Last-Minute Paper Submissions, Forgotten Passwords and Greylisting – An Interesting Dilemma, and How To Solve It

Philipp Reinecke
Humboldt-Universität zu Berlin
Unter den Linden 6
10099 Berlin, Germany
preineck@informatik.hu-berlin.de

Katinka Wolter
Humboldt-Universität zu Berlin
Unter den Linden 6
10099 Berlin, Germany
wolter@informatik.hu-berlin.de

***Abstract*—Internet services often use e-mail messages to interact with the user. On the other hand, nowadays many mail administrators employ Greylisting [1], i.e. artificial delays of incoming mail, in an attempt to combat spam. While the latter is a noble undertaking, it may also lead to unacceptable delays in the transmission of important mail traffic. Problems of this kind, where the user cannot control the characteristics of the system itself, can be addressed by restart of the failed (or delayed) action. In this extended abstract we consider the application of the restart method to reduce greylisting-related delays in transmitting mails. We perform a small case-study and identify open problems. Furthermore, we point out future research directions in the application of the restart method in service-oriented systems.**

I. Introduction

Many systems in today's Internet utilise the mail system to interact with the user, and consequently user satisfaction with these services relies on timely e-mail transmissions. In general, mail transmissions take only a few minutes, and thus users have come to expect low transmission delays as the default behaviour. Unfortunately, most of the traffic that mail servers have to cope with consists of unsolicited messages, alias spam mails. In order to reduce load, mail server administrators have begun implementing greylisting [1] for incoming mail. With, greylisting the mail server initially rejects mail messages from unknown senders with a reply code indicating a temporary failure. This forces the sender to retry delivery at a later date, when it will be accepted (see Section II-A for details). In effect, greylisting causes artificial delays on mail message transmissions.

The mail system does not give any guarantees for transmission times. However, as mentioned above, users have come to expect low delays, and often even depend on them. We will illustrate the scenario with an example that may appear familiar to some readers (Figure 1): The user wants to send a paper to a prestigious conference. After working feverishly in order to make the deadline, all that is now left to do is to submit the paper using the on-line conference management system. With this system, the user has to log in on a web site and can then upload the paper. Unfortunately, the user has forgotten the password, as it has been some time since the last submission. He requests the system to send him the password by mail, and then anxiously waits for that one mail, as the deadline looms ever closer.

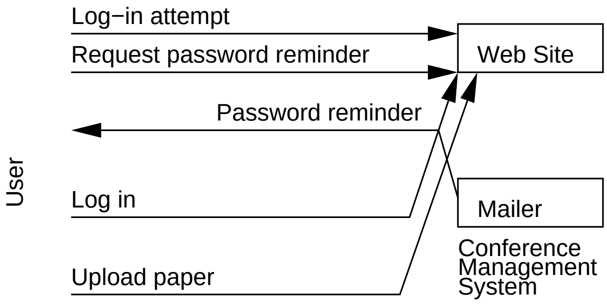

Fig. 1. Paper submission scenario

In situations such as these, where we do not know the cause of the delay, it is natural to restart the delayed (or failed) attempt. In this extended abstract we consider greylisting mail servers as one particular instance to apply restart. We perform a case-study (Section III), and then extend our discussion to encompass future problems in studying the restart method.

II. Scenario

The mail system is involved in our scenario as follows: When the user requests a password reminder, the conference management system generates a message and transfers it to its local mail server for delivery. The mail server looks up the server of the recipient and establishes an SMTP connection to this server, over which it then transmits the mail [2].

A. Greylisting

With greylisting, things differ from the ideal behaviour just sketched: Here, the recipient's mail server only accepts the mail immediately if the sender is white-listed. Otherwise, it rejects the mail with a reply code indicating a temporary failure. It then stores a tuple consisting of the sender and receiver mail addresses and the sending mail server's IP address into an internal database. Using this tuple the receiving mail server can identify subsequent transmission attempts by the same sender. After a customisable period of time (e.g. 10 minutes) the sender is automatically added to the white-list. As the sending mail server retries delivery, one of the repeated attempts will eventually be accepted. In our example, at this point the user will finally be able to log in and submit the paper. Within the receiving mail server, auto-whitelisted entries are periodically flushed if there has been no corresponding mail message for some time (e.g. 24 hours).

Greylisting reduces the amount of unsolicited mail (i.e. spam) that the mail server has to deal with, since most spam distributors do not attempt delivery upon receiving a failure code, even if the failure was marked temporary [1]. On the other hand, it leads to delays in mail transmissions. The delay depends on the constant time until a tuple is auto-whitelisted, and on the time the sending mail server waits before retrying the failed attempt. The latter time is influenced by the settings of server parameters, and also often by the server load, because servers may give priority to new messages. A common effect is that some messages are delayed for several hours, as we will illustrate in Section III.

B. Models

We study this scenario in terms of a user demanding service from a system. The service in question is the delivery of mail messages. By requesting a password reminder, the user requests message delivery. The service task is completed once the user receives the reminder. The user may restart the task by requesting another mail. We want to know whether restart is applicable, and if so, how long one should wait for delivery before restarting.

Greylisting delays depend on the receiving and the sending server. We assume that the user receives all mail through the same server and that greylisting delays are independent of the (constant) time until a sender tuple is white-listed in the receiver. We therefore distinguish three models based on how they describe the sender. Model $\mathcal{M}_1$ describes response times of the mail system as a single random variable X that reflects the behaviour of all mail servers. With this model, the restarted transmission may be performed by a new server. Our second model, $\mathcal{M}_2$, describes each sending server i by its own random variable X_i. Restarts are always answered by the same server. The third model ($\mathcal{M}_3$) describes the choice of a server i by an initial probability α_i. As with $\mathcal{M}_2$, restart is served by the same server as the original request, but we allow for a choice between servers for the original request. Thus, in contrast to the second model, $\mathcal{M}_3$ again describes the whole mail system, not just a single server.

III. Case-Study

We performed a case-study on a real-world mail server. While the ideal spot to collect the required data would be on the mail server itself, this is often not feasible due to technical and organisational constraints. For this reason, we obtained the data from the X-Greylist header in the mails in our in-boxes.

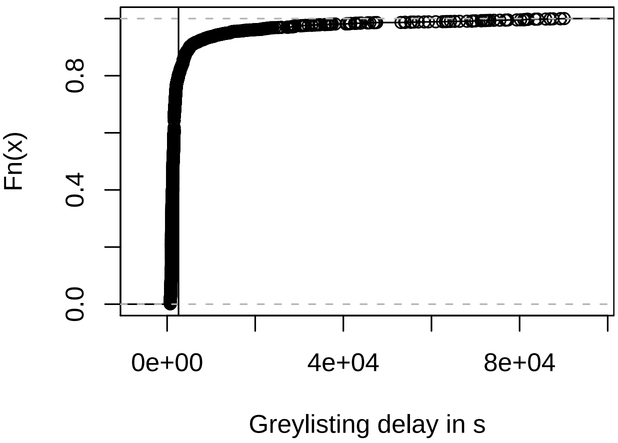

Fig. 2. CDF for the complete data set.

The X-Greylist header is inserted by the mail server when the message is accepted for delivery. If the mail was subjected to greylisting, the header contains the time that the message was delayed and a time-stamp indicating the time at which the header was inserted. For the purposes of this study we are only interested in the effects of greylisting and therefore assume that the greylisting delay reflects the response time of the mail system. A more thorough study would have to measure end-to-end transmission times of mail messages; unfortunately, such a study is difficult to conduct, since it requires that the clocks used for time-stamp generation in the sender and receiver be synchronised.

Prior to analysis we eliminated extreme outliers (e.g. delays of 61 hours) if they did not appear plausible based on the time-stamps in the headers. This left us with 3692 samples from the period between 24 October 2007 and 11 June 2009. We employ the R statistics package [3] throughout the analysis.

Our analysis is guided by the three models proposed in the previous section. Starting with the first model, $\mathcal{M}_1$, we analyse the data set as a whole. In the first line of Table I we note that the minimal delay inserted by greylisting is 12 minutes, while the maximum is around 25 hours. Furthermore, as the squared coefficient of variation $c^2 = 6.6$ indicates, response times have high variability. This is corroborated by the distribution of response times, shown in Figure 2: Most delays are short, but there is a long tail of large delays. For this model, the optimal restart timeout, computed with the algorithm presented in [8], is 2592 s (43 min), indicated in Figure 2.

For the second model we need to describe the behaviour of single mail servers. In our data set there are samples from 1151 individual senders. However, the vast majority of these sent very few messages: There are only 19 senders with more than 30 samples, and only four with more than 100 samples. We limit our analysis to the latter four (senders S_{1147}–S_{1150}), for which we have 104, 109, 120 and 253 samples, respectively. Their statistical properties (Table I) show varying behaviour: S_{1147} and S_{1148} have a very low coefficient of variation, while S_{1150} has high variability. S_{1149} has a lower c^2 than

	Messages	Quantiles					Mean	SD	c^2
		0%	50%	95%	99%	100%			
$\mathcal{M}_1$	3292	720	1353	14057	64414	90131	3784.3	9754.1	6.6
$\mathcal{M}_2^{1147}$	104	746	1600	1986	3556	3601	1603.7	337.9	0.04
$\mathcal{M}_2^{1148}$	109	969	1512	3633	7549	18783	1937.4	1857.3	0.92
$\mathcal{M}_2^{1149}$	120	870	4093	23212	81343	89325	7914.6	13907.9	3.09
$\mathcal{M}_2^{1150}$	253	723	1440	8983	66710	86920	3381.2	10088.6	8.90
$\mathcal{M}_3$ – Clustering according to means									
$\alpha = 0.0078$	11	47282	60962	83596	86288	86961	64496.9	13781.9	0.05
$\alpha = 0.064$	718	728	3024	28934	81366	90131	7111.7	13687.9	3.7
$\alpha = 0.024$	90	766	12520	70988	84479	87593	17625.8	20848.4	1.4
$\alpha = 0.896$	2820	720	1281	3559	8349	86920	1745.2	3439.5	3.9
$\alpha = 0.0087$	52	857	33121	71583	73020	73262	31621.5	22162.6	0.49

TABLE I
PROPERTIES OF THE DATA. ALL TIMES ARE GIVEN IN SECONDS.

S_{1150}, but, considering the high median and 95% quantile as well as the mean, appears to give longer greylisting delays than either of the other three. Optimal timeouts for these four senders are 3629 s (1:00 h), 5011 s (1:23 h), 11232 s (3:07 h), and 2592 s (43 min), respectively.

Our third model consists of initial probabilities α_i and random variables X_i describing the response time for the respective sender. As we have seen in the previous paragraph, the low number of samples for most senders makes it difficult to derive X_i. Therefore, in order to obtain $\mathcal{M}_3$, we first form n sender clusters based on the similarity of the sender's response time distributions. The size of the ith sender cluster is denoted by m_i. We then split the whole data set into n data sets D_i such that D_i contains all samples from senders belonging to cluster i. We derive X_i from D_i and estimate initial probabilities by $\alpha_i = {m_i}/{\sum_{j=1}^{n} m_j}$

We use Lloyd's form of the k-means clustering algorithm, as implemented in R, and set $n = 5$. We use the mean of the samples for each sender to characterise the sender's response time distribution.

The results are shown in Table I. We observe that our means-based clustering results in two data sets with $c^2 < 1$ and three data sets with $c^2 > 1$. Optimal restart timeouts for senders from the five clusters are 87091 s (24:11 h), 5875 s (1:37 h), 2074 s (34 min), 5184 s (1:26 h), and 1728 s (28 min).

IV. OPEN PROBLEMS WITH RESTART FOR GREYLISTING AND BEYOND

There are several problems with applying the restart method in our scenario. First and foremost, we have to decide whether the method can be applied at all. In particular, we need to know whether there is high variability in the delays and whether there is correlation in the delays of subsequent messages.

With respect to variability, we note that the squared coefficient of variation $c^2 = 6.6$ for the first model indicates high variability, while with the second and the third model we observe cases with high and low variability. These observations show that the effective delay due to greylisting is not constant, and that it may depend on the sending server. This supports our assumption (Section II-B) that the greylisting delay is not dominated by the receiver's constant greylisting setting.

Correlation, on the other hand, is difficult to study, as we do not yet have sufficient amounts of data. Such data must give the delays of many messages sent in short succession by the same sender; unfortunately, we cannot obtain such measurements by passive observation, since it is rather unusual for a sender to transmit messages in this manner.

After determining whether restart can be beneficial, we have to decide where the method is to be applied. That is, should the sender send another reminder after some time, or should the user request the password again? While the latter has the obvious disadvantage of putting additional strain on the service, it requires much less implementation effort than the former (in the simplest case, the user may just do it manually).

Then, the restart timeout needs to be set. There are several known algorithms to compute the restart timeout (see e.g. the taxonomy in [4]), and we have used the algorithm from [8], but it is not yet clear whether these are applicable to the kind of completion times encountered in our scenario. One particular problem affects adaptive restart algorithms that base their timeout computations on measurements from previous invocations. In the case-study we already observed that there is very little data available for most of the senders. This renders it difficult to compute an optimal timeout for single senders (model $\mathcal{M}_2$). We have attempted to circumvent the problem by clustering the senders based on their characteristics (model $\mathcal{M}_3$), but this is still work in progress.

Admittedly, restart for greylisting mail servers is a very special problem and might not be the most important or rewarding. However, it serves to illustrate similar problems that need to be addressed in the application of restart in service-oriented systems: First, a thorough understanding of the response times in SOAs is required, and such a model relies on data obtained from systems in realistic operating conditions. Unfortunately, very little data and models for such systems are available. We address this scarcity in an ongoing effort to extend the work presented in [4], [5], [6], [7].

Second, we need to decide at which level restart should be applied. In general, restart is performed by the client, however, in a service-oriented system services may themselves be clients to other services. Consequently, it may be beneficial not to limit restart to the user side, but also perform it in the services that interact with the user. Here, restart may be applied where the service acts as a client to other services as well as in the communication with the user (i.e. from the server side).

Third, it needs to be studied whether the available algorithms are optimal for the kind of response times encountered in a service-oriented system. Response times might be correlated, to which the restart algorithm must be adapted. Furthermore, the problem of scarce data available to an on-line algorithm becomes more pressing the more interactions with different services take place and the more complex service behaviour becomes.

V. Conclusion and Future Work

In this extended abstract we studied the application of the restart method to reduce greylisting delays. We performed a case-study on a real mail server and computed values for the optimal restart timeout.

Another important contribution of our work, however, is the identification of open problems that transcend the greylisting example. In particular, the problem of scarce data available to an on-line timeout computation algorithm affects restart in SOA systems, in general. In future work we will study application of a clustering algorithm, as proposed here, to solve this problem.

Acknowledgements

This work was supported by DFG grant Wo 898/2-1.

References

[1] E. Harris. The next step in the spam control war: Greylisting. http://www.greylisting.org/articles/whitepaper.shtml, August 2003. (Link validated 15 June 2009.).

[2] B. Krishnamurthy and J. Rexford. *Web Protocols and Practice*. Addison Wesley, 2001.

[3] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[4] P. Reinecke, A. P. A. van Moorsel, and K. Wolter. The Fast and the Fair: A Fault-Injection-Driven Comparison of Restart Oracles for Reliable Web Services. In *QEST '06: Proceedings of the 3rd International Conference on the Quantitative Evaluation of Systems*, pages 375–384, Washington, DC, USA, 2006. IEEE Computer Society.

[5] P. Reinecke, S. Wittkowski, and K. Wolter. Response-Time Measurements using the Sun Java Adventure Builder. In *Proceedings of the 1st International Workshop on the Quality of Service-Oriented Software Systems (QUASOSS)*, 2009. (to appear).

[6] P. Reinecke and K. Wolter. Phase-type approximations for message transmission times in web services reliable messaging. In S. Kounev, I. Gorton, and K. Sachs, editors, *Performance Evaluation – Metrics, Models and Benchmarks*, volume 5119 of *Lecture Notes in Computer Science*, pages 191–207. Springer, June 2008.

[7] P. Reinecke and K. Wolter. Towards a multi-level fault-injection test-bed for service-oriented architectures: Requirements for parameterisation. In *SRDS Workshop on Sharing Field Data and Experiment Measurements on Resilience of Distributed Computing Systems*, Naples, Italy, 2008. AMBER.

[8] A. P. A. van Moorsel and K. Wolter. Analysis of Restart Mechanisms in Software Systems. *IEEE Transactions on Software Engineering*, 32(8), August 2006.