Tracefileanalysen für ns-2

Andreas Blüher
Hasso-Plattner-Institut, Universität Potsdam
Email: Andreas.Blueher@hpi.uni-potsdam.de

***Abstract*—In dieses Ausarbeitung sollen die Grundlagen der Tracefileanalyse für den Netzwerksimulator ns-2 erklärt werden. Beginnend mit einer Einführung in die Thematik, soll im Anschluss beleuchtet werden welche vielseitigen Informationen bei einer guten und gezielten Auswertung gewonnen werden können. Abschließend wird ein selbstentwickeltes Programm vorgestellt, mit dessen Hilfe gezielt wichtige Informationen aus den automatisch genierten Tracefile's gewonnen werden können.**

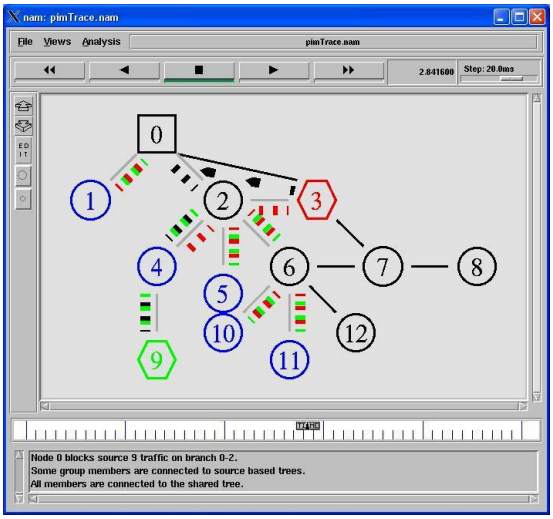
***Index Terms*—Tracefile, ns-2, Analyse, Auswertung**

I. Einführung

Der Netzwerksimulator ns-2 wird seit 1989 stetig weiter entwickelt und kann zur Simulation fast jeder Art von Netzwerk benutzt werden. Verschiedenste Netzwerk- und Routingprotokolle wurden integriert um dem Anwender eine Vielzahl von Einstellungsmöglichkeiten zu geben, damit alle möglichen Arten von Netzwerken mit ns-2 simuliert werden können. Mit Hilfe des Programms NAM kann die Simulation grafisch dargestellt und unter Berücksichtigung weniger Kriterien ausgewertet werden. Im Seminar „Attacken und Sicherheitsstrategien in mobilen Mesh- und Ad-hoc-Netzen" wurde ein alternativer Feedbackalgorithmus für drahtlose Netzwerke entwickelt. In dieser Ausarbeitung soll gezeigt werden mit welchen Mitteln die Analyse vorgenommen wurde und es soll der Nachweis erbracht werden wie die Erweiterung sowie die geschickte Kombination verschiedener Feedbackalgorithmen bisher vorhandene Nachteile in mobilen Netzwerken eliminieren kann.

II. Tracefile's in ns-2

In der Einführung wurde angesprochen das mit Hilfe von NAM (network animator) die Struktur und Abläufe in einer Netzwerksimulation besser veranschaulicht werden können als mit Hilfe von Text in den Tracefile's. Durch Grafiken können der Aufbau eines Netzes und vorhandene Verbindungen zwischen den Knoten schneller gezeigt werden als durch Angabe der Koordinaten. Auf der anderen Seite können in den Tracefile's viel mehr Informationen untergebracht werden als bei der grafischen Darstellung mit NAM, denn in einem Tracefile für mobile Netze ist beispielsweise zu jedem Paket genau definiert mit welchem Protokoll es gesendet wird, wie die Netzwerkadresse ist, wie viel Energie beim Versenden verbraucht wird und wieso es gedroppt wurde. NAM visualisiert also nur die Informationen aus dem Tracefile's die bei der grafischen Darstellung am relevantesten sind, so kann man gut die Struktur des Netzes auf dem Screenshot erkennen [2].



Die Daten die der Darstellung von NAM zugrunde liegen stammen aus den regulären Tracefile's, sodass man durch die geschickte Kombination von NAM und einer eigenen Analyse schnell viele Kerngrößen herausfinden kann, um möglicherweise Änderungen am Netzwerk vornehmen zu können.

Es existieren verschiedene Tracefileformate die diese sehr einfache Analyse enorm erschweren. Das bei der Einführung von ns-2 entwickelte Traceformat wurde für verdrahtete Netzwerke entwickelt und konnte den Anforderungen mobiler Netzwerke nicht mehr gerecht werden. Deswegen wurde ein neueres Format eingeführt, welches alternativ von ns-2 generiert werden kann. In einem mobilen Netzwerk sind die Koordinaten der Knoten, Energielevel und die Sendereichweite wichtige Informationen, während sie in normalen verdrahteten Netzwerken keine bzw. nur eine untergeordnete Rolle spielen. Je nachdem auf Grundlage welches Routingprotokolls die Übertragungen stattfinden, werden entsprechend den Besonderheiten des Protokolls unterschiedliche Tracefile's erstellt. Es existieren verschiedene Formate für AODV, DSR, TORA und DSDV.

Die dritte Art von Tracefile's ist für NAM entwickelt worden und enthält nur die zur Darstellung wichtigen Informationen die bei der Visualisierung des Netzwerkes benötigt werden. Bei der Ausführung von ns-2 wird ein Tracefile für NAM generiert und ein Tracefile entsprechend dem Netzwerk und des verwendeten Protokolls. Ausführlichere Informationen findet man unter [1]. Auf dem Bild sieht man den Aufbau eines NAM Tracefile beginnend mit den im Netz vorhandenen Knoten sowie ihrer Form und ihrer Farbe gefolgt von einigen Parametern wie der Grundfläche der Simulation, Tracefileversion, den Koordination der Knoten und abschließend die einzelnen Pakete die in der Simulation für den Anwender sichtbar verschickt werden.

```
n -t * -s 0  -x 0 -y 0 -Z 0 -z 5  -v circle -c black
n -t * -s 1  -x 0 -y 0 -Z 0 -z 5  -v circle -c black
n -t * -s 2  -x 0 -y 0 -Z 0 -z 5  -v circle -c black
n -t * -s 3  -x 0 -y 0 -Z 0 -z 5  -v circle -c black
n -t * -s 4  -x 0 -y 0 -Z 0 -z 5  -v circle -c black
n -t * -s 5  -x 0 -y 0 -Z 0 -z 5  -v circle -c black
V -t * -v 1.0a5 -a 0
W -t * -x 1000 -y 500
A -t * -n 1 -p 0 -o 0xffffffff -c 31 -a 1
A -t * -h 1 -m 2147483647 -s 0
n -t 0.000000 -s 18 -x 750.000000 -y 450.000000 -U 0.000000 -V 0.000000
n -t 0.000000 -s 19 -x 745.000000 -y 450.000000 -U 0.000000 -V 0.000000
n -t 0.000000 -s 20 -x 999.000000 -y 450.000000 -U 0.000000 -V 0.000000
n -t 0.000000 -s 21 -x 200.000000 -y 450.000000 -U 0.000000 -V 0.000000
n -t 0.000000 -s 22 -x 355.000000 -y 450.000000 -U 0.000000 -V 0.000000
n -t 0.000000 -s 23 -x 530.000000 -y 450.000000 -U 0.000000 -V 0.000000
+ -t 0.200000000 -s 5 -d -1 -p tcp -e 40 -c 2 -a 0 -i 0 -k AGT
- -t 0.200000000 -s 5 -d -1 -p tcp -e 40 -c 2 -a 0 -i 0 -k AGT
h -t 0.200000000 -s 5 -d -1 -p tcp -e 40 -c 2 -a 0 -i 0 -k AGT
+ -t 0.200000000 -s 5 -d -1 -p AODV -e 49 -c 2 -a 0 -i 0 -k RTR
- -t 0.200000000 -s 5 -d -1 -p AODV -e 49 -c 2 -a 0 -i 0 -k RTR
```

III. DER TRACEFILE ANALYZER

Im folgenden Abschnitt möchte ich den von mir entwickelten Tracefile Analyzer vorstellen und die einzelnen Funktionen beschreiben. Dazu werde ich aufzeigen wie einige Funktionen in c# realisiert wurden und danach erklären wie das Programm benutzt werden muss.

A. Auszuwertende Dateien

In den ns-2 Ordnern stehen nach einer erfolgreichen Durchführung einer Netzwerksimulation die Tracefiles für NAM sowie das reguläre Tracefile zur Verfügung. Durch die von uns vorgenommenen Anpassungen werden weitere Dateien relevant die ich im Folgenden kurz beschreiben möchte.

In der Datei „malicious_nodes.txt“ wird genau beschrieben welche Knoten im Netzwerk sich „böse“ verhalten sollen. Wie Alexander schon in seiner Ausarbeitung beschrieben hat gibt es mehrere Arten bösen Verhaltens, aber wir haben uns dafür entschieden das Droppen von Paketen zu implementieren, da es unserer Meinung die am häufigsten auftretende Variante ist. Weil die bösen Knoten nicht mehr im Quellcode benannt und festgelegt werden müssen, können neue böse Knoten in einer Simulationen hinzugefügt oder gelöscht werden ohne den Code wieder neu kompilieren zu müssen. Wenn man einen neuen Knoten in der Datei hinzufügen will muss man seine Knoten ID angeben und festlegen wieviel Prozent seiner Pakete ein Knoten droppen soll. Hierzu kann man genauer in der Ausarbeitung von Sebastian nachlesen, denn er hat den Algorithmus hierfür implementiert.

Die „blacklist.txt“ speichert in einer Liste alle Knoten die während der Simulation von einem anderen Knoten in die Blacklist geschrieben wurden. Dies ist bei der Auswertung sehr wichtig, denn wenn man die „malicious_nodes.txt“ mit der „blacklist.txt“ vergleicht erhält man im Idealfall dieselben Knoten. Das würde bedeuten, dass alle bösen Knoten von dem Feedbackalgorithmus als böse erkannt wurden, aber auch nicht mehr Knoten der Blacklist hinzugefügt wurden.
Der geänderte Algorithmus kann mit Hilfe verschiedener Parameter angepasst und optimiert werden, so kann man konfigurieren wie hoch die Reputation eines Knoten zu Beginn der Simulation sein soll, wieviel Reputation ein Knoten verliert wenn er nicht die Aktion durchführt die sein Knoten von ihm erwartet und ganz wichtig, welche Art von Feedback aktiv sein soll. Im groben beschreibt der neue Algorithmus wie man durch geschickte Kombination bereits vorhandener Feedbackalgorithmen ein wesentlich besseres Ergebnis errreichen kann als bisher. Zusammen mit allen anderen bereits genannten Konfigurationen kann man in der Datei „config.txt“ speichern ob One-Hop-Feedback, Two-Hop-Feedback oder/ und Reciever-Feedback aktiv sein soll.
Die letzte Datei die für die Auswertung verwendet wird ist auch zugleich die Datei mit den meisten Informationen. Es handelt sich um die „ns.tr“, dass reguläre Tracefile von ns-2. Das NAM Trace spielt in unserer Auswertung keine Rolle, da wir eine rein textbasierte Analyse betreiben und keine graphischen Informationen zum Netzwerk benötigen. Je nach Größe der Simulation kann dieses Tracefile auch über 100 Mb groß werden, was bei der Auswertung der Datei enorme Schwierigkeiten verursacht. Sollte eine Analyse fehl schlagen, könnte dies an einem sehr großen Tracefile liegen.

B. Textparsen mit Reguläre Ausdrücken

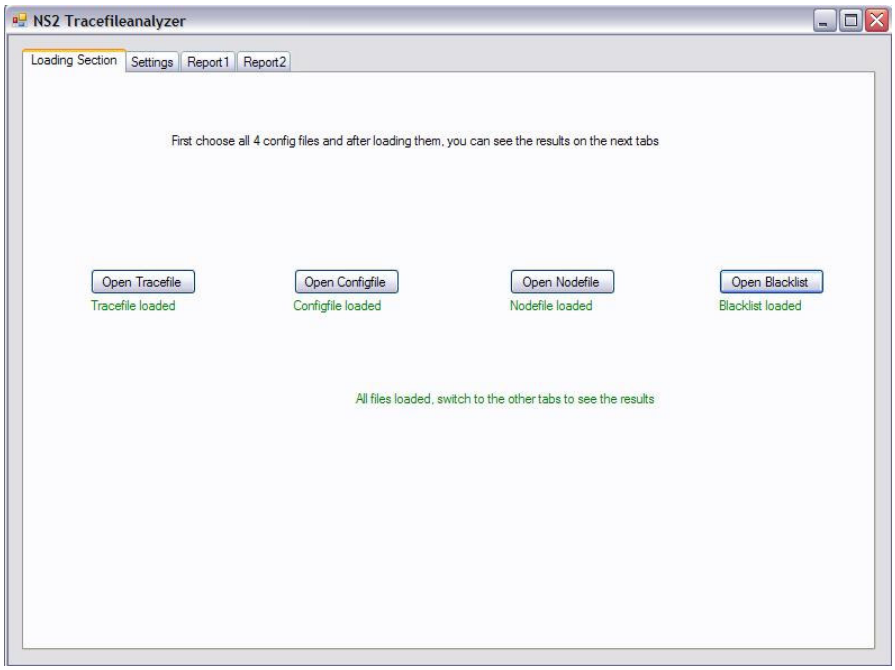
Kernstück der Anwendung sind Reguläre Ausdrücke oder wie sie im Englischen heißen: „Regular Expressions“. Mit Hilfe syntaktischer Regeln kann der Programmierer Zeichenketten anhand bestimmer Kriterien (Filter) aus einer großen Menge Zeichenketten herausfiltern. So ist es beispielsweise möglich, alle Wörter, die mit *S* beginnen und mit *D* enden, zu finden, ohne die zwischenliegenden Buchstaben explizit vorgeben zu müssen.
In unserer Anwendung werden reguläre Ausdrücke hauptsächlich zur Auswertung der normalen Tracefiles benutzt, da diese schnell einige Tausend Zeilen beinhalten können. Durch die Filterung ist es möglich das Tracefile unter verschiedenen Gesichtspunkten (Kriterien) zu analysieren und statistische Auswertungen vorzunehmen. Es ist sehr einfach möglich zu zählen wieviele Pakete in der gesamten Simulation gesendet, empfangen und gedroppt wurden, weiterhin kann man die gesendeten Pakete unterteilen und dort weitere Zählungen vornehmen. Ein zentraler Wert in unserer Analyse ist die Anzahl der Pakete die durch Knoten mit bösen Verhalten gedroppt werden. Im Tracefile erscheinen diese verloren gegangenen Pakete mit der Beschreibung „ATTACK“ sodass zuerst alle Pakete gefiltert werden die gedroppt wurden und anschließend die Pakete gezählt werden die die zutreffende Beschreibung haben. In dem angefügten Bild sieht man einen Ausschnitt aus dem Tracefile in dem eine Zeile durch rote Umrandung hervorgehoben ist. Diese Zeile beinhalten folgende Informationen: Der Knoten mit der ID 3 droppt in Sekunde 0.24209 ein Paket, weil der Knoten 3 wie man auf dem zweiten Bild sieht in der Datei „malicious_nodes.txt“ aufgelistet ist und mit einer Droppingrate von 100 alle Pakete verwirft die der Knoten erhält.

```
r -t 0.242065288 -Hs 3 -Hd 3 -Ni 3 -Nx 750.00 -Ny 10.00 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw --- -Ma 13a -Md 3 -Ms 5 -Mt 800
s -t 0.242075288 -Hs 3 -Hd -2 -Ni 3 -Nx 750.00 -Ny 10.00 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md 5 -Ms 0 -Mt 0
r -t 0.242090288 -Hs 3 -Hd 3 -Ni 3 -Nx 750.00 -Ny 10.00 -Nz 0.00 -Ne -1.000000 -Nl RTR -Nw --- -Ma 13a -Md 3 -Ms 5 -Mt 800
d -t 0.242090288 -Hs 3 -Hd 3 -Ni 3 -Nx 750.00 -Ny 10.00 -Nz 0.00 -Ne -1.000000 -Nl RTR -Nw ATTACK -Ma 13a -Md 3 -Ms 5 -Mt
r -t 0.242380118 -Hs 5 -Hd -2 -Ni 5 -Nx 999.00 -Ny 10.00 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md 5 -Ms 0 -Mt 0
```

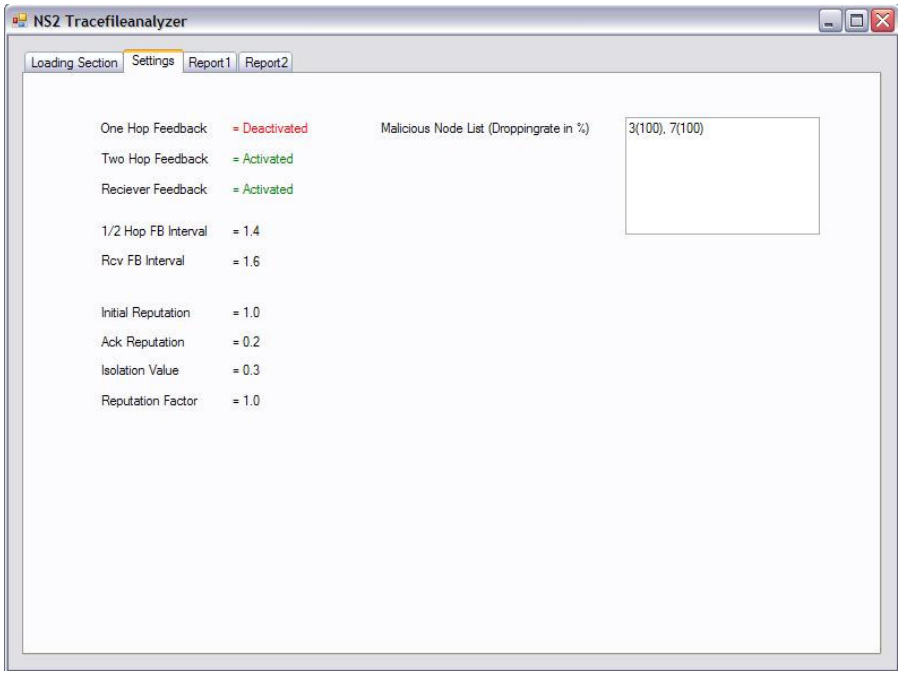
Die bei der Konfiguration des Netzes und des Feedbackalgorithmus gewählten Werte können entscheidenen Einfluss auf die erzielten Ergebnisse haben, deswegen wurde entschieden das zu jeder Analyse alle eingestellten Werte aus der „config.txt“ angezeigt werden, um bei einem weiteren Durchlauf gezielt Änderungen vornehmen und die Ergebnisse vergleichen zu können. Für eine spätere weiterentwickelte Version dieser Software wäre es denkbar die Werte im Programm änderbar zu machen, sodass die Dateien nur noch auf den Server kopiert und die Simulation neu durchgeführt werden müssen.

C. Die Anwendung des NS2 Tracefileanalyzer

Um die Software einsetzen zu können benötigt man mindestens das .NET 2.0 Framework, anderenfalls kann die Anwendung nicht gestartet werden. Desweiteren werden das Tracefile, die Configdatei, die Blacklist und Malicious_Nodes Datei benötigt, da die Analyse auf Grundlage dieser Dateien durchgeführt wird. Nach dem Start der Anwendung müssen die vier ebend genannten Dateien geladen werden. Sollte dies für alle Dateien erfolgreich durchgeführt worden sein, erscheint folgender Screen, anderenfalls ist ersichtliche bei welcher Datei es Probleme gab bzw. welche Datei noch nicht geladen wurde.



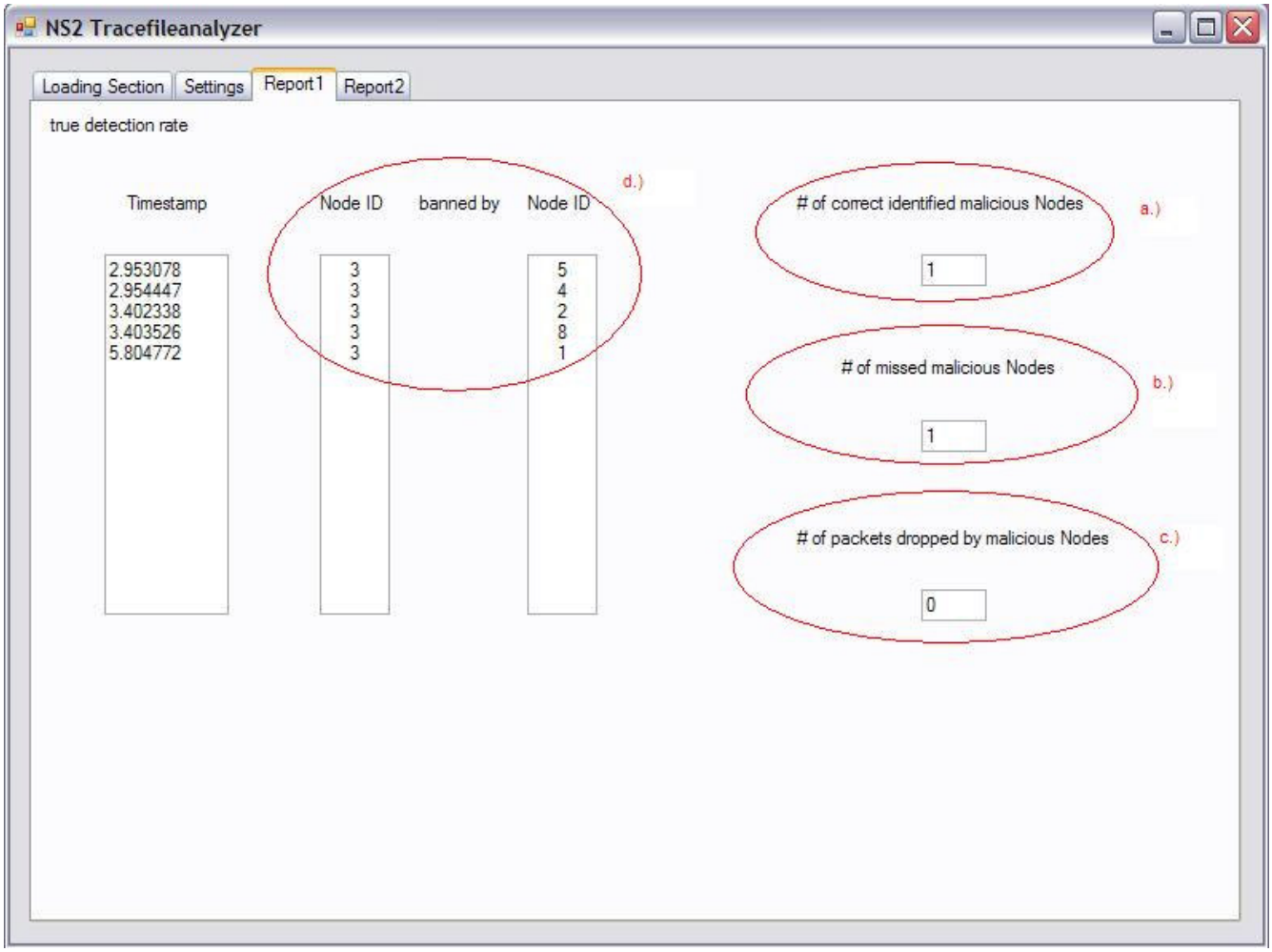
Wenn wie hier auf diesem Bild zu erkennen das Laden der Dateien erfolgreich durchgeführt wurde und mit der Nachricht „All files loaded, switch to the other tabs to see the results“ bestätigt wurde, kann man durch Klicken auf die anderen Reiter die entsprechenden Ansichten öffnen. In der Settingsansicht werden alle verfügbaren Konfigurationseinstellungen angezeigt um bei möglichen Modifikationen der Netze sofort überblicken zu können auf Grundlage welcher Werte diese Netzwerksimulation durchgeführt wurde. Die Änderung der Konfigurationswerte ist nicht möglich, könnte aber in einer späteren Erweiterung implementiert werden. Außerdem kann man in dieser Ansicht sehen welche Knoten als böse definiert wurden.



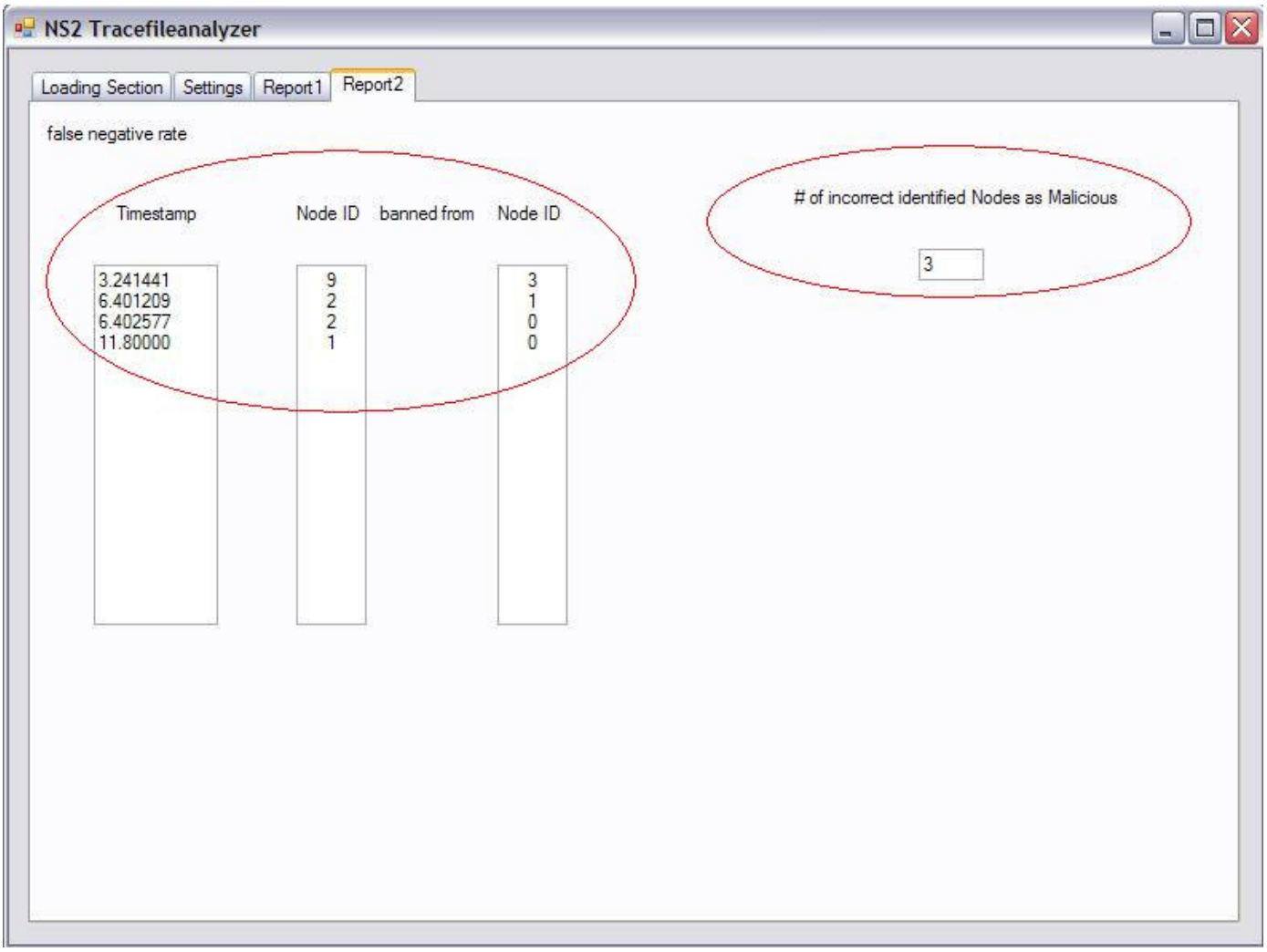
Die Ansichten Report1 und Report2 enthalten die entscheidenen Informationen die bei der Auswertung und dem Vergleich zwischen den verschiedenen Netzwerken helfen sollen. Ich habe die wichtigsten Werte hervorgehoben und durchnummiert und möchte nun im Einzelnen auf den Report1 eingehen:

a.) Gibt an, wieviele als böse definierte Knoten auch von einem beliebigen Knoten als bösartig erkannt wurden
b.) Gibt an, wieviele als böse definierte Knoten von keinem anderen Knoten als bösartig erkannt wurden
c.) Gibt an, wieviele Pakete währen der gesamten Simulation von als böse definierten Knoten gedroppt wurden
d.) Stellt dar, wann welcher bösartige Knoten korrekt von einem anderen Knoten erkannt wurde, dabei sind der genaue Zeitpunkt und die beiden Knoten IDs angegeben.

Wenn zuviele Pakete gedroppt wurden, spricht das dafür dass der Schutz entweder zu spät eingreift oder aber aufgrund von falschen Konfigurationseinstellungen nur mangelhaft funktioniert. Die hier aufgeführten Werte werden aus den verschiedenen zu Grunde liegenden Dateien geparsed.
In der hier gezeigten Analyse wird der Knoten 3 von 5 verschiedenen Knoten korrekt als bösartig erkannt und auf die interne Blackliste der jeweiligen Knoten geschrieben. Damit findet keine Kommunikation zwischen den Knoten und Knoten 3 mehr statt. Dass hier in diesem Beispiel 0 Pakete gedroppt werden konnten, lässt darauf schließen das die bösartigen Knoten sehr schnell erkannt wurden, aber es wurde auch der Einfachheit halber ein sehr einfaches Netz für die Beschreibung der Analyse gewählt. Die Tatsache, dass ein bösartiger Knoten nicht erkannt wurde, kann auf verschiedene Gründe zurück geführt werden, zum Einen kann es sein, dass dieser Knoten in der derzeitigen Simulation fast oder überhaupt nicht in den Nachrichtenaustausch verwickelt ist und so auch nicht durch sein böses Verhalten in Erscheinung treten kann.



Während im Report1 die Knoten aufgelistet werden, die wirklich als böse definiert wurden und auch von anderen Knoten als bösartig erkannt wurden, listet der Report2 alle die Knoten auf, die irrtümlich als böse Knoten identifiziert wurden. Grund dafür kann sein, dass beim Warten auf Feedback nicht lang genug gewartet wird, sodass der Nachbarknoten es technisch überhaupt nicht schaffen innerhalb der gegebenen Zeit zu antworten. Auch die Wartezeiten sind in der Settings Sektion einsehbar und können in der Konfigurationsdatei geändert werden.
Äquivalent zum Report1 wird im linken Teil der Ansicht aufgelistet welche Knoten zu welchem Zeitpunkt von wem irrtümlich gebannt wurden. Je mehr Knoten sich in dieser Liste befinden, desto schlechter funktioniert scheinbar der Algorithmus, aber hier spielen auch einige Faktoren aus dem NS2 herein, da nicht 100%ig geklärt ist, welche Seiteneffekte durch die Änderungen im aodv Algorithmus hier noch reinspielen.



IV. SCHLUSSFOLGERUNG

Bei der Simulation mit ns2 entstehen sehr große Datenmengen die durch einfaches Überblicken bzw. Lesen nicht mehr erfasst werden können. Mit der Entwicklung des Tracefileanalyzer steht einem Entwickler bei der Bearbeitung von Netzwerksimulationen ein Tool zur Verfügung mit dessen Hilfe er schnell die wichtigsten Kerngrößen erkennen und ablesen kann. Für andere Arten von Simulationen können andere Werte wichtig sein, aber die hier gewählten Elemente wurden für die Anforderungen im Seminar Vehicular- und Ad-hoc Networks festgelegt.

References

[1] NS-2 Trace Formats, 2007-05-28, http://nsnam.isi.edu/nsnam/index.php/NS-2_Trace_Formats
[2] NS-2 Implementation, 2007-05-30, http://marco.uminho.pt/~joao/pim-ns2/
[3] NS-2 Description, 2007-06-01, http://en.wikipedia.org/wiki/Ns2