest dieses Beitrages ist wie folgt strukturiert: In den Abschnitten 2, 3 und 4 werden die neuen Komponenten beschrieben. In Abschnitt 5 werden das Konzept und die bisher erzielten Ergebnisse vorgestellt, eingeschlossen der Implementierung. Zum Abschluss werden in Abschnitt 6 unsere Ansätze zusammengefasst und mögliche Erweiterungen skizziert.

2 Anomalieerkennung im Ereignismeldungs-Datenmodell

Die Zusammenführung von Ereignismeldungen aus unterschiedlichen Quellen dient als Ausgangsbasis für einen Ansatz zur Beurteilung der “Normalität” des aktuellen Meldungsaufkommens. Somit wird an dieser Stelle ein Ansatz der Anomalieerkennung eingesetzt. Die Details dazu werden im folgenden erläutert.

Eine der wesentlichen Herausforderungen für ein in kooperierenden IDS eingesetztes Anomalieerkennungsverfahren ist sein Umfeld. Über entsprechende Transportwege trifft eine große Zahl von Ereignismeldungen aus den angeschlossenen Domänen ein. Es ist jedoch auf Grund der Heterogenität der Datenbasis nur sehr begrenzt möglich, Annahmen über die Art und die Häufigkeit der Meldungen zu machen. Aus bereits angesprochenen Gründen kann sowohl die Frequenz der eintreffenden Ereignismeldungen, deren Inhalte und deren Aufbau stark von der konkreten Installation und Konfiguration abhängen. Das hier eingesetzte Verfahren ist ursprünglich für die Überwachung und Anomalieerkennung des

Verkehrs in Netzwerken entwickelt worden und funktioniert folgendermaßen:

Es hat sich gezeigt, dass die typischen Strukturen über die Zeit recht stabil sind, also nur selten grundlegende Veränderungen auftreten (vgl. [TdW02]). Aus diesem Grund kann man regelmäßig in kurzen Abständen den aktuellen Verkehr erfassen (möglich durch Netzwerk-Monitoring-Geräte oder auch durch Packetsniffer, in geswitchten Netzwerken an einem Spiegelport eines Switches) und in Form einer Verkehrsmatrix abspeichern. Diese Verkehrsmatrix kann als Graph $G = (V, K)$ interpretiert werden. Knoten $v_i \in V$ des Graphen G sind damit an der Kommunikation im aktuellen Zeitabschnitt beteiligte Geräte, während Kanten $k_{i,j} \in K$ der Kommunikation zwischen zwei Geräten v_i und v_j entsprechen. Die Kanten sind dabei mit der "Intensität" der Kommunikation gewichtet.

Diese Graphen können mittels *Graph-Clustering*-Algorithmen in Teilgraphen zerlegt werden. Das Clustering dieses Graphen repräsentiert somit die typische Kommunikationsstruktur innerhalb des überwachten Netzes. Clustering bedeutet das Finden einer Zuordnung jedes Knotens zu einem von mehreren Clustern. Diese exklusive Klassifizierung nennt man auch Partitionierung der Objektmenge. Jedem Objekt der Objektmenge wird genau ein Cluster zugeordnet. Formal ist ein Clustering $\Re$ die Aufteilung eines Graphen $G = (V, K)$ in Cluster

- $C_i \subseteq V, C_i \neq \emptyset, 0 \leq i \leq n-1$ mit
- $C_0 \cup C_1 \cup \cdots \cup C_{n-1} = V$ und
- $\forall 0 \leq i, j \leq n-1, i \neq j : C_i \cap C_j = \emptyset$.

Plötzliche Änderungen dieser Strukturen werden als Anomalie aufgefasst und gemeldet. Dazu ist eine Metrik erforderlich, die Ähnlichkeiten von zeitlich aufeinanderfolgenden Clusterings $\Re_t$ und $\Re_{t+1}$ bewertet.

Das im Bereich der Verkehrstrukturen genutzte Verfahren benötigt nun einige Anpassungen an das hier betrachtete Ereignismeldungsmodell. Viele der an der Anomalieerkennungskomponente eintreffenden Meldungen können analog den oben betrachteten Verkehrsgraphen unmittelbar für den Aufbau eines Graphen verwendet werden. In dieser Kategorie fallen alle Ereignismeldungen, die detailliert Ziel (das betroffene System) sowie Quelle (vermuteter Auslöser) eines Ereignisses angeben. Damit ist eine weitere Kante $k \in K$ des aufzuteilenden Verkehrsgraphen G gefunden: Die Kante k führt von der Quelle $v_{Quelle} \in V$ zum Knoten $v_{Ziel} \in V$.

In Abhängigkeit des Detaillierungsgrades kann es vorkommen, dass durch die Anwendung der lokalen Richtlinie zur Informationsweitergabe (vgl. Abschnitt 3) in manchen Ereignismeldungen nur Informationen über das betroffene Zielsystem vorhanden sind, jedoch keine Angabe über einen (vermuteten) Auslöser gemacht werden. Um solche Ereignismeldungen trotzdem in geeigneter Form im Ereignismeldungsgraph zu berücksichtigen, kann den Domänen für bestimmte Ereignismeldungstypen ein Pseudoknoten zugeordnet werden, der über Ereignismeldungen repräsentierende Kanten mit betroffenen Systemen repräsentierende Knoten verbunden wird.

Der mittels Graph-Clustering-Verfahren aufgeteilte Graph beschreibt somit die typische Struktur der eintreffenden Ereignismeldungen. Abweichungen der typischen Meldungs-

struktur sind auch hier wieder als Anomalie anzusehen und zu melden. Dabei können die im Kontext der Anomalieerkennung in Netzwerkverkehrsstrukturen genutzten Vergleichsmaße eingesetzt werden.

Auch bei dieser Anwendung von Anomalieerkennungsverfahren besteht wieder die bekannte Herausforderung, dass solche Verfahren grundsätzlich anfällig sind für die Erzeugung von Fehlalarmen (*False Positives*) oder reale Bedrohungen nicht erkennen (*False Negatives*). Auch die hier angewandten Methoden können diese Problematik nicht grundsätzlich umgehen, sondern es ist durch sorgfältige Abstimmung aller Parameter im konkreten Einsatzszenario darauf zu achten, dass die Anzahl der False Positves und False Negatives erträglich bleibt. Grundsätzlich ist anzumerken, dass Verfahren dieser Kategorie nicht ausschließlich verwendet werden sollten und immer nur zusätzliche Informationen zum aktuellen Gesundheitszustand des überwachten Netzes geben können.

3 Informationsbereinigung bei Ereignismeldungen

Bevor Ereignismeldungen eine Domäne verlassen, müssen Sie entsprechend der lokalen Richtlinie zur Informationsweitergabe (Information Sharing Policy, *IShP*) bereinigt (d.h. anonymisiert bzw. pseudonymisiert, vgl. [Fle02a], [Fle02b]) werden. Dies kann beispielsweise durch entsprechende Gateways an den Domänengrenzen bewerkstelligt werden. Dieser Prozess kann als ein Problem der bedingten Musterübereinstimmungsprüfung und Mustertransformation (*Conditional Pattern Matching and Transformation, CPMT*) für Ereignismeldungen aufgefasst werden. Es erweist sich als sinnvoll, die Problematik wie folgt zu formalisieren:

Sei Σ die Menge der Ereignismeldungen oder *Ereignisse*. Ein *Ereignis-Matching-Template* E ist ein Ausdruck, der sich zu einer Menge von Ereignissen expandieren lässt, d.h. $Expand(E) \in \mathcal{P}(\Sigma)$ mit der Potenzmenge $\mathcal{P}(\Sigma)$ der Ereignismeldungen. Ein Ereignis e stimmt genau dann mit einem Template E überein (notiert als $e \dashv E$), wenn $e \in Expand(E)$ gilt. Die *Matching-Funktion* ergibt sich intuitiver Weise als

$$E : \Sigma \rightarrow I\!B, \quad E(e) = \begin{cases} true, & \text{wenn } e \in Expand(E) \\ false, & \text{wenn } e \notin Expand(E) \end{cases}$$

Sei $P = \{(p_1, \ldots p_m)\}$ eine beliebige Menge zeitvarianter Parameterbelegungen. Eine *bedingte Transformationsvorschrift* ist ein Tripel

$$R = (E^M, P, E^T)$$

mit einer Menge von *Matching-Templates*

$$E^M = \{E_i^M \mid E_i^M : \Sigma \rightarrow I\!B,\ i = 0, \ldots, n\}$$

und einem *bedingten Transformations-Template*

$$E^T : \Sigma \times P \ \rightarrow \ \mathcal{P}(\Sigma)$$

Sei T das betrachtete Zeitintervall, $e(t) \in \Sigma,\ \forall t \in T$ eine Sequenz von Ereignismeldungen und $p(t) \in P$ eine Sequenz von Parameterbelegungen. Die *bedingte Transformationsfunktion* der Vorschrift R ist

$$f_R : \Sigma \times T \rightarrow \mathcal{P}(\Sigma) \text{ mit}$$

$$f_R(e(t), t) = \begin{cases} E^T(e(t), p(t)), & \text{wenn } e(t) \dashv E_i^M,\ \forall i = 0, \ldots, n \\ \{e(t)\}, & \text{sonst} \end{cases}$$

d. h. abhängig von der Übereinstimmung der Templates E_i^M mit dem Eingabe-Ereignis $e(t)$ werden Null oder mehr Ausgabe-Ereignisse erzeugt. Man beachte, dass diese Transformation zustandslos ist, d.h. keine Ereignisse der Eingabe im Filter gespeichert werden.

Betrachtet man den Anwendungskontext der Informationsbereinigungs-Gateways an den Domänengrenzen, so muss zunächst aus der lokalen IShP eine Menge von bedingten Transformationsregeln der Form $R = (\{E_0^M\}, P, E^T)$ generiert werden, wobei wir im Folgenden o. B. d. A. von $P = \emptyset$ ausgehen. Unglücklicherweise kann ein sinnvoller Informationsbereinigungsprozess nicht ausschliesslich auf einer statischen Textsubstitution basieren. Wenn beispielsweise IP-Adressen als Bestandteil von Meldungen durch feste Werte ersetzt werden sollen, gehen alle Informationen über die Topologie des Netzwerks verloren. Offensichtlich behindert dies den Erkennungsprozess, insbesondere, wenn verkehrsbezogene Anomalien entdeckt werden sollen. Daher benötigt man eine flexiblere Methode, um Transformationsregeln zu spezifizieren: *Submatching-Referenzen*.

Sei $s(E_0^M)$ eine Menge von qualifizierten Teilausdrücken im Matching-Template E_0^M, und sei $s(E_0^M, e(t))$ die Menge von in $e(t)$, die den Teilausdrücken in $s(E_0^M)$ entsprechen. Wenn man nun das Transformations-Template zu

$$E^T : \Sigma \times s(E_0^M) \times P \ \rightarrow \ \mathcal{P}(\Sigma)$$

erweitert, erhält man die Möglichkeit, neue Meldungen zu konstruieren, die auf den übereinstimmenden Teilausdrücken des alten Ereignisses basieren. Dies ist offensichtlich eine Grundvoraussetzung für die Anwendung als Verfahren zur Informationsbereinigung.

Man beachte, dass es – nur durch Festlegung entsprechender Regeln – mit diesem Verfahren ebenfalls möglich ist, eine *Normalisierung* von Ereignismeldungen zu realisieren (z. B. wenn dasselbe Ereignis mit unterschiedlichen Signaturbezeichnungen oder Referenzen auf Angriffsdatenbanken gemeldet wird). Um die Möglichkeiten der Transformationsregeln noch erweitern zu können, ist es offensichtlich sinnvoll, Systemparameter – wie den Zeitpunkt des letzten Matchings, die aktuelle Systemzeit oder die IP-Adresse des GWs – als Parametermenge P einzubeziehen.

4 Module für die Verarbeitung von Meldungskombinationen

In unserem Ansatz gibt es zwei Verarbeitungsmodule, die den Umgang mit hohen Meldungsraten durchführbar machen sollen: *Redundanzfilter* und das Anwenden von vordefinierten Erkennungsregeln für miteinander korrelierte Elementarereignisse (*Kombinati-*

onsdetektor). Beide Aufgabenstellungen sind Spezialfälle der sog. *Event Corellation* und bereits mehrfach untersucht worden.

Im Laufe unserer Forschungsarbeiten hat sich jedoch herausgestellt, dass man zur Realisierung ein Verfahren einsetzen kann, das die in Abschnitt 3 beschriebenen Methode zur Informationsbereinigung erweitert und ein sehr universelles und mächtiges Werkzeug zur Verfügung stellt.

4.1 Filterung von Redundanzen

Um die Anzahl von Ereignismeldungen reduzieren zu können, ist es offensichtlich notwendig, Redundanzen zu vermeiden. Deshalb ist es oft sinnvoll, mehrere Meldungen mit einem "ähnlichen" Inhalt zu einer einzelnen, neuen Meldung zusammenzufassen. In [DW01] beispielsweise wurde diese einfache Form der Zusammengehörigkeit durch die Definition von sog. Duplikaten modelliert.

Es ist möglich, das im Abschnitt 3 beschriebene CPMT-Verfahren derart zu erweitern, dass es zur diesbezüglichen Filterung des Meldungsflusses herangezogen werden kann. Die Matching-Templates E^M müssen durch entsprechende Erweiterungen auch die Ähnlichkeitseigenschaft von Meldungen spezifizieren können. Da man nicht nur absolute (initiale) Matchings benötigt, sondern auch *relative Matchings* (d.h. abhängig von vorhergehenden, unter Festlegung der Differenz), muss man den Matchingvorgang erweitern. Man führt ein zweites Matching-Template E_1^M ein, das auf Submatchings des initialen Matchings E_0^M verweist. Für die Menge der Matching-Templates $E^M = \{E_0^M, E_1^M\}$, erhält man

$$E_0^M : \Sigma \rightarrow I\!B, \qquad E_1^M : \Sigma \times s(E_0^M) \rightarrow I\!B$$

Beispiel: Es sollen alle Meldungen mit identischer Ereignisklassifikation und Quell-/Ziel–IP-Adresse gefiltert werden. Gleichzeitig soll der Entstehungszeitpunkt der Meldungen innerhalb einer Sekunde liegen. So muss

- E_0^M einen geeigneten Teilausdruck für den Entstehungszeitpunkt der Meldung und für die Quell-/Ziel–Adresse besitzen,
- E_1^M einen bedingten Ausdruck für die Quell-/Ziel–Adresse haben, der identisch zu den entsprechenden Werten des initialen Matchings $e(t) \dashv E_0^M$ ist und
- E_1^M einen bedingten Ausdruck für den Wert des Entstehungszeitpunktes beinhalten, der nicht um mehr als eine Sekunde vom Entstehungszeitpunkt des initialen Matchings entfernt ist.

Im Gegensatz zu (zustandslosen) bedingten Transformationen, bei denen ein absolutes Matching für alle Templates E_i^M erforderlich ist, benötigt man zwei Speicherplätze (*Buckets*)

$$B_i = \{e | e \vdash E_i^M\},\ i = 0, 1$$

mit $0 \leq |B_0(t)| \leq 1, |B_1(t)| \geq 0\ \forall t \in T$ für bereits mit den Templates übereinstimmende Ereignisse. Zusätzlich benötigt man den Speicherzeitpunkt t_{B_0} des ersten Ereignisses,

welches im Bucket B_0 abgelegt wurde, um das Ende der Zusammenfassungphase nach der maximalen Speicherzeit Δt_{B_0} feststellen zu können. Die Transformationsfunktion hängt nicht vom aktuellen Ereignis $e(t)$, sondern von den vorher gespeicherten Ereignissen ab, d.h. wir definieren ein Transformationstemplate

$$E^T : \mathcal{P}(\Sigma) \times \mathcal{P}(\Sigma) \times s(E_0^M) \times P \rightarrow \mathcal{P}(\Sigma)$$

sowie für die Transformationsfunktion f_R einen Algorithmus für den Redundanzfilterprozess, der nicht auf externe Parameter zurückgreift und sich daher wie folgt gestaltet:

Eingabe: $R = (\{E_0^M, E_1^M\}, \emptyset, E^T)$, Δt_{B_0} und eine Sequenz von Ereignissen $e(t)$.
Ausgabe: Eine Sequenz von Ereignismengen $\{e\}$, in der ähnliche Ereignisse zusammengefasst sind.
Algorithmus:

```
(1)   B_0(0) := ∅ ; B_1(0) := ∅ ; t_{B_0} := 0
(2)   t := 1
(3)   while true do
(4)       read e(t)
(5)       if t_{B_0} ≠ 0 and t − t_{B_0} ≥ Δt_{B_0} then
(6)           output E^T(B_0(t), B_1(t), s(E_0^M))
(7)           B_0(t) := ∅ ; B_1(t) := ∅
(8)           t_{B_0} := 0
(9)       fi
(10)      if e(t) ⊀ E_0^M and e(t) ⊀ E_1^M then
(11)          output {e(t)}
(12)      fi
(13)      else if |B_0| = 0 and e(t) ⊣ E_0^M then
(14)          B_0(t) := {e(t)}
(15)          t_{B_0} := t
(16)      fi
(17)      else if |B_0| > 0 and e(t) ⊣ E_1^M then
(18)          B_1(t) := B_1(t − 1) ∪ {e(t)}
(19)      fi
(20)      t := t + 1
(21)  done
```

Man beachte, dass dieser Algorithmus nicht voraussetzt, dass “ähnliche” Ereignisse sequentiell und ohne andere dazwischenliegende Meldungen eintreffen. Zusätzlich ist es möglich, mehrere Buckets $B_{ij}(t)$ parallel zu verwenden. Die Anzahl dieser Buckets muss aber begrenzt sein, um Überlastsituationen und DoS-Angriffe zu vermeiden. Weiterhin ist es möglich, anstatt $e' = E^T(B_0, B_1, s(E_0^M))$ erst nach Beendigung der Zusammenfassungsphase zu berechnen (Algorithmus Zeile `(6)`), nach jedem Teil-Matching das spätere Ausgabeereignis e' entsprechend anzupassen. Da in diesem Falle nur ein Bucket für ein Ereignis benötigt wird, reduziert sich der Speicheraufwand erheblich.

4.2 Kombinationsdetektoren

Im Gegensatz zur Redundanzeigenschaft definieren wir *Ereigniskombinationen* als Mengen von miteinander korrelierten Ereignismeldungen, die – separat betrachtet – keine besondere Aufmerksamkeit erfordern, in ihrer Gesamtheit jedoch eine möglicherweise wesentliche höhere Kritikalität des Systemzustandes implizieren. Die Anwendung besteht also nicht primär in der Substitution, sondern in der zusätzlichen Meldung derartiger kritischer Kombinationen. In der Literatur finden sich verschiedene Beispiele für diesen komplexeren Fall der "Zusammengehörigkeit". Das in [Jul02] beschriebene Clustering von Meldungen beschreibt eine Zuordnung mehrerer Meldungen zu einer einzigen, die als Ursache der gemeldeten Ereignisse betrachtet werden kann (*Root Cause*). In [CM02] werden Ereigniskombinationen zusätzlich mit externen Bedingungen bezüglich der Konfiguration der betrachteten Netze verknüpft. Auch die dezentralisierte Verarbeitung der Korrelationsregeln durch das Versenden von Nachrichten an verteilte Verarbeitungsknoten wurde bereits untersucht [KTK02].

Auch die Kombinationserkennung läßt sich in Form eines CPMT-Problems formulieren. Offensichtlich handelt es sich um einen verallgemeinerten Fall der Redundanzfilter, die im vorhergehenden Abschnitt beschrieben wurden. Statt genau eines absoluten und eines relativen Matchings spezifiziert man eine Menge von relativen Matching-Templates $E^M = \{E_1^M, \ldots, E_n^M\}$, um eine Ereigniskombination beschreiben zu können. Jedes Matching kann sich dabei nicht nur auf Submatchings im initialen Template E_0^M beziehen, sondern auch auf Submatchings in den übrigen Templates

$$E_i^M : \Sigma \times \prod_{j=0, j\neq i}^{n} s(E_j^M) \rightarrow I\!B, \; \forall i = 1, \ldots, n$$

Unter Zuhilfenahme mehrerer Buckets $B_i, 0 \leq i < n$ ist es möglich, auf alle in den Buckets gespeicherten Ereignisse zu verweisen, indem E^T erweitert wird zu

$$E^T : \prod_{i=0}^{n} \mathcal{P}(\Sigma) \times \prod_{j=0}^{n} s(E_j^M) \times P \; \rightarrow \; \mathcal{P}(\Sigma)$$

Mit diesem Ansatz können wir verschiedene Korrelationsbeziehungen modellieren, die für die Erkennung sicherheitsrelevanter Vorkommnisse von Bedeutung sind. So lassen sich beispielsweise folgende Korrelationseigenschaften zwischen zwei Matching-Templates E_0^M und E_1^M ausdrücken (vgl. [Cup01]):

- *Zusammenhänge der Meldungs-Klassifikation*:
  Definiere Ausdrücke in E_0^M und E_1^M, die eine Teilmenge von möglichen Ereignisklassifikationen (z.B. Aufzählungen oder Bereiche von IDs aus einer Angriffsdatenbank) beschreiben.
- *Zeitliche Zusammenhänge*:
  Definiere einen Teilausdruck in E_0^M, der den Zeitwert beinhaltet, und einen arithmetischen Ausdruck in E_1^M, der den Zusammenhang spezifiziert (z.B. eine maximale Zeitdifferenz der Detektion).

- *Räumliche Zusammenhänge*:
  Definiere einen Teilausdruck in E_0^M, der den Adresswert beinhaltet, und einen arithmetischen Ausdruck in E_1^M, der den Zusammenhang spezifiziert (z.B. Zugehörigkeit zum selben Subnetz).

Allerdings ist der bei einer Implementierung der zu erwartende Speicheraufwand im Falle einer *Online*-Verarbeitung (d. h. ohne Datenbankzugriff, einzig auf die in der Verarbeitungspipeline befindlichen Meldungen zurückgreifend) gewaltig. Der Grund besteht darin, dass alle betreffenden Ereignisse im Speicher vorgehalten werden müssen, da während der Aggregation noch nicht gesichert ist, dass alle Bedingungen für die Kombination erfüllt werden.

5 Bisherige Ergebnisse

Die hier beschriebenen Kompontenten wurden zum überwiegenden Teil bereits prototypisch implementiert und ihre Funktionsfähigkeit verifiziert.

5.1 Architektur

Zum Nachweis der Funktionsfähigkeit der verschiedenen Komponenten wurde eine *Meta-IDS-Architektur* implementiert, die die Anforderungen dynamischer Koalitionsumgebungen eher erfüllt als der Ansatz kooperierender Einzelsysteme (vgl. [LJT+04], [JTBH04]). Die Architektur besteht im wesentlichen aus zentralen Einheiten für die Datenanalyse und Steuerung des Gesamtsystems (*Management-Konsolen*) und Aufschaltpunkten an den Domänengrenzen (*IDS-Gateways, GWs*), die einerseits die anfallenden Ereignismeldungen an die Konsolen vorverarbeiten und weiterleiten sowie andererseits die etwaigen Warnmeldungen über potentiell sicherheitskritische Vorgänge an die Domänen übermitteln. Diese Komponenten und ihre Funktionseinheiten sind in Abb. 1 dargestellt. Die Implementation erfolgte auf Basis generischer IDS-Infrastrukturkomponenten, die bei früheren Arbeiten entwickelt worden sind [Jah02].

Das gemeinsame Datenmodell der Ereignismeldungen entspricht dem Intrusion Detection Message Exchange Format (*IDMEF*, [CD03]). Meldungen werden mittels des Intrusion Detection Exchange Protocol (*IDXP*, [FMW02]) transparent übertragen. Um einen gewissen Schutz gegen Denial-of-Service-Angriffe zu gewährleisten, werden alle in [Jah03] geschilderten Maßnahmen angewendet, so z.B. die Überwachung der Gateway-Prozessumgebungen sowie die redundante Auslegung der Management-Konsolen für den Fall einer Netzverbindungs- oder Prozessterminierung.

Während die zentralen Komponenten von einer geeigneten, für alle Beiteiligten akzeptierbaren Autorität betrieben werden, stehen die Gateways unter der Kontrolle des lokalen Sicherheitspersonals einer Domäne. Es hat u. a. die jeweiligen Informationsweitergaberichtlinien in Form einer geeigneten Konfiguration des Gateways durchzusetzen und

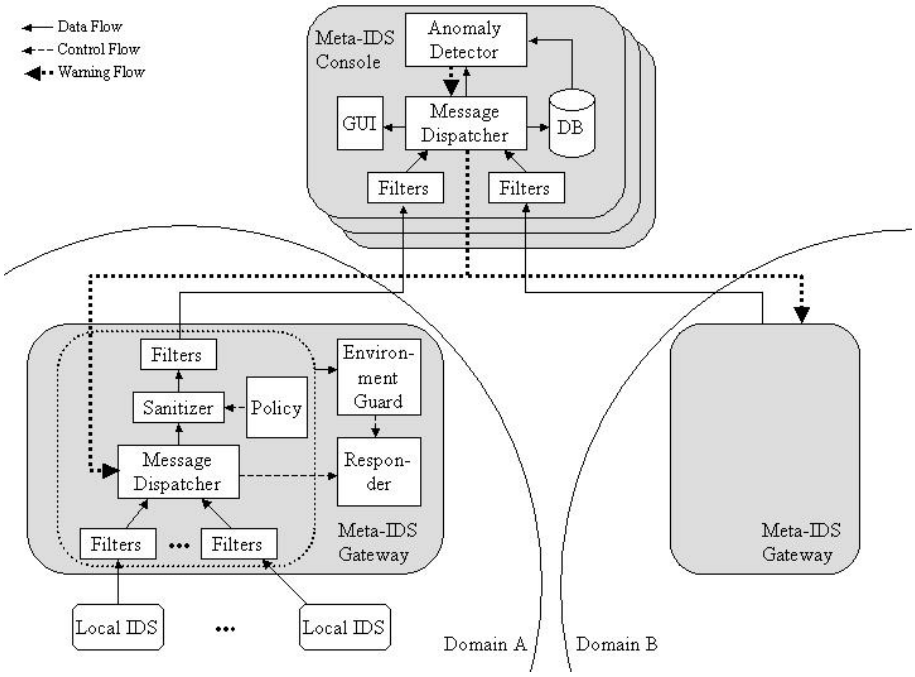

Abbildung 1: Architekturkomponenten des prototypischen Meta-IDS.

gleichzeitig darauf zu achten, dass Randbedingungen für die Art und den Grad der Anonymisierung der Meldungen eingehalten werden. Diese Randbedingungen ergeben sich aus der Art der Detektionsstrategie und müssen in weitergegenden Arbeiten noch ermittelt werden.

Damit eine Domäne nicht etwa durch geschicktes Abfragen des anonymisierten Meldungsbestandes der zentralen Komponenten verborgene Information wiederherstellen kann, ist dieser Meldungsbestand entsprechend unter Verschluss zu halten. Sind in Warnmeldungen, die an alle Domänen weitergegeben werden sollen, domänenspezifische Angaben enthalten, so sollte die Weitergabe (u. U. manuell) autorisiert werden. Anderenfalls könnte eine Domain durch Provozieren einer Anomalie dem System gezielt Informationen über eine andere Domäne entlockt werden.

5.2 Anomalieerkennung

Zur Zeit wird das Anomalieerkennungssystem in die bestehende IDS-Infrastruktur integriert. In einer ersten Erprobungsstufe findet eine Beschränkung auf den ursprünglich für die Verkehrsanomalie-Erkennung entwickelten Ansatz statt, d.h. es werden zur Erkennung

typischer Ereignismeldungsstrukturen ausschließlich Source-Target-Beziehungen für den Aufbau der Graphen verwendet. Die Portierung der bisher verwendeten Algorithmen geschieht aber schon mit Rücksicht auf die eingeplanten Erweiterungen des Verfahrens.

Zur Bewertung des Systems wird auf operationelle Datenquellen aus einer bestehenden DMZ zurückgegriffen, die gezielt und reproduzierbar mit künstlich generierten Ereignismeldungen gemischt werden können. Diese künstlichen Meldungen werden beispielsweise aus Paketfilter-Logdaten gewonnen, die zuvor von einem "Wurm-Simulator" (vgl. [Sch02]), dem Ausbreitungsverhalten realer oder fiktiver internet-Würmer entsprechend, für die jeweilige Netzkonfiguration ermittelt wurden. Erste Ergebnisse haben gezeigt, dass nicht nur die Ausbreitung von Würmern, sondern auch diverse gezielte (reale) Angriffe gegen einzelne Knoten zuverlässig erkannt werden können.

5.3 Informationsbereinigung bei Ereignismeldungen

Die in den Abschnitten 3 und 4 beschriebenen Ansätze der Informationsbereinigung (einschließlich einer Normalisierung), der Redundanzfilterung und der Erkennung von Ereigniskombinationen können auf die bedingte Ereignismeldungs-Transformation reduziert werden. Ein häufig gewählter Ansatz, um komplexe Transformationen von XML-formatierten Daten durchzuführen, ist die Anwendung von XSLT-Stylesheets [W3C99]. Die bekanntermaßen nicht besonders hohe Performanz von XSL steht einer großen Flexibilität und universellen Einsetzbarkeit entgegen. XSL erlaubt es, ohne Modifikation des Programmcodes – selbst bei Änderungen der IDMEF-Spezifikation – bei der Regelbearbeitung auf beliebige Bestandteile der zu verarbeitenden Meldungen zuzugreifen, die über die üblichen Zeitangaben, Klassifikationen, Adressen und Portnummern hinausgehen. Die Verwendung marktverfügbarer Werkzeuge bei der Definition, Umsetzung und Wartung von Sicherheits- und Informationsweitergabe-Richtlinien ermöglicht zudem einen kostensparenden Einsatz.

Unglücklicherweise beinhalten XSLT-Stylesheets keine Regulären Ausdrücke (REs), und zum Implementierungszeitpunkt waren auch keine geeigneten Erweiterungen verfügbar. Deshalb haben wir für einen ersten Meta-IDS-Prototypen einen XML-Prozessor implementiert, der XSLT-Sheets um POSIX.1-REs sowie um zusätzliche boolesche und arithmetische Ausdrücke bereichert. Er wendet das beschriebene Matching- und Transformationsverfahren auf IDMEF-Meldungen an.

In dem folgenden Beispiel wird ein kombiniertes Matching- und Transformations-Template (sog. *Transformations-Sheet*) als Teil einer modifizierten IDMEF-Nachrichtensyntax spezifiziert:

```
(1)  <IDMEF-Message xmlns:xsl=...>
       <Alert>
       ...
(2)      <address>
(3)        192\.22\.([0-9]{1,3})$X$\.([0-9]{1,3})$Y$
(4)        <condition>
(5)          ($Y<255)
```

```
(6)         </condition>
(7)         <transform>
(8)           <xsl:copy>
(9)            191.72
(10)               .<xsl:value-of select="$X"/>
(11)               .<xsl:value-of select="$Y"/>
(12)          </xsl:copy>
(13)        </transform>
(14)      </address>
        ...
(15)    </Alert>
(16) </IDMEF-Message>
```

Die Matchings der letzten beiden Bytes einer IP-Adresse `192.22.x.y` werden als Variablen `$X` und `$Y` referenziert (Zeile `(3)`). Und bei der Transformation entsprechend eingesetzt (Zeilen `(9)`-`(11)`). Die zusätzliche Bedingung dafür, dass die Transformation durchgeführt wird, ist, dass `$Y` nicht dem traditionellen Broadcastsuffix 255 entsprechen darf (Zeilen `(4)`-`(6)`).

Mit diesen Transformationstechniken wurden die Informationsbereinigungs-Funktionen der Gateways auf Basis der GNOME libxml2 und libxslt in Form von C++-Filterklassen im Rahmen des SNAF/SIDI-Frameworks (vgl. [Jah02]) implementiert. Um die Effizienz der Implementierung zu messen, wurde zunächst eine Anzahl von Transformations-Sheets erstellt, die beispielsweise IP-Adressen und Adressbestandteile, Hostnamen, Zeitangaben sowie Produktangaben über Sicherheitswerkzeuge anonymisieren. Unter Verwendung eines PIII/1GHz-PCs mit 256MB RAM und Debian GNU/Linux bewältigte ein Gateway für die Durchführung von 10 sequentiellen Transformationsschritten zwischen 19 und 23 Meldungen pro Sekunde (ohne Codeoptimierung), was zumindest für die prototypische Anwendung ein ausreichendes Ergebnis darstellt.

5.4 Redundanzfilter

Auch die in Abschnitt 4 beschriebene Redundanzfilterung wurde auf Basis von XSLT-Stylesheets implementiert. Ein Beispiel für die zu diesem Zweck abermals erweiterte IDMEF-Syntax zur Spezifikation ist folgendes:

```
        ...
(1)       <Source>
(2)        <Node>
(3)          <Address>
(4)            <address>
(5)              ([0-9]{1,3}\.[0-9]{1,3}\.5\.1)$SA$
(6)            </address>
(7)          </Address>
(8)        </Node>
(9)         <Service>
(10)         <protocol>(.*)$SP$</protocol>
(11)       </Service>
(12)     </Source>
```

```
(13)   <relMatch deltaT="20000" maxMatch="40">
(14)     <xsl:copy>
(15)       <Source>
(16)         <Service>
(17)          <protocol><xsl:value-of select="$SP"/></protocol>
(18)         </Service>
(19)       </Source>
(20)       <Target>
(21)         <Node>
(22)           <Address>
(23)             <address>(.*)$TA$</address>
(24)          </Address>
(25)         </Node>
(26)         <Service>
(27)            <port>(.*)$TP$</port>
(28)         </Service>
(29)       </Target>

(30)       <summary do="create">
(31)         <AdditionalData>
(32)           matched: sourceaddress=$SA sourceprotocol=$SP
(33)         </AdditionalData>
(34)       </summary>

(35)       <summary do="update">
(36)         <AdditionalData>
(37)           target: address=$TA port=$TP
(38)         </AdditionalData>
(39)       </summary>
      ...
```

In diesem Beispiel wird E_0^M durch die Zeilen `(1)-(12)` und E_1^M durch die Zeilen `(13)-(29)` spezifiziert. Die resultierende Nachricht wird erzeugt, wenn das absolute Matching E_0^M zutrifft und wird durch E^T (Zeilen `(30)-(34)`) beschrieben. Bei jedem darauf folgenden Eintreffen einer Nachricht, die mit E_1^M übereinstimmt, wird dieses Resultat modifiziert, wie in den Zeilen `(35)-(39)` beschrieben.

Auch die Implementierung der Redundanzfilterung wurde bezüglich ihrer Leistungsfähigkeit auf obengenannter Hardware getestet; sie wurde in Kombination mit dem Informationsbereinigungsmodul betrachtet. Erwarteter Weise stellten sich bei den in der Abbildung 2 skizzierten Messungen deutlich höhere Durchsatzraten für den Fall heraus, in dem die Informationsbereinigung erst nach der Redundanzfilterung durchgeführt wurde, anstatt sie vorzuschalten (gestrichelte im Ggs. zur durchgezogenen Linie).

Während dieser Tests bestätigte sich ausserdem, dass das Zusammenfassen von Eigenschaften mehrerer Templates – sofern anwendbar – zu einer signifikanten Verbesserung der Leistungsfähigkeit führt (in der Abbildung Einzelpunkte im Ggs. zu Linien). Dies ist durch die Tatsache begründet, dass Meldungsbestandteile nicht mehrfach auf Übereinstimmungen geprüft werden müssen. Allerdings gestaltet sich eine derartige Zusammenfassung nichttrivial und wird schnell unübersichtlich.

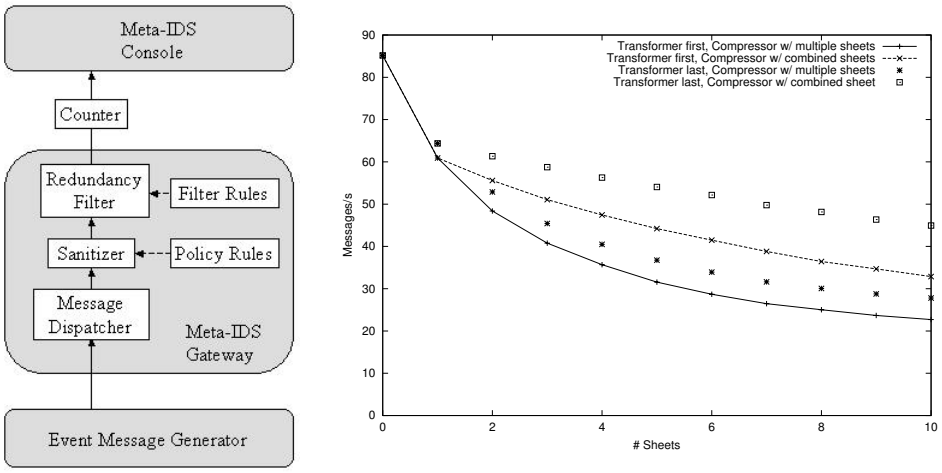

Abbildung 2: Szenario und Ergebnisse der Durchsatzmessungen für die kombinierte Redundanzfilterung mit Informationsbereinigung.

6 Zusammenfassung und Ausblick

In diesem Beitrag wurden verschiedene Verfahren vorgestellt, die sich zur Realisierung von Verarbeitungskomponenten kooperativer Intrusion-Detection-Syteme in dynamischen Koalitionsumgebungen eignen. Hier stellten sich insbesondere die Heterogenität der lokalen Sicherheitsrichtlinien, der zur Durchsetzung verwendeten Sicherheitswerkzeuge (IDS, Paketfilter, Integritätsprüfprogramme etc.), sowie der Richtlinien zur Informationsweitergabe als wesentliche Herausforderungen für die Realisierung eines koalitionsweiten Systems zur frühzeitigen Warnung vor großflächigen sicherheitsrelevanten Aktivitäten in Computernetzen heraus. Die vorgestellten Komponenten bestehen zum Einen aus einem Anomalieerkennungsmodul, das ein abnormales Aufkommen von Ereignismeldungen aus Abweichungen von einem zuvor erlernten Normalprofil erkennen kann, und zum Anderen aus Modulen zur Datenreduktion und zur Erkennung zuvor spezifizierter Kombinationen von Ereignismeldungen.

Die Erkennung von Anomalien erfolgt mittels etablierter Techniken, die auf der Bildung von Clustern in Quelle-Ziel-Graphen beruhen. Diese Graphen werden anhand der eintreffenden Ereignismeldungen gebildet und eine Abweichung eines speziellen Distanzmaßes zwischen dem aktuellen und dem Normalprofil wird als Anomalie gewertet. Alle Aufgaben, die die Verarbeitung von Ereignismeldungen betreffen, wurden auf bedingtes Pattern-Matching und Transformation zurückgeführt, das in Form eines erweiterten XSLT-Stylesheet-Prozessors realisiert wurde. Die beschriebenen Komponenten wurden zum Teil in einem prototypischen Meta-IDS realisiert und auf ihre Leistungsfähigkeit hin untersucht.

Das Verfahren der Anomalieerkennung wird derzeit ausgewertet, verfeinert und in das Gesamtsystem integriert. Seine Grenzen bei wachsender Modifikation der Ereignismeldungen durch die Informationsbereinigung sowie bei stark dynamischem Verhalten der Koalitionsumgebung werden untersucht. Für die Zukunft sind verschiedene Erweiterungen des Systems – insbesondere hinsichtlich der Extraktion relevanter Informationen, die als ergänzende Angaben in Anomaliewarnungen dienen – angedacht. Aspekte der Skalierbarkeit der Meldungsverarbeitung durch Parallelisierung werden be-

trachtet. Zudem wird an einem kontinuierlich laufenden Testszenario mit realen Ereignismeldungen aus verschiedenen entmilitarisierten Zonen (DMZ) von Forschungsnetzwerken gearbeitet.

Literatur

[CD03] D. Curry and H. Debar. Intrusion Detection Message Exchange Format – Data Model and Extensible Markup Language (XML) Document Type Definition. IETF Internet Draft draft-ietf-idwg-idmef-xml-10.txt, January 2003. IETF IDWG.

[CM02] F. Cuppens and A. Miège. Alert Correlation in a Cooperative Intrusion Detection Framework. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 202–215, Oakland, CA, May 2002. IEEE Computer Society, Technical Committee on Security and Privacy, IEEE Computer Society Press.

[Cup01] F. Cuppens. Managing Alerts in a Multi-Intrusion Detection Environment. In *Proceedings of the 17th Annual Conference on Computer Security Applications (ACSAC'01)*, New Orleans, LA, December 2001. ACM.

[DW01] H. Debar and Andreas Wespi. Aggregation and Correlation of Intrusion-Detection Alerts. *Lecture Notes in Computer Science*, 2212:85ff., 2001.

[Fle02a] U. Flegel. Anonyme Audit-Daten im Überblick. *DuD – Datenschutz und Datensicherheit*, 1(26), 2002.

[Fle02b] U. Flegel. Pseudononyzing Unix Log Files. In *Proceedings of the Infrastructure Security Conference (InfraSec2002), Bristol, UK*, October 2002.

[FMW02] B. Feinstein, G. Matthews, and J. White. The Intrusion Detection Exchange Protocol. IETF Internet Draft draft-ietf-idwg-beep-idxp-07.txt, October 2002. IETF IDWG.

[Jah02] M. Jahnke. An Open and Secure Infrastructure for Distributed Intrusion Detection Sensors. In *Proceedings of the NATO Regional Conference on Communication and Information Systems (RCMCIS'02), Zegrze, Poland*, October 2002.

[Jah03] M. Jahnke. Schutz verteilter Intrusion-Detection-Systeme gegen Denial-of-Service-Angriffe. In *10. DFN-CERT/PCA-Workshop "Sicherheit in vernetzten Systemen"*, pages J–1–J–12, Hamburg, February 2003.

[JTBH04] M. Jahnke, J. Tölle, M. Bussmann, and S. Henkel. Components for Cooperative Intrusion Detection in Dynamic Coalition Environments. Presented at NATO/RTO IST Symposium "Adaptive Defence in Unclassified Networks", Toulouse, April 2004.

[Jul02] K. Julisch. Mining Alarm Clusters to Improve Alarm Handling Efficiency. In *Proceedings of the NATO/RTO IST Workshop on Inforensics and Incident Response, The Hague, The Netherlands*, October 2002.

[KTK02] C. Krügel, T. Toth, and C. Kerer. Decentralized Event Correlation for Intrusion Detection. *Lecture Notes in Computer Science*, 2288:114, 2002.

[LJT+04] M. Lies, M. Jahnke, J. Tölle, M. Bussmann, and S. Henkel. Ein Intrusion-Warning-System für dynamische Koalitionsumgebungen. In *Proceedings of the 11th DFN-CERT/PCA Workshop "Sicherheit in vernetzten Systemen", Hamburg*, February 2004.

[Sch02] H. Schmidt. Simulation und Erkennung der Ausbreitungsstrukturen von Würmern. Master's thesis, Universität Bonn, Institut für Informatik, 2002.

[TdW02] J. Tölle and C. de Waal. A Simple Traffic Model Using Graph Clustering For Anomaly Detection. Proc. of Applied Simulation and Modelling (ASM) Crete, Greece, June 2002.

[W3C99] W3C. W3C Recommendation 16: XSL Transformations (XSLT) Version 1.0. http://www.w3.org, 1999.