Protecting Military Networks with GrADAR – Graph-based Automated DoS Attack Response

Marko Jahnke, Gabriel Klein, Jens Tölle
Fraunhofer Institut für Kommunikation, Informationsverarbeitung und Ergonomie (FKIE)
Wachtberg, Germany
E-mail: {jahnke, g.klein, toelle}@fgan.de

Peter Martini
Institute for Computer Science IV
University of Bonn
Bonn, Germany
E-mail: martini@cs.uni-bonn.de

ABSTRACT

This contribution presents an intuitive approach named 'GrADAR' for automatically selecting response measures to Denial-of-Service attacks and its application to military IP-based networks. GrADAR creates and maintains a model of a computer network and of the availability of its resources from the observations of deployed monitoring systems. Based on this model, it is possible to predict the impact of all applicable response actions on the resources and the users. Using appropriate metrics, comparing the outcomes of different responses helps to select the most appropriate one before it is actually applied. After presenting the approach in detail, this contribution discusses the deployment of the approach in military networks, especially for network services and infrastructure components in strategic and tactical networks.

Keywords: Denial-of-Service, Computer Network Defense, Automated Attack Response.

INTRODUCTION

Attacks against computer systems and networks in their different characteristics are omnipresent and a part of day-to-day business. Denial-of-Service (DoS) attacks aim at inhibiting platforms, networks and applications to provide the services which they are designed for. In the military domain, the availability of a service is one of the most important objectives of information assurance.

When an attack has been indicated by a monitoring system – such as a Network Management Systems (NMS) or an Intrusion Detection Systems (IDS) – Network Security Officers (NSOs) need to carefully select an appropriate response to the attack. The definition of this 'appropriateness' depends heavily on the properties and the deployment objective of the network and its components. Only a small number of approaches for selecting response mechanisms automatically and in a dynamic fashion exist; this is mainly due to the fact that poorly deployed or maintained automatic response systems may harm the network rather than mitigate the effects of an attack.

In contrast to many existing approaches, this contribution presents an intuitive and experimentally validated methodology to model the current network status and to estimate the impact of available response measures in order to select the most appropriate one for mitigating the effects of a detected denial-of-service attack.

IMPACT OF DOS ATTACKS IN MILITARY IP NETWORKS

As is true for all critical networks, DoS attacks may have a serious impact on military networks. Potentially, a DoS attack against network services or infrastructure components may lead to disturbed *business processes* which are the foundation of military tasks and missions. To estimate the potential DoS impact, it is useful to look at strategic and tactical environments separately, as they imply different application domains for attacks and defenses.

Strategic Environments

Strategic military networks could be compared to computer networks operated by large distributed companies or organizations, including their external partners or suppliers. These networks typically rely on fixed infrastructures, including wired links (LAN, WAN) and radio links (point-to-point or access point-based). Compared to tactical networks, the subnetworks are relatively stable in structure and topology, high-bandwidth connections are widely available, and IP networking compatibility can be generally assumed.

The critical business processes realized in strategic military networks include – but are not limited to – logistics and deployment, medical supply, and strategic command and control. It is easy to see that blocking or slowing down these processes may have serious impact, not only on strategic tasks, but also on tactical missions which rely on the strategic domain.

The business processes are usually realized using *services* (including hardware and software services, but also services provided by human personnel). As a consequence of introducing the notion of Service-Oriented Architectures (SOA), there is an obvious trend to integrate these services into larger conceptual frameworks in order to provide a higher degree of interoperability, easier maintenance, and re-usage of services. This, in turn, leads to a higher degree of interdependency between the different services and hence the business processes realized by them. Reducing the availability of just one single highly important service (e. g., access control or directory services) could have a huge impact on many other services, since they rely on the attacked one.

The threats against the availability of these services include *outsider attacks* (e. g. service overloads, server crashes) and different kinds of *insider attacks* (including end-user application or data misuse and sabotage of the network infrastructure components). Due to the deployment of IP network technology, DoS attacks known from the public Internet are potentially applicable in strategic networks as well, since they are comparable in terms of size and heterogeneity. Examples include DoS-capable malware, e. g., worms and viruses, which could turn end-user computer systems, servers, and potentially infrastructure equipment into botnets or zombie armies. Recent cases have shown that – e. g., using removable storage media – malware can easily be transferred to protected networks, even classified ones, unless there is a long list of properly maintained security measures in place.

Tactical Environments

IP networks in tactical environments are used to realize tactical business processes, for example tactical command and control, generation of a common operational picture (COP) on different layers of the command chain, and ammunition and equipment logistics. The services which are needed for building these processes may also be highly interdependent, and DoS attacks against basic services may have serious impact.

Tactical networks are usually not as big as the strategic part of a military network, but there are different aspects which make them even more difficult to defend against DoS attacks. Defending the network becomes even harder if coalition/joint missions require interconnecting the tactical networks of different nations. The degree of heterogeneity in terms of technology and procedures increases even more.

An example of a tactical communications scenario looks as follows. Three infantry platoons are connected to a battalion command post via radio links. The platoon radio subnets can be of arbitrary structure, e. g., ad hoc or centralized. The service landscape in such a scenario can be very diverse and include, for example, voice-over-IP and command & control information system (C2IS) services. These services are often heavily interdependent and the

C2IS service might rely on GPS tracking, health and equipment monitoring, and others to function correctly.

Consider the following: Through a platoon gateway node, an attacker in one of the subnets performs a DoS attack against the battalion command post communications server (e. g., by sending an excessively large number of requests). As an effect of this attack, the availability of the C2IS service throughout the network is significantly reduced. This is because the central communication server is no longer able to distribute C2IS data to its clients. A presumably good reaction to such an attack could be to disconnect the attacker's entire subnet from the rest of the network by severing the radio link to the platoon gateway. The expected effect of such a response is restored C2IS availability in the entire network – with the exception of the attacker's subnet – because its nodes do not have access to the rest of the network.

However, the actual result of disconnecting the attacker's subnet in this example will not be an availability increase of the C2IS service in the entire network because each C2IS client is also dependent on aerial reconnaissance data provided by a UAV which is controlled and accessed through an operator also located in the attack's subnet. Thus, from mission view, the application of this response measure will have little to no positive effect. Therefore, the selection of an appropriate response measure needs to be preceded by a careful evaluation of service dependencies.

THE GRADAR APPROACH

The GrADAR (Graph-based Automated DoS Attack Response) approach is based on creating and updating model instances of the relevant resources in the network. The following subsections present the methodology in a formal way.

Resources, Availability, and Dependency

As already suggested by Toth and Kruegel [4], our model is based on properties of functional components – our *resources*. We denote the set of resources as $\mathcal{R}$. Resources can be either service instances (instances of a service provided by hardware, operating systems, applications, local or network services) or users. The corresponding sets are henceforth denoted as $\mathcal{S}$ and $\mathcal{U}$, respectively.

We assume that each resource $r \in \mathcal{R}$ of a system to be secured has certain availability, expressed as a value $A(r) \in [0, 1]$. This value may be observed, e. g., as the time needed for a transaction with the respective resource, such as request-response delays, or as the number of transactions performed per time period. As an example: if a router is able to handle only 10 % of the traffic it was designed for, its current availability is denoted as 0.1. The current availability value of a resource is assumed to be the result of two independent factors: its internal state and the values of other associated resources. Thus, we separate the intrinsic availability value $A_I(r)$ from the propagated availability value $A_P(r)$. An example for intrinsic availability is the ability of a HTTP server to respond to requests, while a propagated availability is associated with the responsiveness of a DNS resolver that the HTTP server needs for looking up the IP addresses of its clients. We define the resulting availability as

$$A(r) = A_I(r) \cdot A_P(r)$$

for each resource $r \in R$. Concerning the availability, we observe different kinds of dependencies between resources. As an example: many communication systems depend on the availability of directory services or back-end databases. These dependencies may differ in strength (the availability loss of a resource may have a stronger impact on a dependent

resource than another) and in type (e. g., a resource needs access to each element of a set of resources, or it is enough to access one element). Resource and service dependencies have been studied in the area of network management for many years (see e. g. [13], [14]).

In our formalism, a resource r depends on resources s_1, ..., s_q in terms of availability (denoted as r $\triangleright$ s_1, ..., $r \triangleright s_q$), if r's propagated availability is given by a dependency function $D_r : [0, 1]^q \rightarrow [0, 1]$ and corresponding weight functions $w_{r,s} : [0, 1] \rightarrow [0, 1]$ with

$$A_P(r) = D_r\left(w_{r,s_1}\left(A(s_1)\right),\ldots,w_{r,s_q}\left(A(s_q)\right)\right)$$

such that $D_r\left(w_{r,s_1}(1),\ldots,w_{r,s_q}(1)\right) = 1$ (definition of optimal conditions).

Dependency weights are needed to express the strengths of dependencies. Dependency functions express the kind of access strategy; a set of frequently observed dependency functions include mandatory dependency (r is not able to perform its task without having access to s) and alternative dependency (r performs well as long as it has access to one of the resources s_1, ..., s_q). In case of multiple dependencies, these may either be sequential (r accesses s_1, ..., s_q one after the other) or parallel (r accesses s_1, ..., s_q simultaneously).

Dependency Graph and Accessibility Graph

A *dependency graph* f a system with the set $\mathcal{R}$ of resources is a directed acyclic graph $\hat{G} = \left(\hat{V}, \hat{E}\right)$ with $\hat{V} \subseteq \mathcal{R}$ and $\hat{E} \subseteq \left((\mathcal{U} \cup \mathcal{S}) \times \mathcal{S}\right)$. $\hat{G}$ contains an edge (r, s) iff. r $\triangleright$ s. The vertices of $\hat{G}$ are labeled with the corresponding dependency function D_r, and the edges with the weight functions $w_{r,s}$. The dependency graph is used as an idealized map of the network. It reflects the requirements for a non-derogated state of all components. Let the set of dependency graphs be denoted as $\hat{\mathcal{G}}$.

An *accessibility graph* of a system with the set $\mathcal{R}$ of resources is a directed acyclic graph $G = (V, E)$ with $V \subseteq \mathcal{R}$ and $E \subseteq \left((\mathcal{U} \cup \mathcal{S}) \times \mathcal{S}\right)$. G contains an edge (r, s) whenever a resource s is directly accessible from r. The vertices of G are labeled with their availability values. To emphasize that the availability values belong to G, we denote the values as A (G, r), A_I (G, r), or A_P (G, r), respectively. The accessibility graph is used as a simplified model of the real-world network to represent current status information either as provided by a monitoring system or as estimated by an update algorithm (see below). Let the set of accessibility graphs be denoted as $\mathcal{G}$. To quantify the overall availability of the network when supporting users in conducting a specific mission, we define it as the weighted average of the availability of the user vertices in G:

$$A(G) = \sum_{u \in \mathcal{U}} m(u)\, A(G,u)$$

with $\Sigma_{u \in \mathcal{U}}\, m(u) = 1$. $m(u)$ is the *relative importance* of the user $u \in \mathcal{U}$ for the common mission, that either needs to be defined beforehand or to be determined adaptively. This is necessary to express different degrees of importance of users, e. g., as a result of different user roles (e.g., infantry soldier, platoon commander, system administrator).

Update Algorithm

To create a comprehensive view on the availabilities of the resources in the network, it is necessary to update the accessibility graph as soon as new observations arrive. Let $V_O(G) \in V$ be the set of vertices for which observed availability values exist. The update

procedure as such needs to update the availability values of every single vertex $r \in V$ as follows:

- Update the propagated availability $A_P(G, r)$ as $A_P(r) = D_r\left(w_{r,s_1}\left(A(s_1)\right),\ldots,w_{r,s_q}\left(A(s_q)\right)\right)$ (see above).
- If $r \in V_O(G)$, then use the provided availability value $A(G, r)$ to adjust the intrinsic availability value as $A_I(G,r) = A(G,r) / A_P(G,r)$, as long as $A_P(G,r) \neq 0$.
- If $r \notin V_O(G)$, then estimate the overall availability value as $A(G,r) = A_I(G,r) \cdot A_P(G,r)$.

One possible algorithm to perform the above update procedure is based on a parallel depth-first search (DFS), starting from the user vertices in the accessibility graph, and terminating at vertices without any outgoing edge (i. e. $deg_{out}(r) = 0$). This behavior guarantees that all relevant parts of the graph are taken into account and that the value of a vertex can only be upgraded if the values of all of its parents are already upgraded.

We have created a prototypical implementation of the DFS-based algorithm. It reads the description of the dependency and accessibility graphs as well as their changes as results of response measures and updates the availability values accordingly.

Modeling Response Measures

Given the current dependency graph $\hat{G}$ and accessibility graph G, we define a *response measure* or *countermeasure* as a transformation

$$\theta : \left(\hat{\mathcal{G}} \times \mathcal{G}\right) \rightarrow \left(\hat{\mathcal{G}} \times \mathcal{G}\right)$$

such that for $\theta\left(\hat{G}, G\right) = \left(\hat{G}', G'\right)$, the following conditions hold:

- $\hat{G}' = \left(\hat{V}', \hat{E}'\right)$ is derived from $\hat{G}$ by adding or removing vertices or edges,
- $G' = \left(V', E'\right)$ is derived from G by adding or removing vertices or edges, or by changing availability values of the vertices.

For each change of the accessibility graph G', the above update algorithm needs to be applied, so that availability values to be set by the response are treated as observed values. The set of possible responses is denoted as Θ.

A response measure θ might be divided into $N_\theta \in \mathbb{N}^+$ non concurrent elementary *response steps* $\theta^{(i)}$:

$$\theta = \theta^{(1)} \circ \cdots \circ \theta^{(N_\theta)}$$

Each of the steps $\theta^{(i)}$ corresponds to one transformation primitive (setting availability values, adding or removing vertices or edges) which itself is associated either with an immediate impact of one practical action that is applied to the real-world system (so-called *explicit impact*) or with a subsequent change of the environment (*implicit impact*). Note that these impacts need to be determined for each response by observation or by analysis.

For $\theta = \theta^{(1)} \circ \cdots \circ \theta^{(N_\theta)}$, we define the *intermediate* dependency and accessibility graphs $\left(\hat{G}^{(i)}, G^{(i)}\right)$ by

$$\theta^{(i-1))}\left(\hat{G}^{(i-1)}, G^{(i-1)}\right) = \left(\hat{G}^{(i)}, G^{(i)}\right)$$

for all $i = 1,\ldots,N_\theta$ with $\hat{G}^{(0)} := \hat{G}$ and $G^{(0)} := G$. Thus, they are the corresponding results after application of each step.

Given that all relevant properties of the response action are expressed in the respective graph structures, it is possible to define three of the four mentioned practically relevant metrics that have been discussed in [10]. Let $\hat{G} = \left(\hat{V}, \hat{E}\right)$ and $G = \left(V, E\right)$ be the dependency and accessibility graphs before applying a response $\theta\left(\hat{G}, G\right) = \left(\hat{G}', G'\right)$.

- *Expected Response Success*. We define the expected success of a response as the difference of the overall availability before and after the application of θ:
$$\delta_S\left(G, G'\right) := A\left(G'\right) - A\left(G\right)$$
Intuitively, for a successful application of the response, a positive value is the result; if it is zero or below, the response has failed or has no value.
- *Expected Response Costs*. We define the costs of applying a response θ to the graph G as the accumulated observed application delays $t_{app}\left(\theta^{(i)}\right) \in \mathbb{R}^+$ of the response steps, multiplied with the intermediate availability losses $(1 - A(G^{(i-1)}))$ as the respective result of the previous step:
$$\delta_C\left(G, G'\right) := \sum_{i=1}^{N_\theta} \frac{t_{app}\left(\theta^{(i)}\right)}{t_{app}{}^{max}} \cdot \left(1 - A\left(G^{(i-1)}\right)\right)$$
with an arbitrary but maximum application delay $t_{app}{}^{max} \in \mathbb{R}^+$ (used for normalization). Thus, a response is more expensive, the more time it takes to be applied, and the more intermediate availability loss it involves.
- *Expected Error-Proneness*. Given that all resources necessary for applying responses are also considered as vertices and dependencies in the graphs (e. g. management consoles, remote shells, necessary network connections), it is possible to add a user vertex u_{Admin} that depends on them and represents the security administrator who actually applies a response. By defining
$$\delta_E\left(G, G'\right) := \left(1 - A\left(u_{Admin}\right)\right),$$
the error-proneness is given by the availability of the security administrator vertex. Fortunately, the update of these values is already handled by the propagation algorithm as described in the previous subsection, since u_{Admin} is treated as an ordinary user vertex.
- *Expected Response Durability*. If there is a time limit $t_{dur}(\theta, G)$ for the existence of the restored availability of the graph (e. g. due to limited energy resources or due to the deployment of deprecated hardware), this limit needs to be used as the *durability* of the response action:
$$\delta_D\left(G, G'\right) := \min_{i=1,\ldots,N_\theta} t_{dur}\left(\theta^{(i)}\right).$$
For simplicity reasons, it is useful to restrict the set of potential values for the durability metric to a small set of [0, 1]-normalized values such as $\delta_D(\mathrm{G}, \mathrm{G}') \in \{10^{-2}, 10^{-1}, 1\}$, expressing that immediate, mid-term, or no additional actions are required to keep the current state. Note that for this metric, there is no obvious way to determine the corresponding value, other than determining the lifetime of each elementary response step result informally.

Response Selection

Finally, the selection process for the most appropriate real world response action is performed as follows. Let the set of metrics be defined as above, i. e., $\{\delta_S, \delta_C, \delta_E, \delta_D\}$, and the set of available responses be $\Theta = \{\theta_1, \ldots, \theta_n\}$.

The optimal real-world response is the one that corresponds to the graph response $\theta \in \Theta$ with
$$\lambda = \arg\min_{\theta \in \Theta} w_C \delta_C\left(G, G'\right) + w_E \delta_E\left(G, G'\right) - w_S \delta_S\left(G, G'\right) - w_D \delta_D\left(G, G'\right),$$
$\theta\left(\hat{G}, G\right) = \left(\hat{G}', G'\right)$, and weights $w_C, w_E, w_S, w_D \in \mathbb{R}$ to express the relative importance of the four metrics. Note that for the HB (higher is better) metrics δ_S and δ_D, the values are sub-

tracted from rather than added to the overall value as for the LB (lower is better) metrics δ_C and δ_E.

Experimental Validation

The GrADAR approach has been successfully validated using practical experiments as presented in [11]. The experiments were conducted in an e-commerce environment with different interacting services, such as HTTP, FTP, IRC, and DNS servers, as well as a content management system (CMS) and a back-end database, corresponding to 45-vertex dependency and accessibility graphs. The validation was divided into two parts: *In vitro experiments* for determining the dependency and weighting functions for the different resources, and *in vivo experiments* for comparing the predicted availability values under realistic conditions, including attacks and different response actions.

As a result, it can be safely stated that it is possible to predict the user-perceived availability of the network for different response alternative, as long as the corresponding graph transformations of the alternatives have been determined sufficiently exact. However, if interactions between the resources are not expressible in terms of availability propagation, the availability prediction could fail and thus a proper selection would not be possible. A potential extension of the methodology for overcoming this challenge is currently under development (see [12]).

For more information about the validation and its results, please refer to [11].

DEPLOYING GRADAR IN MILITARY IP NETWORKS

Deployment in Infrastructure-based Networks

One way to implement this concept in infrastructure-based IP networks is to use a GrADAR implementation in combination with existing NMSs/IDSs. NMS are used in many military networks (both strategic and tactical) for automatically monitoring and adjusting performance parameters and for alerting the NSO in cases of failures. IDSs are deployed for detecting both general attacks against the network and its components as well as misbehavior of users. This is the case for connection points between strategic networks (including unclassified networks and the public Internet), but also for the temporary interconnection of different tactical networks, such as those used for joint missions. To create the GrADAR dependency graph, it is first necessary to identify the vital services and the resources which provide them. For each of the resources, a definition of availability needs to be identified. For many types of resources, a definition based on the time needed for a transaction performed by the resource is useful, such as the delay of a response to an ICMP echo request for the corresponding IP stack, a request-response delay for an HTTP server, or a delay for VoIP packets.

Ideally, an NMS provides up-to-date availability information about these resources. If these cannot be obtained, additional sensors can be easily implemented, e. g. using scripting languages. By measuring and manipulating these availability values over a certain training period, it is possible to determine the dependencies between the resources in terms of arithmetic functions. This procedure has been discussed in the literature in the context of network management before. The accessibility graph representing the current state of the network is created as a result of the current observations of the NMS (and an IDS, if this is able to deliver robust indications of degraded service availabilities). The graph is completed by the GrADAR update algorithm which estimates unavailable measurement values by propagating existing values according to the previously determined dependency functions. To select the most appropriate response, each of the alternatives needs to be applied to the current model instance. This means that actions like shutting down a firewall port in the real

world lead to a reduced availability of the corresponding vertex in the resulting graph. Using appropriate selection metrics (see above), the most appropriate alternative response would be chosen automatically, which corresponds closely to the decision process of an experienced human administrator.

If the NMS is able to apply administrative reconfiguration actions using its internal command and control infrastructure and distributed NMS agents, they can be triggered according to the selected response measure (including its parameterization). However, there might still be different challenges for the application of the monitoring systems in large-scale or extremely heterogeneous network environments (see above), which are not further addressed in this contribution.

Deployment in Mobile Ad hoc Networks

Mobile ad hoc networks (MANETs) are an example for military networks with a high level of resilience and self-optimization capabilities. Networks of this kind comprise a number of mobile nodes which are connected using a radio broadcast medium. The most prominent property of MANETs is that each of the nodes is able to route and forward IP traffic for other nodes (multi-hop routing).

To implement the GrADAR approach in such a decentralized fashion – e. g., as part of a distributed intrusion detection system (such as [16]) – it becomes necessary for multiple network nodes to build and maintain their dependency and accessibility graphs according to their own observations. In this case, the nodes need to analyze the (passively observed) network traffic and to determine, e. g., transaction delay times and success rates by correlating the network packets. As a consequence of the autonomy of the model instances (e. g., maintained by neighbor network nodes), the possibility of having divergent or even contradicting observations needs to be discussed. Especially in cases where partially overlapping observations may occur (e. g. due to transmission ranges in radio networks) and where autonomous decisions on countermeasures may be made, the question of coordination arises. Future work should address these harmonization issues.

CONCLUSION AND FUTURE DIRECTIONS

Denial-of-Service attacks in military networks may have a serious impact on business processes and subsequently on general military tasks and missions. There are many different possibilities for performing these kinds of attacks in strategic and tactical IP networks running the services which are realizing these business processes, and in many cases, these cannot be ultimately excluded.

Reliable automated response initiation against DoS attacks would improve the resilience of these services significantly, since the time where the priority services are not available can be kept small. GrADAR provides an approach for selecting appropriate response measures to DoS attacks, based on up-to-date availability information from monitoring systems and from knowledge about the inter-dependencies of the resources by applying the alternative response measures to a network model and predicting the impact. Implementing this approach would fulfill the requirements for reliable automated DoS response initiation, since the decision is fast (due to a simple model) and reliable (since it is based on objectively measured availability values).

Currently, we are working on an extension that allows more complex interactions between the resources in the model, especially the relationship of availability, workload and processing capability effects.

BIBLIOGRAPHY

[1] N. Stakhanova, S. Basu, and J. Wong. A Taxonomy of Intrusion Response Systems. In: Int. Journal of Information and Computer Security Vol. 1(1), pp. 169–184 (2007).

[2] The Snort Inline Project Team: Snort Inline Project Homepage. Online accessible at http://snort-inline.sourceforge.net/ (2007).

[3] W. Lee, W. Fan, M., Miller, and S. Stolfo. Toward Cost-Sensitive Modeling for Intrusion Detection and Response. In: Journal of Computer Security Vol. 10, pp. 5–22 (2002).

[4] T. Toth and C. Kruegel. Evaluating the impact of automated intrusion response mechanisms. In: Proc. of the 18th Computer Security Applications Conference (ACSAC'02), Las Vegas, NV, USA (2002).

[5] I. Balepin, S. Maltsev, J. Rowe, and K. Levitt. Using specification-based intrusion detection for automated response. In: Proc. of the 6th International Symposium on Recent Advances in Intrusion Detection (2003).

[6] F. Wu, B. Foo, Y. Mao, S. Bagchi, and E. Spafford. Automated adaptive intrusion containment in systems of interacting services. In: Computer Networks Vol. 51(5), pp. 1334–1360 (2007).

[7] N. Stakhanova, S. Basu, and J. Wong. A Cost-Sensitive Model for Preemptive Intrusion Response Systems. In: Proc of the IEEE International Conference on Advanced Information Networking and Applications, Niagara Falls, Canada (2007).

[8] J. Mirkovic, A. Hussain, B. Wilson, S. Fahmy, P. Reiher, R. Thomas, W. Yao, and S. Schwab. Towards User-Centric Metrics for Denial-Of-Service Measurement. In: Proc. of the Workshop on Experimental Computer Science (2007).

[9] M. Jahnke, C. Thul, and P. Martini. Graph-based Metrics for Intrusion Response in Computer Networks. In: Proc. of the 3rd IEEE LCN Workshop on Network Security, Dublin, Ireland (2007).

[10] M. Jahnke, C. Thul, and P. Martini. Comparison and Improvement of Metrics for Selecting Intrusion Response Measures against DoS Attacks. In: Proc. of the "Sicherheit2008" conference, Saarbrücken, Germany (2008).

[11] M. Jahnke, J. Tölle, C. Thul, and P. Martini. Validating GrADAR – Graph-based Automated DoS Attack Response. To be published in: Proc.of the the 34th IEEE Conference on Local Computer Networks (LCN), Zurich, Switzerland (2009).

[12] G. Klein, M. Jahnke, J. Tölle, and P. Martini. Enhancing Graph-based Automated DoS Attack Response. In: Proc. of the Conference on Cyber Warfare, Cooperative Cyber Defence Centre of Excellence (CCD-CoE), Tallinn, Estonia (2009).

[13] S. Bagchi, G. Kar, and J. Hellerstein. Dependency analysis in distributed systems using fault injection: Application to problem determination in an e-commerce environment. In: Proc. 12th Intl. Workshop on Distributed Systems: Operations & Management (2001).

[14] P. Bahl, P. Barham, R. Black, R. Chandra, M. Goldszmidt, R. Isaacs, S. Kadula, L. Li, J. Maccormick, D. Maltz, R. Mortier, J. Wawrzoniak, and M. Zhang. Discovering Dependencies for Network Management. In: Proc. of the 5th HotNets Workshop (2006).

[15] T. Cormen, C. Leiserson and R. Rivest. Inroduction to Algorithms. Chapter IV: Graph Algorithms. MIT Press, 1990.

[16] M. Jahnke, A. Wenzel, G. Klein, N. Aschenbruck, E. Gerhards-Padilla, P. Ebinger, and S. Karsch. MITE – MANET Intrusion Detecion for Tactical Environments. In: Proc. of the NATO/RTO Research Symposium on Information Assurance for Emerging and Future Military Systems. IST-076, Ljubliana, Slovenia (2008).