Validating GrADAR – an Approach for Graph-based Automated DoS Attack Response*

Marko Jahnke, Jens Tölle, and Christian Thul
Research Institute for Communication,
Information Processing and Ergonomics (FGAN-FKIE)
Wachtberg, Germany
Email: {jahnke,toelle,thul}@fgan.de

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

***Abstract*—This contribution presents validation results of an intuitive approach named 'GrADAR' for automatically selecting response measures to DoS attacks. It creates and maintains a model of a computer network and of the availability of its resources from the observations of deployed monitoring systems. The graph-based model is able to express both the effects of DoS attacks and response measures as reactions to the attacks. Certain properties of the model graphs are utilized to determine different metrics which are well-known from the pragmatic decisions of network security officers.**

I. Introduction

Attacks against computer systems and networks in their different characteristics are omnipresent and a part of day-to-day business. Almost every network that connects computers has already been subject to acts of reconnaissance, penetration, stealing, or damaging information of all kinds, with more or less serious subsequent effects. Properly configured and maintained distributed monitoring systems (Intrusion Detection Systems/IDS, Network Management Systems/NMS) provide comprehensive views on the network health and are able to identify many types of attacks.

When an attack has been indicated by a monitoring system, network security officers need to select an appropriate response to the attack carefully. The way to define this 'appropriateness' depends heavily on the properties and the deployment objective of the network and its components. Only a small number of approaches exist for selecting response mechanisms automatically and in a dynamic fashion; 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 the existing approaches (for a discussion of existing work, please refer to [1], [5], [3]), this contribution presents the experimental validation of an intuitive 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 (DoS) attack.

II. The GrADAR Approach

The GrADAR approach has been presented and discussed in different publications [2], [3], [4], [5], and thus we restrict ourselves to explain its basic properties here. In Fig. 1, a general overview of the approach is depicted.

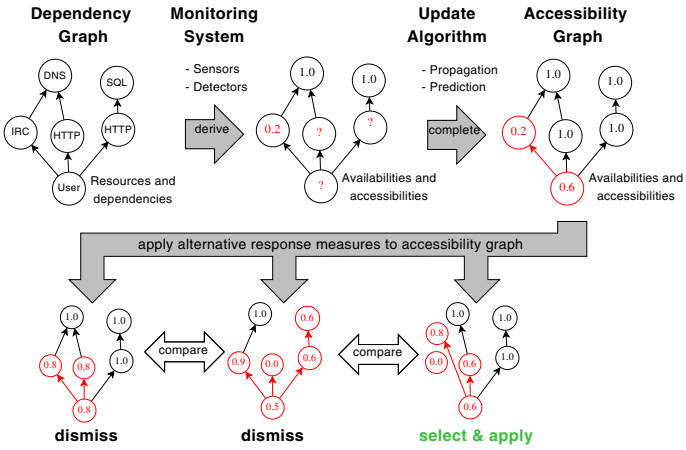

Fig. 1. Schematic overview of the GrADAR approach.

GrADAR 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*.

We assume that each resource $r \in \mathcal{R}$ has a certain *availability*, expressed as a value $A(r) \in [0, 1]$. This value may be observed, e. g., as the time needed for a typical transaction with the respective resource (e. g. request-response delays), or as the number of transactions performed per time period. 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 $A_P(r)$ such that $A(r) = A_I(r) \cdot A_P(r)$.

In GrADAR, information about a resource's dependencies on other resources (in terms of availability) is expressed in a so-called *dependency graph* $\hat{G}$, where edges are associated with *weight functions* $w_{r,s}$, and vertices are labeled with the *dependency functions* D_r of the corresponding resource so that $A_P(r) = D_r(w_{r,s_1}(A(s_1)), \ldots, w_{r,s_q}(A(s_q))$. Up-to-date availability information from a monitoring system

* Published in: Proc. of the 34. IEEE Conference on Local Computer Networks (LCN2009), Zurich, Switzerland, Oct. 2009.

is used to create another graph structure, the *accessibility graph* G, which expresses the current state of the network. Vertices are attributed with the current availability $A(r)$ of the corresponding resource, and an edge (r, s) exists iff. r is actually able to access s.

Using an appropriate update algorithm, it is possible to estimate the availability values of resources where the monitoring system is not able to deliver observations. Additionally, it is possible to estimate the availability of the users. The overall availability of the network to be protected is given as

$$A(G) := \sum_{u \in \mathcal{U}} m(u) \cdot A(u) \tag{1}$$

where $\mathcal{U}$ is the set of users, and $m(u)$ is a weight for each user, reflecting his importance for the common mission that is supported by the network.

By comparing properties of the accessibility graph G after an attack and the accessibility graph G' after the application of a response θ with $\theta(\hat{G}, G) = (\hat{G}', G')$, it is possible to quantify the effects of θ on the current system state in terms of different practically relevant metrics, including *response success*, *application costs*, *error-proneness*, and *durability* (henceforth denoted as functions $\delta_S, \delta_C, \delta_E, \delta_D$ of the accessibility graphs G after the attack and G' after the application of the respective response).

For more detailed information about the approach and our prototypical implementation, please refer to [5]. An extension for enhancing the availability update algorithm using workload propagation has been suggested in [4].

III. Experimental Validation

The goal of the experimental validation was to show that the metric values for the application of the response in a real-world system are close to the predicted values obtained from the update algorithm after the application of the response's counterpart in the accessibility and dependency graphs.

A. Validation Scenario and Availability Definition

As a scenario, we selected an e-commerce DMZ setup with an HTTP/PHP-based webshop CMS (content management system) and an IRC discussion board for customer support. Different supporting resources are needed (e. g. the operating system, network adapters, and different network services) are also included in the 45-vertex setup.

The setup is depicted in Fig. 2. The hosts include the servers 'S1' and 'S2' as well as the packet filter 'PF' and the client platform 'C' which may not be observed by the monitoring system. In an e-commerce scenario, the availability of the webshop and the customer support service is the most important goal. It may be lowered either by transaction delays (e. g. due to high workload of the server) or by a low transaction success rate (e. g. due to high network traffic load or a crashed server).

We determined the *transaction delay-based availability* of our resources using normalized transaction delays in the shape

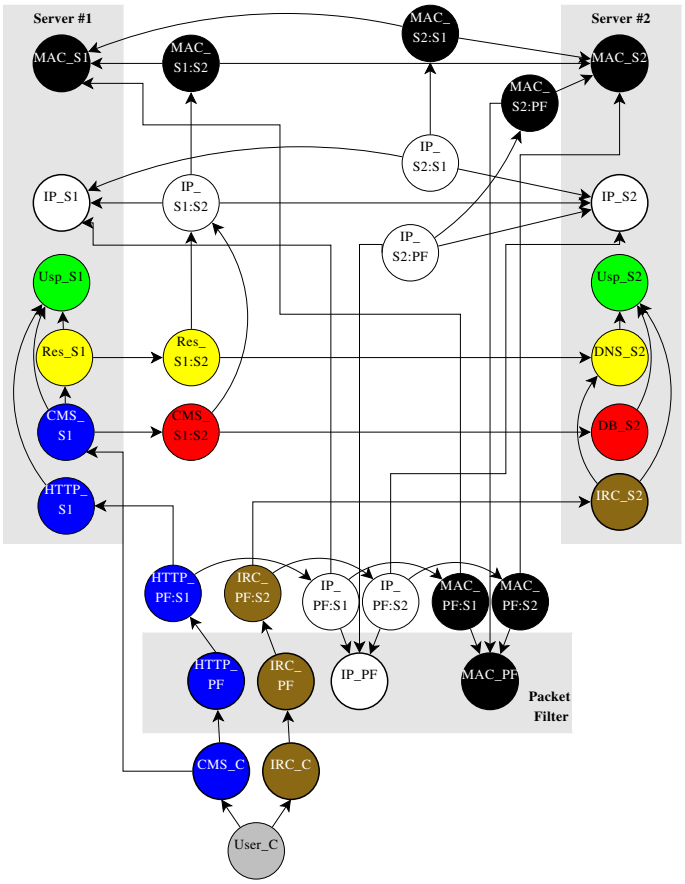

Fig. 2. Webshop-based evaluation scenario (33 of 45 vertices; resources for response implementation are not shown).

of

$$A(r) := 1 - \frac{d(r) - d_{min}(r)}{d_{max}(r) - d_{min}(r)} \tag{2}$$

For measuring the availability values, automated probing procedures were established, according to a representative usage of the resources. For the network services (HTTP, IRC, DNS, SQL), the success rate and delay for responding to a service request were measured to determine the availability, using e. g. `wget` and `netcat`. Similar procedures were established for the IP stacks, the CMS, and the userspace-based operating system resources. Using these measurement procedures it was possible to determine the transaction delay-based availability values for each of the resources in the validation scenario.

B. Dependency Determination

For each resource pair r, s so that r depends on s (denoted as $r \triangleright s$), the *input parameter* $A(s)$ was lowered gradually. This was achieved by delaying protocol packets for network accessible resources, and by a CPU/memory/filesystem/ process scheduler stressing benchmark program for userspace-based resources of the operating system. These tools were parametrized with respect to the delay and the relative success rate of the corresponding transactions. The effects of this reduction were measured as *output parameter* $A(r)$. An example

of observed parameters and the resulting linear dependency is given in Fig. 3.

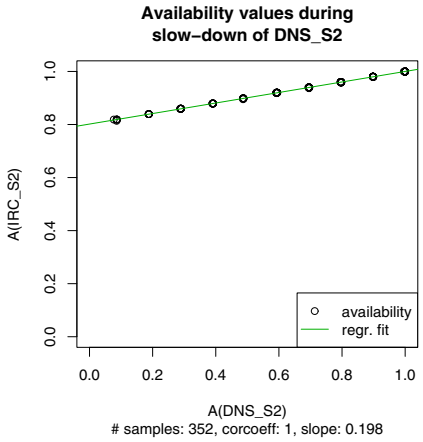

Fig. 3. Example for availabilities of resources that depend on each other.

As suggested by our results, many relationships between availability values of resources that directly depend on each other are (partially) linear. For successful transactions, it is possible to determine the slope and the y-axis offset of each weight function $w_{r,s}$ using linear regression (using least square fit). Additionally, the dependency functions D_r as given by multiple concurrent dependencies were determined.

C. In-Vitro Results

The functions D_r and $w_{r,s}$ were then used to predict the output parameters, in an initial analysis under controlled conditions. In the scenario from Fig. 2, the relevant output parameter is the availability of the user vertex *User_C*. Assuming that the customer depends on the CMS and the IRC to an equal extent, $A(User\_C)$ is given as the average of the service client availabilities, i. e. of the web browser *CMS_C* and the chat client *IRC_C*.

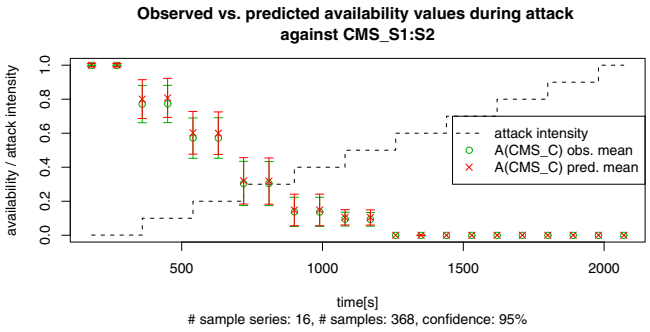

Fig. 4. Example for predicted vs. measured availabilities of the user vertex under an in-vitro DoS attack against the CMS.

The concordance of the predicted and measured values is notable when both are determined using identical input values, as underlined by the example from Fig. 4, where the aspired attack intensities against the connection of the CMS on S1 to the SQL server on S2 are depicted as dashed or dotted line, respectively. An attack intensity of 10 % is associated with an enforced average transaction delay of 10 % of the maximum value. The mean measured and predicted availability values of the CMS client application are shown as points in 'x' and 'o' shape, respectively, surrounded by their 95% confidence intervals.

D. In-Vivo Results

Simple generic attacks against TCP-based network services are SYN floods which aim at opening many different connections to the target and leaving them open without any further transaction. By exhausting the data structures for half-open TCP connections, legitimate connections may no longer be established. We assume a distributed SYN flood attack against the HTTP server from many different hosts in order to compare different responses to it. The impact of this attack at the user vertex is depicted in Fig. 5. Where the dashed line has a value of one, the flood is performed with full packet send rate. Due to the fact that almost any legitimate connection attempt fails, the overall availability drops down to approximately 0.5. Occasionally, a legitimate request is answered by the server.

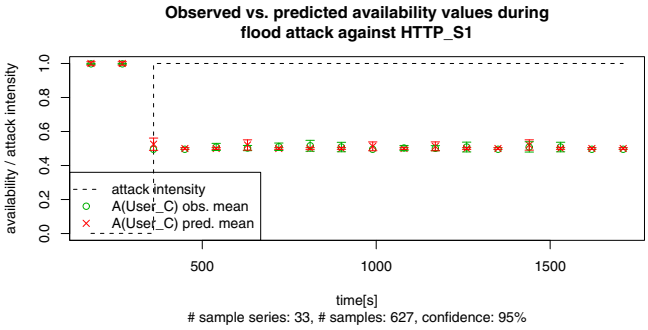

Fig. 5. Example for predicted vs. measured availabilities of the user vertex under an in-vivo DoS attack.

For our experiments, we selected 12 alternative response measures (for an excerpt, see Table I) to determine their metric values and to validate the graph-based parts by comparing them with the measured values. The comparably strong difference between predicted and measured succes in θ_3 is caused by a small variation of measured values before the response. Fig. 6 shows this comparison for one of these responses (θ_2, i. e. migrate HTTP server and CMS from S1 to S2). It comprises three separate steps, i. e. $\theta_2 = \theta_2^{(1)} \circ \theta_2^{(2)} \circ \theta_2^{(3)}$ with application durations depicted as vertical gray bars. The rightmost gray bar shows the step of reverting the response which is not part of the response itself.

During $\theta_2^{(1)}$, the CMS and the HTTP server on S2 are installed. Since the original server is still under attack, the availability does not change – neither concerning its average value nor its variance. Step $\theta_2^{(2)}$ transfers the necessary data from S1 to S2 and configures the new components accordingly. $\theta_2^{(3)}$ finally starts the new server and changes the DNS entry of the HTTP server host, so that the new service will be

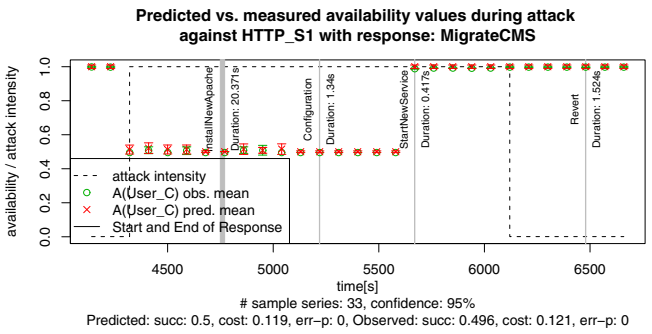

Fig. 6. Example for predicted vs. measured availability values of the user vertex under an in-vivo DoS response.

available to the customers after their DNS caches are expired and contain the new IP address of the server.

As visible from Fig. 6, the approach predicts a fully restored network after migrating the HTTP server from S1 to S2, which is very intuitive. The resulting overall availability will be 1.0 and thus, the success value is $\delta_S(G, G') \approx 0.5$. The measured times for the response steps lead to a cost value of $\delta_C(G, G') = 0.1$. The resources for applying the response include a network management console on S2 as well as an SSH connection from S2 to S1 and root shells on S1 and S2 (not shown in Fig. 2). These were not affected by the attack and thus, the error-proneness value becomes $\delta_E(G, G') = 0.0$.

Unfortunately, if the flood attack was performed against the symbolic name of the HTTP server platform rather than against its IP address, it is expected to find its target after the DNS cache update and thus the overall availability will soon drop to zero again. As a result, the durability becomes $\delta_D(G, G') = 0.01$.

Sym.	Description	Modus	Success δ_S	Costs δ_C	Error-Proneness δ_E	Durability δ_D
θ_1	Shut down HTTP on host S1	pred. meas.	-0.013 -0.014	0.011 0.012	0.0 0.0	1.0
θ_2	Migrate services HTTP and CMS from host S1 to S2	pred. meas.	0.5 0.496	0.119 0.121	0.0 0.0	0.01
θ_3	Block user node access to HTTP on packet filter	pred. meas.	0.003 -0.029	0.003 0.003	0.0 0.0	1.0
θ_4	Block user node accesss to IRC	pred. meas.	-0.5 -0.483	0.005 0.005	0.0 0.0	1.0
...	...	...	...	...	...	...
θ_{12}	Do nothing (wait)	pred. meas.	0.0 0.0	0.0 0.0	0.0 0.0	1.0

TABLE I
SUMMARY OF THE EXAMPLE RESPONSE METRIC VALUES

The decision about which alternative response $\theta_1, \ldots, \theta_{12}$ should actually be applied depends on the importance of the different partial metrics. If only the success in terms of restored availability does count, θ_2 (migrating services) would be the right choice. θ_3 (blocking HTTP access for all customers) has high durability but does not lead to increased availability and is therefore an inappropriate reaction. Also θ_{12} (do nothing) could be a choice, since it provides durability, and there are no costs involved. As proposed in [5], according weight factors $w_C, w_E, w_S, w_D \in \mathbb{R}^+$ could be used to phrase the decision criterion according to the actual purpose of the network.

As a result of the concordance of the availability measurements and predictions (see Table I), our graph-based metrics *response success* and *error-proneness* have been evinced to be an appropriate method for quantifying the response effects. Since the *application costs* metric does depend on the intermediate availability loss, it is also considered validated. The *durability* metric is not derived from availability estimations, thus it provides the ground for further work.

During the experiments, also the limits of the approach have been revealed, such as the inability to express complex interactions of workload, processing capacities and stochastic processes (e. g., the effects of dropping a certain percentage of SYN packets at the packet filter in a randomized fashion). We are currently investigating an according extension of the model (see [4]).

IV. Conclusions

This contribution discussed the validation of a graph-based approach for estimating effects of response measures to detected denial-of-service attacks. This approach has a number of advantages, compared to the existing work in the area of automated intrusion response selection, including a large degree of intuitivity of the approach, its ability to be adapted for differently scaled and granulated scenarios as well as its relatively easy maintainability.

An experimental validation in a real world e-commerce scenario has shown that the estimation of the overall effects of response actions in terms of availability is accurate, as long as the graph transformation primitives corresponding to a response action are known. Thus, under these circumstances, the GrADAR approach provides a simple but yet powerful methodology for estimating the impact of different response alternatives before they are actually applied to a real-world system. Extensions for enhancing this process are under development.

References

[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] 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).

[3] 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).

[4] G. Klein, M. Jahnke and P. Martini. Enhancing Graph-based Automated DoS Attack Response. To be published in: International Conference on Cyber War, Cooperative Cyber Defence Centre of Excellence (CCD-CoE), Tallinn, Estonia (2009).

[5] M. Jahnke, G. Klein, J. Tölle, and P. Martini. Protecting Military Networks with GrADAR – Graph-based Automated DoS Attack Response. To be published in: Military Communication Conference (MCC09), Prague Czech Republic (2009).