

INFORMATION SOCIETY TECHNOLOGIES (IST)
PROGRAMME



Large Scale Monitoring of BroadBand Internet Infrastructure
Contract No. 004336

DRAFT
D1.1b: End-Users's Feedback on Anonymization

Abstract:

This document describes the feedback on anonymization provided by potential users of the Lobster infrastructure. The feedback was gathered by means of a comprehensive web-based survey. The main aim of the survey was to find out what kind of anonymization functions are desired by the end users and, hence, should be supported by Lobster.

Contractual Date of Delivery	M9
Actual Date of Delivery	M14
Deliverable Security Class	Public

The LOBSTER Consortium consists of:

FORTH	Coordinator	Greece
VU	Principal Contractor	The Netherlands
CESNET	Principal Contractor	Czech Republic
UNINETT	Principal Contractor	Norway
ENDACE	Principal Contractor	United Kingdom
ALCATEL	Principal Contractor	France
FORTHnet	Principal Contractor	Greece
TNO	Principal Contractor	The Netherlands
TERENA	Principal Contractor	The Netherlands

Contents

1	Introduction	7
2	Overview of the evaluation	11
2.1	Compliance to requirements	11
2.2	Feedback on the design	14
3	Evaluation results	17
3.1	Compliance with requirements	17
3.2	Feedback on the design	21
4	Summary	31

CONTENTS

List of Figures

3.1	Sort of access currently given to third parties	18
3.2	Conditions for giving 3rd parties access to traffic	19
3.3	The level of anonymization used	20
3.4	The form of field anonymization desired	20
3.5	Willingness to share with own organisation	21
3.6	Willingness to share with selected parties	22
3.7	Willingness to share with monitoring consortium	22
3.8	Willingness to share with third parties on reciprocal basis	23
3.9	Willingness to share with random third parties	23
3.10	Conditions for participating in LOBSTER	24

LIST OF FIGURES

Chapter 1

Introduction

The main goal of LOBSTER is to deploy an advanced pilot European infrastructure for accurate Internet traffic monitoring. In order to support collaborative passive network monitoring across a large number of geographically distributed—and possibly heterogeneous—sensors, LOBSTER is based on a distributed uniform access platform, which provides a common interface for applications to interact with the distributed monitoring sensors.

This platform is realized mainly by building onto the Monitoring Application Programming Interface (MAPI) [5, 6], which was developed within the SCAMPI project¹. Based on a generalized network flow abstraction, MAPI is flexible enough to capture emerging application needs, and expressive enough to allow the system to exploit specialized monitoring hardware, wherever available. In LOBSTER, MAPI is extended with *remote* monitoring functionality, allowing applications to interact with distant sensors across the Internet. Furthermore, the Distributed MAPI (DiMAPI) introduces the notion of the network flow *scope*, which enables the manipulation of compound network flows that may consist of traffic captured at several geographically distributed monitoring sensors.

An essential goal of Lobster is to address the concern for the sensitivity of the data that makes network operators, be they public bodies or private organisations, reluctant to share network traces. These concerns include issues to do with privacy and competitive advantage. For instance, giving third parties the power to trace webtraffic to individual users is unacceptable to most, if not all, organisations. This is true *a fortiori* for sensitive data content such as credit card details, medical records, etc. Less intuitive is the sensitivity of network statistics (such as average load, loss ratio, etc.). In practice, however, these turn out to be well-guarded secrets for commercial organisations, such as ISPs, as they represent commercial value. For instance, if an equipment vendor notices that an ISP's load is close to capacity, it may use that in negotiating a price for equipment. Similarly, if it becomes known that the load in an ISP's network over the past year has *decreased*, it may give a signal to the ISP's competitors that it is in trouble.

¹<http://www.ist-scampi.org/>

In summary, no organisation will give unlimited access to its network traces to just any other party. In fact, organisations want to exercise tight control over who gets access to which information. For this purpose, we developed the LOBSTER anonymization framework. It aims to provide users with (i) the means to specify policies about access to information about traffic, (ii) enforcements of those policies, and (iii) an extensive and flexible set of tools to anonymize data. The anonymization framework that underlies these aspects, is geared to allow users to mangle their data specifically for an intended party, so that, for instance, users from within the organisation are given much more access to the raw data than users from outside. The granularity at which the policy may be specified is quite fine, offering a large amount of flexibility.

The design of the anonymization framework was based on the input provided by the user community (see deliverable D0.2). By means of a questionnaire, LOBSTER endeavoured to find out what anonymization framework was desired by the potential users of the framework. Unsurprisingly, the wish list varied, and it became obvious that the framework needs to be sufficiently flexible to accommodate the various wishes.

Guided by the information gathered from the user community, we developed the anonymization with flexibility, tight control and ease of deployment in mind. The result was detailed in deliverable D1.1a. Next we took this design back to the user community asking for feedback. We tried to obtain feedback from a cross-section of the potential community: users that had filled in the previous questionnaire, administrators of the networks of the partners involved in the project, and users that had not been contacted by LOBSTER before.

As might have been expected, we did not receive many responses to our request for feedback. First, potential users that had already filled in the first questionnaire on the whole were not inclined to invest a lot of effort again, just to confirm that the design (that was based on their requirements to begin with), really met these requirements. Secondly, providing feedback on a concrete design requires a significant amount of effort, because the design document has to be read, thought about, and evaluated. Despite circulating several requests for feedback to a large group of principals (the entire group contacted for the first questionnaire, plus the consortium, plus principals who saw this for the first time), the response was quite disappointing. If a reason for not responding was given, it often stated that although the principal was interested in the project, there was no time to provide accurate feedback. Nevertheless we did manage to obtain a small set of responses, and these will be summarised in this deliverable.

The total number of principals contacted was approximately one hundred. Despite several reminders the amount of direct feedback remained very small. In order to see how well we satisfied the users requirements, we therefore took the answers to the original requirements questionnaire and tried to interpret them as feedback. In other words, if a user required X , we evaluated how well we support X . While we are aware of the circular relation in the design/evaluation process (evaluating a framework with a set of requirements that were used to design the framework

to begin with), it still proved to be a useful exercise and at least provides some indirect feedback in the absence of more immediate comments.

So the deliverable is based on two datasets. First, we match the anonymization framework against the specific feedback provided in response to our first questionnaire (i.e., the questionnaire that was sent out to establish the requirements for the anonymization framework). Second, we discuss the feedback provided in response to our second questionnaire. We emphasise again that the amount of feedback was quite limited and there is no point in performing hard statistics on such small datasets. Rather, if most of the feedback is in agreement, we intend to view the feedback either as loose confirmation or denial of our claim that the LOBSTER anonymization framework is a good approach that will benefit the stakeholders.

The remainder of this deliverable is organised as follows. In Section 2 we provide details about the questionnaire and the responses. In Section 3 we discuss the results and in Section 4, we summarise and draw conclusions.

CHAPTER 1. INTRODUCTION

Chapter 2

Overview of the evaluation

In this chapter, we outline the two questionnaires, the population that was targeted and the responses. In the next chapter, we will discuss the results and see how well our anonymization framework matches the target audience's expectations. The input for this deliverable was derived from two sources and we will discuss them in chronological order. In the first questionnaire we tried to establish the requirements for LOBSTER (strict requirements as well as a possible wish list). Some of the questions pertained to explicitly to the anonymization framework and it these questions that we have used for this deliverable to try and determine how well we have fulfilled the wishes of the respondents. The second questionnaire asked for immediate feedback on our architecture for anonymization. The goal of this questionnaire is largely to confirm or deny that the architecture is useful and of interest to the stakeholders.

2.1 Compliance to requirements

A detailed questionnaire was circulated among 65 stakeholders in the field. The number of external respondents was 24. The questionnaire concerned many aspects of LOBSTER, but for the purpose of this deliverable we will restrict ourselves to questions that immediately relate to what the stakeholders desire from an anonymization framework. The questionnaire was set up as an online multiple choice survey, that for most questions allowed respondents to add additional feedback in plain text. Briefly, we will consider the following issues:

1. **What sort of access are you currently giving to third parties?** Our aim was to find out how much sharing is already taking place and what sort of anonymization is being used to enable this. As organisations may have different relationships with different parties, it is possible that multiple different types of sharing and anonymization is taking place at the same time. Respondents were asked to pick zero or more of the following options, or to provide additional options in plain text:

- (a) Full traces;
- (b) Anonymized traces or headers;
- (c) Flow records (e.g., netflow);
- (d) Only high level aggregate information;
- (e) Only state of the network;
- (f) None whatsoever.

The results of question (1) can be found in Figure 3.1.

2. **Under what conditions will you give external parties access to your traffic traces?** For the purpose of this document, we were interested in two things: (i) whether there was any interested in a system that allowed for sharing of anonymized information, and (ii) how important anonymization and access control are in this. Respondents were asked to pick zero or more of the following options, or to provide additional options in plain text:

- (a) Trustworthy anonymization;
- (b) Membership in a consortium;
- (c) Partners selected by us;
- (d) Not asked at all.

The results of question (2) can be found in Figure 3.2.

3. **What level of anonymization do you use?** In case the respondents did not use anonymization, they were asked to list their preferred anonymization method. The main goal of this question was to get an initial estimation of the need for flexibility. Respondents were asked to pick zero or more of the following options, or to provide additional options in plain text:

- (a) Drop payload
- (b) Hash of payload;
- (c) Zero IP addresses;
- (d) Scramble IP addresses (e.g., hash);
- (e) Zero TCP/UDP ports;
- (f) Scramble TCP/UDP ports (e.g., hash).

The results of question (3) can be found in Figure 3.3.

4. **With what *form* of field anonymization would you be comfortable?** The idea behind the question is that regardless of *which* fields one wants to anonymize, there may be different methods of *how* to perform the anonymization (e.g., zeroing, hashing, encryption, etc.). Some of these methods are reversable, others are not. Respondents were asked to pick zero or more of the following options, or to provide additional options in plain text:

- (a) Zero the field;
- (b) Replace by one-way function (hashing);
- (c) Encrypt with strong encryption.

The results of question (4) can be found in Figure 3.4.

5. **Which information would you be willing to share?** The same question was asked five times: (1) for use within the same organisation, (2) for use by selected partners, (3) for use within a monitoring consortium (such as LOBSTER), (4) for use by third parties on a reciprocal basis, (5) for non-reciprocal use by arbitrary third parties. For the purpose of this document, we are mainly interested in two things: (i) is there a difference between the amount of access respondents would grant different classes of users (as this would suggest that there is indeed a need for specifying anonymization policies targetted at different groups, and (2) what sort of information the respondents would want to divulge, regardless of the users (a wide variation suggests a need for a flexible solution).

In addition, we should be able to support at least all the policies that are desired by the respondents, and possibly more. Respondents were asked to pick zero or more of the following options, or to provide additional options in plain text:

- (a) Full trace;
- (b) Full headers;
- (c) Drop payload;
- (d) Hash payload;
- (e) Encrypted payload;
- (f) Zero IP addresses;
- (g) Scramble IP addresses (irreversible hash);
- (h) Scramble IP addresses (strong encryption);
- (i) Zero TCP/UDP ports;
- (j) Scramble TCP/UDP ports (irreversible hash);
- (k) Scramble TCP/UDP ports (strong encryption);
- (l) Aggregates (e.g., utilization).

The results of question (5) can be found in Figures 3.5-3.9.

6. **How important is the flexibility of the anonymization framework?** This question was asked in a manner that allowed us to compare the importance of different issues. We are mainly after three pieces of information: (i) would stakeholders be willing to share information, and (ii) if so, how important

is it that one is able to control which party gets to see what information, and (iii) how important is it that one is able to apply different forms of anonymization? The exact question was as follows:

“In order to protect privacy of network users, Lobster develops a flexible anonymization framework (SiSaL), which allows network operator to choose which parties should be given access to what data. Lobster aims to perform a single baseline anonymization on the network card (in hardware), while additional anonymization levels can be specified. Indicate under what circumstances you would want to participate.”

Respondents were asked to pick zero or more of the following options, or to provide additional options in plain text:

- (a) None whatsoever;
- (b) If anonymization is done on the card;
- (c) If the common access platform resides on a remote host;
- (d) If the common access platform is in a 'sandboxed' environment (e.g., Java);
- (e) If we get to pick parties that see my data;
- (f) If we get to decide sort of anonymization for any party.;
- (g) If access is done on reciprocal basis;
- (h) If nothing is published about traffic in our network

The results of question (6) can be found in Figure 3.10.

2.2 Feedback on the design

After designing the anonymization framework on the basis of requirements collection, we also attempted to obtain feedback on the anonymization framework itself. For this reason, we circulated a second questionnaire together with a description of our architecture (as outlined in deliverable D1.1a).

As we expected that it might be a fair amount of work to provide the feedback, we tried to make the threshold as low as possible. One of the things people complained about regarding the first questionnaire was its length and the fact that it was not really possible to collaborate on the response, e.g., via email. So, rather than the long questionnaire that was circulated in the first round, we just distributed five fairly intuitive questions. The questions did have subquestions, but overall were designed to be fairly simple to answer. This was confirmed by the respondents. The questionnaire was initially circulated among the same 65 principals to which we added 10 new principals that were considered to be stakeholders. Later, we extended the size of the target population, by using individual contacts to approach people and ask them to provide feedback (for instance, we were present on the Sentinels Security Day in the Netherlands to request the participants in the networking

session to comment on the framework). So our target population consisted of the original respondents from the first questionnaire, plus ten new principals, plus an unknown number that was approached via personal contacts, plus some of the consortium administrators. The return was very poor. Despite several reminders, only 9 stakeholders completed the questionnaire and returned the results to LOBSTER.

All questions are related to the description of the architecture in deliverable D1.1a:

1. Overall, do you feel this anonymization framework would be useful
 - (a) to you?
 - (b) to a community of network administrators (e.g., ISPs, NRENs, universities)If not, why not?
2. Regarding the "Anonymization Architecture" (Chapter 3), please provide feedback on the following issues:
 - (a) is there any aspect of anonymization that is not catered to in this architecture? If so, which?
 - (b) is the approach with admission control for enforcing the appropriate anonymization sufficiently powerful? Are the credentials a good approach? Would you have confidence in this method? What do you think is lacking?
 - (c) do you feel the cooking/uncooking and anonymization of TCP streams is useful in practice? If so, when would it be useful and when would it not be useful? If not, why not?
[Note: cooking/uncooking is intended for offline anonymization of traces rather than online anonymization at wirespeed.]
 - (d) is function reordering a useful approach? Do you have comments/suggestions about it?
 - (e) do you have any suggestion for improving the anonymization architecture?
3. Regarding the definition of anonymization policies (Chapter 4), please provide feedback on the following issues:
 - (a) do you think that using policies for specifying anonymization is useful? If not, why not?
 - (b) do you think that the policy engine is sufficiently expressive to cater to all real-life anonymization needs? If not, why not?
 - (c) do you think the tool for defining the policy (see Section 4.1 in the attached document) is useful? What features are missing?

CHAPTER 2. OVERVIEW OF THE EVALUATION

- (d) do you have any suggestion for improving the policy definition?
- 4. Regarding the protocol field names (Appendix A), are there any protocols/fields that you would like to see included (other than the ones that are already listed)?
- 5. Regarding the "Complete List of the Anonymization Functions" (Appendix B), do you feel there are functions missing? If so which? (Also if you know of any other approaches that are not listed in Chapter 2, "State-of-the-art", we would love to hear about them).

Chapter 3

Evaluation results

In this chapter, we analyse the data provided by the respondents of the two questionnaires. As we provide the results, we will also derive some overall observations regarding the meaning of the results. For instance, if many respondents indicated in various questions that they require different types of anonymization, we observe that there is a strong desire for such a feature. In addition, we will match both the requirements and the respondents' comments against our framework to see how well we have done.

3.1 Compliance with requirements

As indicated earlier, the results of question (1) can be found in Figure 3.1, the results of question (2) can be found in Figure 3.2, the results of question (3) can be found in Figure 3.3, the results of question (4) can be found in Figure 3.4, the results of question (5) can be found in Figures 3.5-3.9 (for each of the different parties referred), the results of question (6) can be found in Figure 3.10.

As the number of respondents was only small (24), we refrain from applying statistics to the results. Rather, we want to limit ourselves to more cautious statements. For instance, if a majority of the respondents indicate that they want to use feature X , we take this to mean that there is a demand for X and if we want to cater to these stakeholders, we should support X . We then interpret this as feedback on our framework by evaluating how well we support feature X .

Figure 3.1 shows that there is already a fair amount of sharing going on. Indeed 12 of the respondents indicated that they share full traces. However, the results show that many different forms of sharing is taking place. We interpret this as a user requirement. Indeed, Figures 3.3-3.10 all confirm that different forms of sharing is considered of the great importance.

Enabling administrators safely to share network data with various parties is the main design goal of the LOBSTER project. The approach we have taken in the LOBSTER architecture is to permit remote users to collect data (statistics and/or packets) from remote sites, even if these sites are administered by different or-

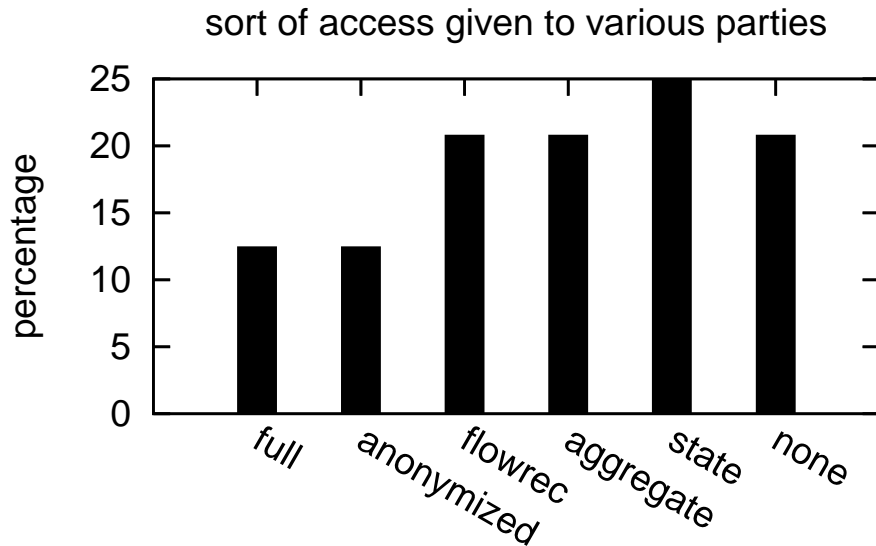


Figure 3.1: Sort of access currently given to third parties

ganisations. To prevent undesirable information leakage, the access to the sites is strictly controlled by trust policies. For instance, perhaps our friends may be allowed to see TCP/IP headers with prefix-preserving anonymization applied to the IP addresses, while random third parties may only see headers with zero-ed IP addresses (i.e., experiencing stricter anonymization). The policies can be easily specified, e.g., using the tool provided by the LOBSTER consortium and will be automatically enforced. In other words, it is impossible to circumvent the anonymization rules. We should note that we do not check the sanity of anonymization rules. This is the responsibility of the administrators.

One of the main advantages of LOBSTER is that we are able to build directly on top of an existing monitoring architecture, known as SCAMPI. The monitoring API (MAPI) developed in SCAMPI allows different functions to be executed on network packets and streams. The function will be executed at the most appropriate place, i.e., on the NIC if possible, in user space if not.

The LOBSTER design has different features that allow sharing of data. First, a MAPI application can access data from a set of remote sensors. In this case, the sensors are approached very much like a local sensor in the SCAMPI project. Not only may the sensors reside in remote nodes, they may also be accessed as a group (known as a *scope* in LOBSTER). Second, the design allows for remote execution to cater to those applications that do not scale if all results are processed at a central site (e.g., applications that require saving a lot of packets). Communication with remote sites is protected by encryption, authentication, and authorisation.

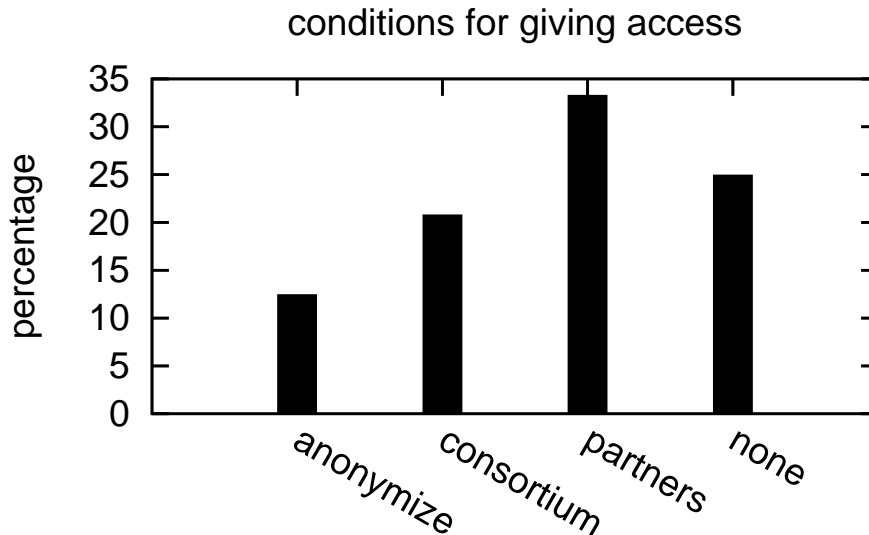


Figure 3.2: Conditions for giving 3rd parties access to traffic

All in all, we believe we cater well to this requirement. In addition, the design is sufficiently open to allow more functions to be added as needed.

The need for the ability to select what partners get access to what data is indicated by Figures 3.2 which shows that the ability to select the partners with whom to share data scores rather high. It is also indicated by Figures 3.5-3.9 which show that respondents want to be able to apply different policies to different user groups, and by Figure 3.10 which shows that many respondents considered the ability to decide who gets access to what data very important.

Again, this requirement has become the heart of the LOBSTER anonymization framework. The specification of policies allows administrators to determine at a fine level of granularity which restrictions hold for which users. Credentials are tried way of reducing the management overhead with respect to fine-grained access control. Indeed, we opted for an implementation that builds on a somewhat road-tested authentication and authorisation system as LOBSTER's `authd` daemon is based on similar component that was used in such projects as the Open Kernel Environment [2] and FFPF [1].

As many different forms of field anonymization are desired by the respondents, LOBSTER has to support all anonymization features that are listed, but also allow for more. It seems that there is a very real demand for different anonymization policies.

Considering the fairly small number of respondents (24), we will not apply advanced statistics on these results and limit ourselves to the observation that feed-

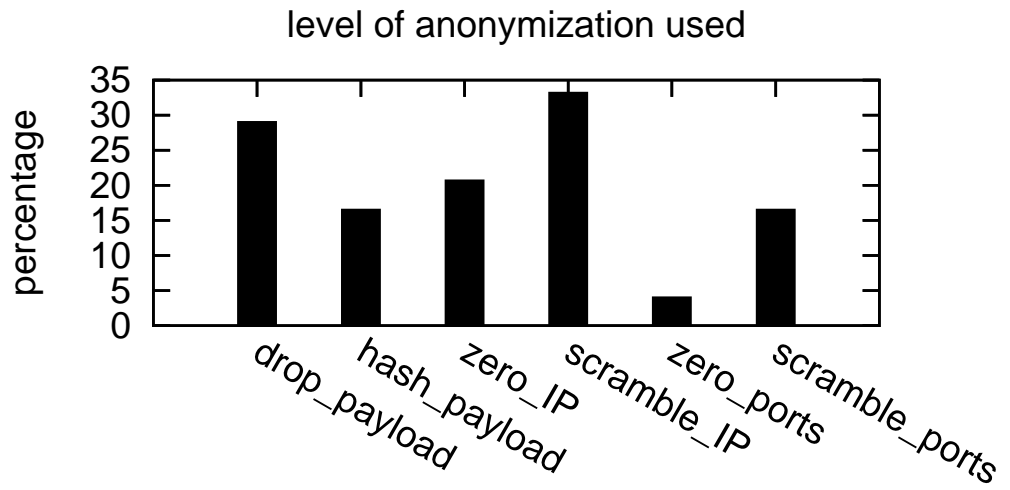


Figure 3.3: The level of anonymization used

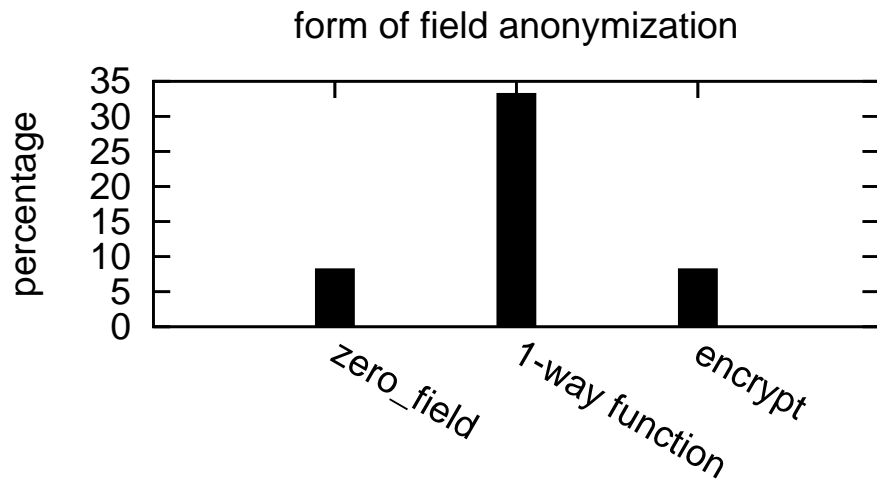


Figure 3.4: The form of field anonymization desired

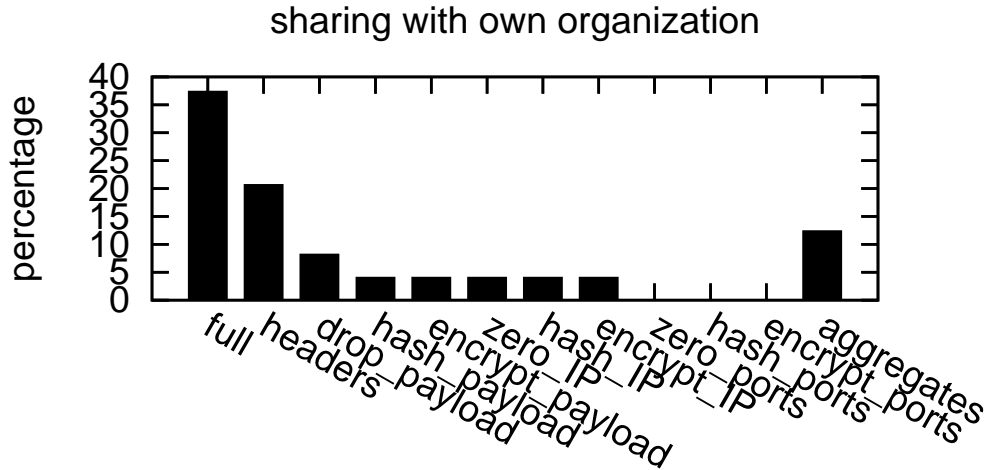


Figure 3.5: Willingness to share with own organisation

back shows that there is demand for different types of anonymization and for diversification of anonymization policies for different parties. We have checked and confirmed that all feedback on the desired features of the anonymization framework that was provided by the first questionnaire has been incorporated in the framework.

3.2 Feedback on the design

The immediate feedback on the proposed design is based on answers to open questions. Overall the feedback was quite positive and for this reason need not be repeated in detailed this section. Indeed, one of the new respondents indicated a desire to join the LOBSTER infrastructure. In the remainder of this section, all praise has been omitted from this document, except where it is particularly relevant (e.g., when both praise and criticism was voiced about the same thing).

Another respondent commented in great detail on deliverable D1.1a on which the questionnaire was based, not just about the content, but also highlighting such things as spelling and grammar issues, and commenting on the presentation in general. We consider such feedback extremely useful, but for the purpose of this deliverable we limited ourselves to feedback on the content.

We will now highlight per (sub-)question what criticism was voiced by the respondents. Where appropriate we will incorporate responses by the consortium inline. Some issues do not have a response yet, which indicates that they are under

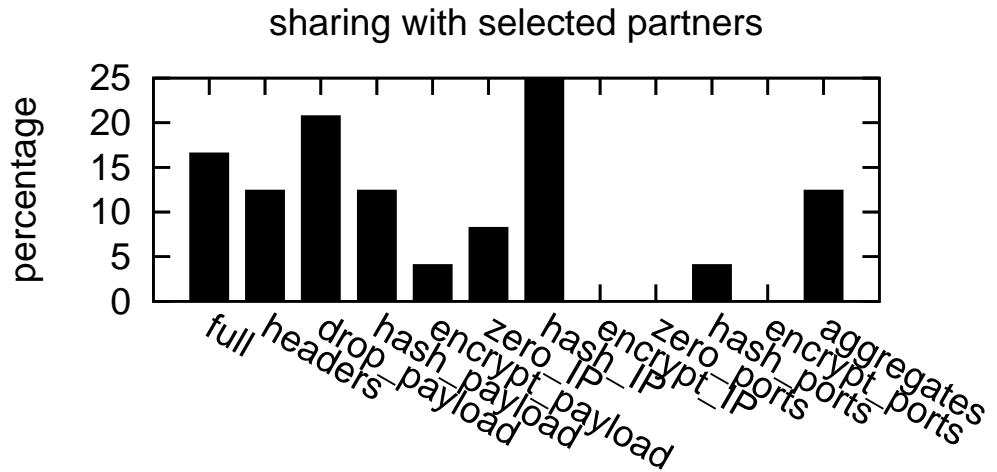


Figure 3.6: Willingness to share with selected parties

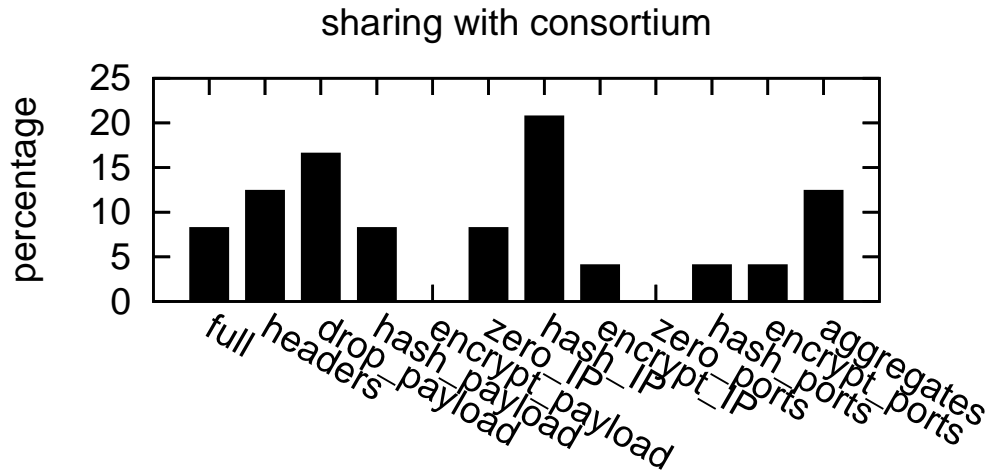


Figure 3.7: Willingness to share with monitoring consortium

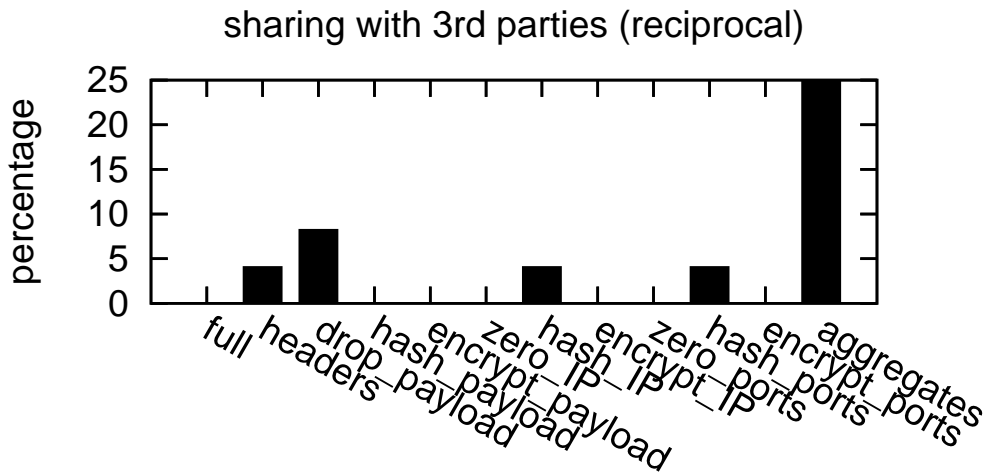


Figure 3.8: Willingness to share with third parties on reciprocal basis

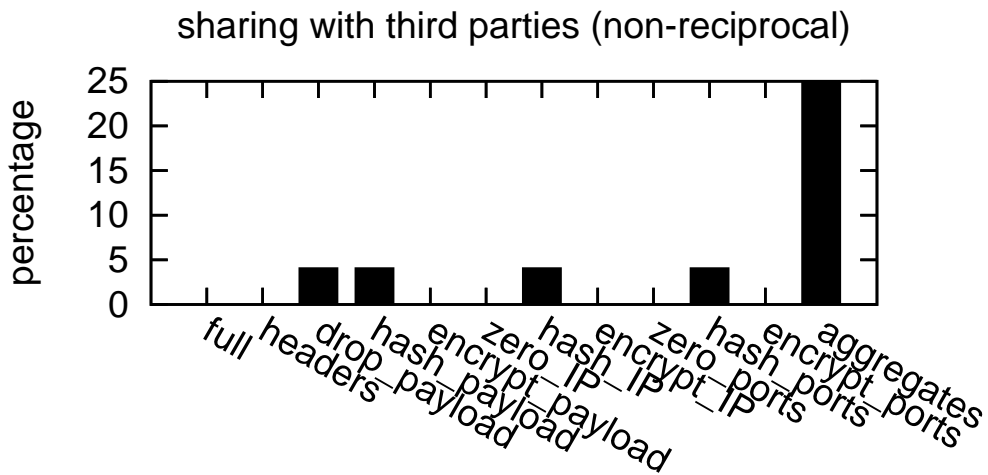


Figure 3.9: Willingness to share with random third parties

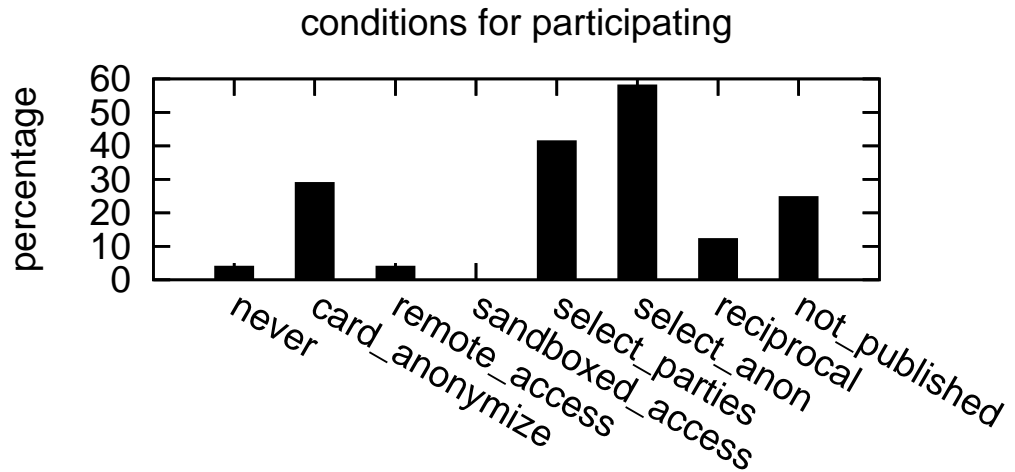


Figure 3.10: Conditions for participating in LOBSTER

discussion. As mentioned earlier, all questions are related to the description of the architecture in deliverable D1.1a:

1. Overall, do you feel this anonymization framework would be useful
 - (a) to you?
 - (b) to a community of network administrators (e.g., ISPs, NRENs, universities)

If not, why not?

→ **Feedback:**

All respondents confirmed that they considered the framework to be very useful.

2. Regarding the "Anonymization Architecture" (Chapter 3), please provide feedback on the following issues:

- (a) is there any aspect of anonymization that is not catered to in this architecture? If so, which?

→ **Feedback:**

Limited anonymization on reassembled TCP streams. One respondent remarked that anonymization on reassembled TCP streams was fairly limited:

“TCP stream anonymization as described in the deliverable is only catered to to a limited extent (although it is hard to see how it could be done differently). Perhaps the SISAL language (not described in the deliverable) could be applied to the streams and handle things generically? On the other hand, such reassembly will only be possible for low-speed connections or offline processing and as almost all network administrators will prevent access to payload altogether, this is not a major issue.”

Response. We agree with this observation and even though we also do not expect access to reconstructed TCP streams to be very common, it may be useful tool for administrators that want to supply realy network traces to their own employees. For this reason, we will add to the LOBSTER repertoire a SISAL-like language that works on streams rather than packets.

- (b) is the approach with admission control for enforcing the appropriate anonymization sufficiently powerful? Are the credentials a good approach? Would you have confidence in this method? What do you think is lacking?

→ **Feedback:**

Man-in-the-middle. Most respondents either liked the approach or did not comment on it. One was sceptical:

“Authentication does not look very robust, I am not convinced replay or man-in-the-middle attacks are not possible.”

Response. We will ask experts in the field of security to look at the authentication and modify it if it is considered unsafe. However, we do point out that the system is a direct descendant of a similar implementation in projects like the Open Kernel Environment [2] and FFPF [1]. For these projects the authentication/authorisation mechanism was evaluated by experts in the field.

Unclearities. Another comment concerned some unclarity in the text, that again raised concerns about man-in-the-middle attacks.

“Where is the authorisation public key identification (anybody can generate a keypair). Also: who chooses the arbitrary integer? The daemon, right (otherwise: replay attack possible). All in all, I think you should leave out the details about authentication.”

Response. Again, we will ask security experts to look at the design again to verify its correctness.

Revocation. One respondent commented on the revocation issue:

“Much confidence, provided sysadmin learn to handle such certificates. However, how would revocation work?”

Response. Revocation is a problem that is well-known in capability-based systems. Since Lobster uses credentials that are not unlike capabilities, the same problem will occur in LOBSTER. Currently,

credentials are revoked simply by changing the policy. Given a convenient policy-making tool, this is not a very complex affair. We plan to investigate whether or not this simple form of revocation is sufficient for the purposes of LOBSTER. If needed, we will modify our design to facilitate revocation of rights.

Maintenance costs and scalability. It was suggested to address the scalability by means of virtual organisations:

“Admission control and credentials are sufficient for enforcing the appropriate anonymization. The described mechanism though will have high maintenance costs as the number of users increases. So, I believe it should be explained how can this mechanism can be used to implement virtual organizations. The use of VOs will push most of the administrative costs away from sensor administrators and will make deployment easier.”

Response. While the issue is not immediately urgent, this is a very interesting suggestion that will be explored in the remainder of this project.

- (c) do you feel the cooking/uncooking and anonymization of TCP streams is useful in practice? If so, when would it be useful and when would it not be useful? If not, why not?

[Note: cooking/uncooking is intended for offline anonymization of traces rather than online anonymization at wirespeed.]

→ **Feedback:**

Not very useful On respondent did not consider cooking to be important:

“Probably not too much useful. I can’t find any example when the data stream has to be both cooked and anonymized. Analysis of an unknown protocol or an attack should be done on full stream.”

Response. We think that it might help remove, for instance, URLs from HTTP traffic. As observed by another respondent:

“Yes. In fact, it may be too limited. The anonymization function can only scan for, e.g. URLs. In reality you may want to find and replace more.”

We are currently extending this so as to allow for more anonymization on TCP streams.

- (d) is function reordering a useful approach? Do you have comments/suggestions about it?

→ **Feedback:**

Doubts about reordering. Two respondents answered that they did not think reordering was very useful.

- “I am not sure about usefulness of the function reordering.”

- “It does not seem particularly useful considering its complexity.”

All other respondents commented favourable about reordering, e.g.: “Function reordering may be very useful, especially for a new user of the anonymization framework. Now, it is an optional component (can be enabled/disabled). Wouldn’t it be more useful if it was always enabled and an ”advanced user”, would just disable it , if desired?”

The function reordering was designed with a keen eye on scenarios where novice users may apply anonymization functions in the wrong order (for example fix the checksum and then do another modification, in which case the checksum will no longer be valid). Advanced users really know the order that function should be applied so no change should be made there.

Function reordering has two main goals. Firstly, to automatically detect common pitfalls in the anonymization functions applied, both in what anonymization functions are applied and in which order. Secondly, to ensure that the semantics of anonymization process are correct. This component is specific for anonymization functions and does not affect any other functions that user applies either before or after anonymization part. The function reordering is performed upon flow connection. It examines all functions in the flow and tries to find the first and the last ANONYMIZE function and to reorder, for instance, when the provided order does not make sense (e.g., anomysation between cooking and uncooking, checksum calculation prior to packet mangling).

It is clear that the reordering confuses people. We think that it might benefit experienced users and emphasize that it is already optional. To make sure novice users do not get confused we will disable the reordering by default.

See question 3

- (e) do you have any suggestion for improving the anonymization architecture?

→ **Feedback:**

Sisal for streams. There were various suggestions:

“A generalised SISAL (as was mentioned by some of the consortium members) with provable policies, so we can apply a set of rules and have 100 percent guarantee that we have not inadvertently allowed access in conflict with policy rules.”

Response. This is what we may look into in a separate project. As it is research it does not fit in the LOBSTER project.

3. Regarding the definition of anonymization policies (Chapter 4), please provide feedback on the following issues:

- (a) do you think that using policies for specifying anonymization is useful? If not, why not?

→ **Feedback:**

Reuse and graphics. All respondents liked this aspect, e.g.:

“It is very useful: An administrator who is responsible for more than one sensors, can easily create a policy once and apply it to many sensors. Graphic presentation: very good !!!It minimizes the risk of making mistakes and helps to the management of policies.”

- (b) do you think that the policy engine is sufficiently expressive to cater to all real-life anonymization needs? If not, why not?

- (c) do you think the tool for defining the policy (see Section 4.1 in the attached document) is useful? What features are missing?

→ **Feedback:**

Feature interaction. An issue came up with respect to the order of checking and reordering:

“Does reordering take place before or after checking credentials?”

Response. This will be defined more rigorously. The most likely answer is that it will be checked before reordering. **[Is this correct?]**

Reordering “It is hard to warn the user for function reordering, since it happens in the sensor. Perhaps the policy tool can be extended to warn the user for possible reorderings for the anonymization policy they specify.”

Response. Again, reordering may confuse people when defining policies. A simple solution is to stipulate simply that user actions should conform to the policy, irrespective of the reordering. This is also the scheme that is adopted by LOBSTER. **[Is this correct?]**

- (d) do you have any suggestion for improving the policy definition?

4. Regarding the protocol field names (Appendix A), are there any protocols/fields that you would like to see included (other than the ones that are already listed)?

→ **Feedback:**

SMTP, email We received various suggestions: “Perhaps support for SMTP protocol and fields should be added. While important HTTP traffic is usually transferred via HTTPS, SMTP traffic almost always is unencrypted since it relies on application level encryption. Application level encryption is not easy to setup for the average user and it has to be configured for both sender and recipient. So users usually prefer to send the password that a colleague needs with unencrypted email, rather than setting up the application level encryption for their email clients.

Apart from full SMTP support, email address stripping could be also useful against using traces for email harvesting. The 'clean' way to implement email address stripping would require cooking the packets and using a simple regex. But if cooking is not needed elsewhere in the anonymization, a 'hacked'-fast implementation with no cooking would be welcome."

"Probably all common protocols that are used for DoS attacks - how else to analyze an attack? (telnet, ssh, smtp, ...)"

Response. We will consider all of these and try to implement as many as possible, starting with the most common protocols.

5. Regarding the "Complete List of the Anonymization Functions" (Appendix B), do you feel there are functions missing? If so which? (Also if you know of any other approaches that are not listed in Chapter 2, "State-of-the-art", we would love to hear about them).

→ **Feedback:**

More RANDOM. functions. Some suggestions for functions:

"In the spirit of `FILENAME_RANDOM`, `EMAIL_RANDOM` could be added."

Response. We will consider these and try to implement as many as possible.

Related work. Some suggestions for related work:

"I hear there is a new project for anonymization by Paxson - have a look"

Response. We already know about various anonymization tools by Paxson's group, e.g. [4] and yes, we do look at all of these approaches. A new project in this group is known as `tcpmkpub` [3]¹. However, in our opinion no existing framework is quite as powerful as LOBSTER as it takes into account not just anonymization of traces, but also access policies.

¹<http://www.icir.org/enterprise-tracing/tcpmkpub.html>

CHAPTER 3. EVALUATION RESULTS

Chapter 4

Summary

We had two goals in this deliverable (i) to assess the need for anonymization, network data sharing and policy diversification, and (ii) to assess how well the anonymization framework designed in the LOBSTER project met the wishes and needs of the stakeholders. For this purpose, we sent out two surveys to a large target population world-wide consisting of potential users (research and education networks, ISPs, academic institutions, etc.). The first attempted to gather information about the wishes of the stakeholders. The second was a request to give direct feedback on the anonymization framework designed within the LOBSTER project.

Considering the fairly small number of respondents (24 for the first questionnaire and 9 for the second), we will not apply advanced statistics on these results. Instead, we limit ourselves to the observation that feedback from the first questionnaire shows that there is demand for different types of anonymization and for diversification of anonymization policies for different parties. We have checked and confirmed that all feedback on the desired features of the anonymization framework that was provided by the first questionnaire has been incorporated in the framework.

The second questionnaire contained direct responses to the anonymization framework that was designed by LOBSTER. By and large the feedback was very positive. Critical remarks were shown and commented on in Chapter 3.

CHAPTER 4. SUMMARY

Bibliography

- [1] Herbert Bos, Willem de Bruijn, Mihai Cristea, Trung Nguyen, and Georgios Portokalidis. FFPF: Fairly Fast Packet Filters. In *Proceedings of OSDI'04*, San Francisco, CA, December 2004.
- [2] Herbert Bos and Bart Samwel. Safe kernel programming in the OKE. In *Proceedings of OPE-NARCH'02*, New York, USA, June 2002.
- [3] Ruoming Pang, Mark Allman, Vern Paxson, and Jason Lee. The devil and packet trace anonymization. Paper under submission, <http://www.icir.org/enterprise-tracing/devil-submit.pdf>, 2005.
- [4] Ruoming Pang and Vern Paxson. A high-level programming environment for packet trace anonymization and transformation. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 339–351, New York, NY, USA, 2003. ACM Press.
- [5] Michalis Polychronakis, Kostas G. Anagnostakis, Evangelos P. Markatos, and Arne Øslebø. Design of an Application Programming Interface for IP Network Monitoring. In *Proceedings of the 9th IFIP/IEEE Network Operations and Management Symposium (NOMS'04)*, pages 483–496, April 2004.
- [6] The SCAMPI Consortium. MAPI Public Release. http://mapi.uninett.no/download/mapi_1.0b1.tgz.