PDCS No 100 15 June 1993

Data Collection for Security Fault Forecasting: Pilot Experiment

T. Olovsson*, E. Jonsson*, S. Brocklehurst#, B. Littlewood#
*Department of Computer Engineering, Chalmers University of Technology, S-412 96, Göteborg, Sweden
#Centre for Software Reliability, City University, Northampton Square, London ECIV OHB, UK

Abstract

In most contexts, it is not feasible to guarantee that a system is 100% secure. Measures and predictions of operational security of computer systems are therefore obviously of interest to any owner of a system which is a candidate for potential intruders. Such measures would allow assessment of current and future expected loss to the system owner due to security breaches in a given attacking environment and a given level of protection. In [Littlewood, Brocklehurst et al. 1991] a probabilistic approach to modelling operational security, analogous to that used in reliability, is suggested. It is clear that empirical data would be useful in deriving a plausible probabilistic approach to security modelling.

Such data can be acquired experimentally, by allowing a group of selected people to perform security attacks on a given computer system in a controlled way. The attack process can then be monitored and relevant data recorded. This document describes such an experiment. As far as we are aware, this is the first attempt to conduct such an experiment, and our intention was more to explore general feasibility than to collect data that provides significant information for modelling. This *pilot* experiment did indeed give some valuable information on how future full-scale experiments of this kind should be performed and the results and recommendations for improvements to the experimental set-up are discussed here.

Release number 2

Release status Final

Availability status Public

Available from Tomas Olovsson, Department of Computer Engineering, Chalmers University of Technology, S-412 96, Göteborg, Sweden.

This work was supported by the CEC ESPRIT programme under Basic Research Action Project 6362, PDCS2 (Predictably Dependable Computing Systems 2). We are also grateful for comments on the proposed experiment from Marc Dacier, Yves Deswarte and David Wright and also to the participants in the experiment.

Data Collection for Security Fault Forecasting: Pilot Experiment

Tomas Olovsson
Erland Jonsson
Department of Computer Engineering
Chalmers University of Technology
S-412 96 Göteborg
Sweden

Sarah Brocklehurst
Bev Littlewood
Centre for Software Reliability
City University
London EC1V 0HB
England

Abstract

In most contexts, it is not feasible to guarantee that a system is 100% secure. Measures and predictions of operational security of computer systems are therefore obviously of interest to any owner of a system which is a candidate for potential intruders. Such measures would allow assessment of current and future expected loss to the system owner due to security breaches in a given attacking environment and a given level of protection. In [Littlewood, Brocklehurst et al. 1991] a probabilistic approach to modelling operational security, analogous to that used in reliability, is suggested. It is clear that empirical data would be useful in deriving a plausible probabilistic approach to security modelling.

Such data can be acquired experimentally, by allowing a group of selected people to perform security attacks on a given computer system in a controlled way. The attack process can then be monitored and relevant data recorded. This document describes such an experiment. As far as we are aware, this is the first attempt to conduct such an experiment, and our intention was more to explore general feasibility than to collect data that provides significant information for modelling. This *pilot* experiment did indeed give some valuable information on how future full-scale experiments of this kind should be performed and the results and recommendations for improvements to the experimental set-up are discussed here.

Acknowledgement

This work was supported by the CEC ESPRIT programme under Basic Research Action Project 6362, PDCS2 (Predictably Dependable Computing Systems 2). We are also grateful for comments on the proposed experiment from Marc Dacier, Yves Deswarte and David Wright and also to the participants in the experiment.

1 Introduction

This experiment arises from some work carried out in an ESPRIT research project, PDCS (Predictably Dependable Computing Systems) [Littlewood, Brocklehurst et al. 1991]. The overall aim is to develop a quantitative theory of *operational security*, similar to that which now exists for reliability (for hardware and, more particularly recently, for software).

In reliability, the important point is that we express our confidence in a system probabilistically in terms that reflect naturally its ability to operate successfully. In particular, there is an acceptance that there is a suitable "time" variable, and it is assumed that failures in time occur in a random process (albeit perhaps a complex one). This natural randomness of the failure process arises in reliability for several reasons. In the case of software reliability, for example, we cannot be certain when the next failure will take place because of the natural unpredictability of the operational environment. The input space is typically very large and complex, and we never know with certainty which inputs will be selected in the future. Neither do we know which inputs will, when selected, trigger a design fault and so result in a failure. We take account of this natural uncertainty in probabilistic measures of reliability such as the *reliability function* (the probability of failure-free operation for a specified time t), the *rate of occurrence of failures*, the *mean time to next failure*, etc.

Computer security, on the other hand, has not so far been expressed quantitatively in terms that acknowledge a similar uncertainty and a dependence upon the nature of the operational environment. Instead, security *evaluation levels*, such as the classes and divisions of the Orange Book [NCSC 1985], are concerned with the nature of the development process, and of the design architecture. They are similar in some respects to the general arguments about the quality of the development process that preceded the availability of techniques for direct measurement of operational reliability. Yet security clearly *is* an operational concept. Intuitively, when we say that one system is more secure than another, we are asserting a belief that it is less likely to exhibit a violation of the security policy than the other - for example, it is less likely to be breached by a particular deliberate attack. Although a high security level may tend to give higher "operational security" in this sense, it cannot be guaranteed, just as the use of good development practices cannot be guaranteed to result in high reliability. Using such development and design practices in security does not give us what is really of interest, i.e., a *measure* of operational security.

The purpose of the pilot experiment described here was to begin to examine the possibilities of collecting data that could be used to derive such a measure. A group of selected people were allowed to attack a chosen computer system in a controlled way. We monitored and recorded the attack and breach processes and collected information on the relevant variables discussed below.

It is important to note that the aims of this initial experiment were fairly modest. Firstly, the scope of our investigations was narrowed by considering only a limited number of the theoretical questions addressed in [Littlewood, Brocklehurst et al. 1991]. Secondly, due to practical considerations associated with using a real system, certain of the attacking behaviour had to be controlled, possibly at the expense of realism of the behaviour of the attackers. Thirdly, the quality of the collected data also depends on how accurately and extensively the participants in the experiment reported on their actions. In view of this, it was anticipated that the experiment may not

give us the data we need to answer even the limited questions that we decided to address. However, we did anticipate that conducting this pilot experiment may teach us how to construct more realistic and comprehensive experiments in the future. Our intention here was somewhat tentative, which is the reason for calling the experiment a pilot: this seems to be the first attempt to carry out such a study, and we are aware of some important difficulties. Other rather similar experiments have been conducted, for example, [Allwood, Bjorkhag 1990] where a group of people were asked to find programming errors in their own code, but our approach differs too much to benefit from these methods. Some security systems are tested by using tiger teams to attack the system, but here the objective is more to identify vulnerabilities and assess the corresponding risks, than to do testing representative of operational use and collect data for predicting operational security. If our pilot experiment is successful in demonstrating feasibility, we intend to conduct more experiments based on experience learnt from this pilot.

2 Background

Our overall aim in this work is to provide the means whereby the operational security of a system can be measured. In thinking about this problem, the example of reliability was foremost in our minds. Reliability theory can be thought of as the description of processes involving *failure events* taking place in *time*. In the case of security, a naïve view would be that we could similarly model the process of *security breaches* in time as well. However, it quickly became clear that, although there were parallels, there were some important distinctions.

2.1 Effort

In the case of security, time alone does not seem to be an appropriate variable. In particular, elapsed time when a system is not under attack is irrelevant for the evaluation of resistance to deliberate attack. Instead, we need to consider a variable which captures the *effort* expended in attacking the system and informally we expect a system which requires more effort to be expended until it is successfully breached to be "more secure". We might then define operational security in terms analogous to reliability: the *security function* would be the probability of surviving without a security breach for the expenditure of a given effort by an attacker; other measures such as the rate of security breaches per unit effort, mean effort to the next security breach, etc, have obvious meanings.

Clearly, different systems will vary in their ability to resist a particular expenditure of effort from an attacker. Factors influencing this resistance include how the system is configured, the presence of security enhancing mechanisms, the quality of the system design and how the system is operated. The idea is that this "ability to resist attack" can be estimated by means of measuring the effort expended in order to achieve a breach. However, effort is a complex variable composed of several factors: we believe that *education, skill* and *experience* of the attacker, and *time, money* and other resources spent by the attacker, are the most important factors. Any differences between the ability of different attackers to breach the system should be captured in the effort variable. The important point here is that the effort variable should capture the intuitive notion that the more effort is invested in attacking the system, the greater is the chance of achieving a breach. An important issue in the pilot experiment is whether it is *possible* to devise an effort measure with these properties.

2.2 Breaches and Rewards

Security is often divided into three aspects: confidentiality, integrity and availability. All three aspects are normally possible targets for attackers and can be subject to security breaches [Olovsson, Jonsson, 1992].

The classical notion of a security breach as a security 'failure event', is for our purposes sometimes too simplistic. Firstly, it is clear that the value of different security breaches to a single attacker may vary enormously[1], and this notion of *reward* should be incorporated into the measures. Secondly, whilst an attacker of a system may certainly gain something of value from these breach *events*, they may also acquire reward continuously - for example by gradually learning about the system. For these reasons we believe that, instead of the failure events in time that characterise the reliability process, the most general description of the security process would be in terms of a *reward process*, involving both discrete and continuous increments, against *effort expended*.

In our experiments we want to collect empirical data on breach events and rewards in order to learn about the nature of the reward process in security. There are various practical limitations to what we could ever expect to achieve here. Firstly, although it would be desirable to include all types of breach events, some cannot be allowed in a controlled experiment where a real system is used due to the possible consequences for other users of the system. Secondly, it will clearly be difficult to get quantitative information about rewards.

The reward an attacker would get from breaking into a system determines his/her motivation and affects whether he/she is willing to spend the effort that is needed in order to perform the attack. Examples of rewards are personal satisfaction, gain of money, revenge or simply pure curiosity, but it is also possible that the attacker may get negative rewards such as those due to the consequences of detection. The system owner *may* only attach reward (actually loss) to attacker activity which is illegal, whilst the attacker may attach some reward also to legal activity (for example activity that allows the attacker to learn something about the system that would be useful in other, illegal, attacks). Such considerations about the reward processes suggests that each attacker has a subjective view of his/her rewards which may be different from other attackers' views in similar circumstances and also different from the system owner's view of his rewards (the loss due to a breach event for a system owner is likely to be different from the reward to the attacker). Each attacker may be expected to apportion their effort optimally in some way according to their view of their (potential) rewards while the owner may be expected to respond to attacker activity in accordance with his view of his rewards (or losses).

2.3 Viewpoints

The previous discussion on breach events and rewards highlights the inevitability that substantially different views of the system will be taken by different people. Even though predictions of the breach events (and rewards/losses) from the owner viewpoint may be the most interesting, we wish to also examine the (possibly different) events from the attackers' subjective views. It is

---

[1] This is also true in reliability: the cost of different types of failure can vary a great deal. However, not all reliability models take account of this.

important to take account of these different views since they will be the motivation for all attacker and owner behaviour. It is clear that each will have an incomplete view of *activity* in the system if only due to the limitations of their own monitoring. In addition, they will have different *beliefs* arising from their past different experiences. These changing views in the light of changing experience will motivate their behaviour and thus cause complex feedback.

In our experiments we wish to measure as much as is practical of not only activity and events, but also of the participants' views of these; we wish to collect as much information as possible on breach events and rewards from the different viewpoints. We shall therefore require participants to report on their own activity, observations, interpretations and motivation, as well as conducting our own monitoring.

2.4 Questions to be Answered

We hope that this and following experiments will give answers, or at least partial answers, to several questions regarding how to measure and model operational security. For our pilot study we limited the modelling issues which we wanted to address to the following (in increasing order of difficulty):

- How can we derive our single quantified measure of effort expended by an attacker, unifying the different factors (time, expertise, etc)?
- What is the distribution of the effort that needs to be expended to achieve a breach, for a single attacker?
- What is the nature of the more general stochastic process of reward from the attackers' and owner's views (i.e. when rewards can be continuously accrued as well as being associated with discrete breach events).

Further modelling issues, which are outside the scope of the pilot study, but which we would like to consider in future experiments are discussed in [Littlewood, Brocklehurst et al. 1991]: for example, effects of collaboration between attackers on their ability to breach.

In addition to the issues related to modelling (i.e., *security evaluation*) there are likely to be some interesting observations that we can make from such experiments which relate to concerns of *security achievement*:

- How many and what types of security breaches were (or could have been) detected by the system owner? How many and what types of breaches could have been detected if we added non-standard hardware or software?
- What kind of security breaches occur: confidentiality and/or integrity issues?
- Are only well-known vulnerabilities in the system exploited by our attackers?
- How many of the attacks and breaches could have been carried out by non-legal users?

Note that this list of security concerns is by no means meant to be complete. It was anticipated that from this pilot experiment we may learn additional information which is relevant to security achievement issues.

3 Experimental Set-up

3.1 The Target System

For our pilot study an existing system, which was available and in use, was chosen to be our target system. This was a normal Unix system and it was possible to use this system in our pilot without carrying out any modifications to it. The selected system had the following specifications:

- A fully operative system was used, which consisted of a set of 22 SUN ELC diskless work-stations running SunOS 4.1.2, using one shared file-server. These systems were in regular use by other "normal users".
- The system was configured as recommended by the vendor, with network access and active users. This configuration increased the chances of our attackers achieving breaches although, even with this configuration, there was a substantial risk that no breaches would be achieved by the attackers in our pilot experiment.
- The system had all the standard monitoring and accounting features enabled (see appendix E), to facilitate experimental monitoring.
- The attackers had physical access to the work-stations themselves but not to the file-server.

One method which could be used to increase the quantity of data obtained from such experiments would be to deliberately implant vulnerabilities in our target system, since this is likely to make it easier for the attackers to achieve breaches. No vulnerabilities were deliberately implanted in this case since, for a number of reasons, this may result in unrealistic data. This possibility may be considered in future experiments.

For future experiments, it may also be possible to use several differently configured systems, in order to compare and to draw conclusions as to why certain systems are more vulnerable than others. This may also give an indication of how robust (over different systems) the effort variable is for modelling operational security.

3.2 The Attackers

For this pilot we wanted to limit the number of attackers to about 10-15 persons. Due to the obvious difficulty of incorporating information about expertise of attackers into the effort variable, we tried to use attackers with similar expertise to facilitate the evaluation of the results. For future experiments we may investigate security in the presence of a more diverse attacking population and with a larger number of attackers.

The attackers were final year students working towards a M.Sc. in computer engineering and we only selected students who have a specific interest in computer systems and who work with computers in their spare time, e.g., students who are members of the university computer club. Initially we were only interested in having normal users (i.e., not professional attackers) to break into a normal system working with a modest level of security. By selecting students who were considered to have more expertise in computers than regular students, we hoped to maximise the chance of success of the pilot experiment, in the sense that we see enough breaches (and a diversity

of them) to get some useful empirical data. If this turns out to be the case, we shall consider using regular students, with less prior experience, in future experiments.

Each attacker was a legal user of the system with normal user privileges. As a legal user, the attackers had an increased number of ways in which they could attack the system and the chances of the attackers achieving breaches was increased.

3.3 The Owner

The owner of the selected target system which we used in our pilot experiment was the usual system administrator of the system. His usual monitoring is fairly modest and mostly consists of actions when the system is not operating normally, i.e., he only takes action when the system fails, for example after a hardware problem or when a user complains about system functionality. The file-server is searched on an irregular basis for unexpected system error messages, mostly in order to find potential problems, but these messages could also reveal illegal attacker activity. The other work-stations are normally not investigated at all, mostly because there are too many of them and an investigation would require too much work.

3.4 The Experimenter

The role of the experimenter was to monitor and coordinate all activities during the experiment. In particular he had to try to make sure that the attackers and the owner were complying with the *experimental rules*. These rules were devised to ensure the success of the experiment and are discussed below in sections 3.7 and 3.8. This means that, as much as possible, the experimenter had to be continually monitoring all activity throughout the experiment and always be available for consultation by the attackers and the owner. This was because, even if the attackers and owner could be trusted to obey the rules for the experiment, some of these rules were hard to define precisely and so guidance was required during the experiment.

The experimenter had to be very careful that within the limits of ensuring that the experimental rules were obeyed, the influence that he had on the attackers' and owner's behaviours was minimised as much as possible. Further, the experimenter had to document all his own actions and all the decisions he made.

It was important for the experimenter to collect as much information as was practical to try to get the data required to investigate the issues discussed previously in section 2. In particular, he had to try to ensure that the subjective beliefs and views of the participants were captured, as well as more factual information.

The success of the pilot experiment depended crucially upon accurate and complete reports being obtained from the attackers and the owner. It was important that the experimenter continually monitored reports from the attackers and owner so that he could immediately rectify inaccurate reporting during the experiment. Much of the important data (particularly data relating to times of activity and effort) would not be recoverable after the event by questioning the participants. The exact details of the reports that were used in the pilot experiment are discussed further in sections 3.9 and 3.10.

Automatic monitoring (see appendix E) allows the experimenter to monitor the activities on each user account and see how much time each attacker has spent on-line as well as how much CPU time he/she has spent during different stages in the process. This may serve as a useful check on the accuracy of the manual reporting by the attackers and the owner. It may also be used to investigate the efficiency of higher level monitoring (for security *achievement* concerns).

3.5 Brief and Motivation for the Attackers

The brief given to the attackers was achieved by giving them a presentation, together with some supporting documents (see appendices A, B.1 and C), just prior to the beginning of the experiment. The experimenter's briefing of the attackers before the experiment began (and during the experiment as unanticipated problems were encountered) was a crucial part of motivating them to behave as we wished them to do for the purposes of this experiment; there were some aspects of the experimental set-up which it was essential that they understood and others that we wanted to keep secret from them.

Firstly the attackers were given a general description of the overall objectives of the experiment in order to motivate them into taking part and fully cooperating. The main objective was to motivate them to behave, within the inevitable practical limitations, like real attackers. These aspects were covered in a document which was handed out to the attackers (shown in appendix A). Further, they were given a set of rules (see section 3.7 and appendix B.1) which they had to understand and motivated to obey. Accurate and complete reporting was one aspect covered in these rules and the report forms (see section 3.9 and appendix C) were also handed out and explained to the attackers. In addition to this, more technical information on the target system and relating to allowable attacking behaviour was given.

Various artificial incentive mechanisms were put into play in an attempt to encourage the attackers to behave realistically and to obey the experimental rules. These artificial incentives were explained to the attackers in the brief before the experiment began.

By *realistic behaviour* we mean that the attackers use their own subjective rewards to influence their attacking behaviour. The aim was to get the attackers to believe that final *prize(es)* would be given for how realistically they do this. A scheme was suggested where *points* would be given in accordance with each attacker's view of their achieved rewards. This points scheme was not devised before the experiment but the experimenter had to make the attackers believe that such a points scheme did exist, and that it accurately reflects real-life rewards. A general motivation for the points scheme was given in the briefing document in appendix A. It was planned that prizes would be awarded according to the number of points obtained.

Each attacker will be given a fixed *gratuity* for taking part in the experiment regardless of the points they achieve (i.e., regardless of whether they achieve effective breaches or not). This gratuity will not be given to any attacker who knowingly does not obey the experimental rules. In particular, no gratuity will be given if they knowingly cause unacceptable disruption to the system, if they collaborate or communicate with other attackers or if their reporting is inaccurate or incomplete.

Prizes and gratuities will be awarded after the experiment is over.

3.6 Brief and Motivation for the Owner

Firstly the owner was given a general understanding of the overall objectives of the experiment in order to motivate him to cooperate fully. This was achieved, in part, by giving the owner the supporting documentation that was given to the attackers (see appendices A and B). Further, he was given a set of rules (see section 3.8 and appendix B.3) which he was made to understand and motivated to obey. The main requirement of the system owner was that he behave, within the inevitable practical limitations, like a real system owner. Accurate and complete reporting was also required by the system owner and the information required from the system owner was explained to him (see section 3.10).

It was important to stress to the owner that an accurate account of any changes he would normally make to the system in response to attacking activity he observed must be given to the experimenter so that the experimenter can simulate the owner's real life behaviour with respect to such preventative measures that he would take.

Clearly the most worrying feature of the experiment, from the owner's point of view, was the risk that attackers performed activities that the owner found unacceptably disruptive. The owner had to distinguish between activity which was genuinely damaging or just potentially damaging and made to understand that we wished to allow the attackers to perform the latter and not the former. The experimenter needed to reassure the owner on these issues in order to dissuade the owner from making changes to the system to prevent possible disruptive activity.

The experimenter tried to encourage the owner to cooperate by promising feedback after the experiment on aspects of the security achievement issues which were learned. Using such feedback, the owner may be able to make his system more secure.

3.7 Rules for the attackers

The attackers had some well-defined rules to follow, and these rules were explained to the attackers in order to ensure the success of the experiment. The list of rules given to the attackers in the briefing session are shown in appendix B.1. In appendix B.2 we explain the reason for enforcing those rules for the attackers which are not self-explanatory.

3.8 Rules for the Owner

Rules (see appendix B.3) were also required for the system owner in order that the experiment was successful. In general it was just required that he behaved as he normally would in spite of the fact that he knew that the experiment was going on; there are some exceptions to this where the owner's behaviour had to be controlled. In appendix B.4, we explain the reason for enforcing those rules for the owner which are not self-explanatory.

3.9 Reporting for the Attackers

We wanted some measure of expertise in order that we could later take this into account when constructing our effort measure. Before the experiment began, the attackers documented their background (formal education, prior experience with computers and computer security etc.)

together with their interest and motivation for participating in the experiment. This was accomplished by the attackers completing an *attacker background report* (see appendix C.1). Of particular interest in this report was question 10 where the attackers were asked about information of previous attacks on systems that they have done. It is for this reason that the attackers needed to be assured of confidentiality since we wanted to encourage them to answer such questions honestly.

The remaining reports filled in by the attackers related to their attacking activity and the breaches that they achieved during the experiment. It was important that the reports were complete with respect to effort they expended (whether they were actually interacting with the system, or not), their subjective rewards (actual and expected) which included gain due to breaches and loss due to owner detection, and so on. Other particularly important data included the (real) times of occurrence of all activity and observations of the attackers; the experimenter may wish to superimpose different views so that he can connect, for example, observations and activity by the attackers with those by the owner. It was important that all reports were completed as soon after the event as possible, so that the data given was as complete and accurate as possible.

Note that we are interested not only in activity, but in beliefs and subjective rewards, and so we monitored to some extent the attackers' views throughout the experiment. That is, we asked them about their own activity, about what they observed and about what they thought (what their motivations were). We wanted to get a picture of the attackers' motivation for their attacking behaviour, for example for which breach (and reward) they were aiming at every stage and what they thought that breach would buy them in terms of reward (and perhaps future related breaches). Clearly, as an attacker learns more about the system, these views are likely to change and we wished to capture these changes. Since we were asking about beliefs which may change it was particularly important that they completed certain parts of the reports at particular stages during the attacking processes. We also wanted to find out what subjective rewards the attacker attached to each breach that he/she actually did achieve.

For each attack that they tried, each attacker had to complete an *attack report* (see appendix C.2). This report is divided into 3 sections, which had to be filled in just before the attack began, during the attack and just after the attack. In appendix D.1 we explain the reason for, and the background of, the questions in the attack report.

For each breach that the attacker achieved he/she was immediately required to fill in a *breach report* (see appendix C.3). Essential to the breach reporting was the time at which the breach occurred, and the attack report and activity within that attack with which it was associated, so that the breach events can be placed in the appropriate place within the effort expended by the attacker. (It may not always be the case that each breach event is the termination of an attack, so this was not assumed). In appendix D.2 we explain the reason for, and the background of, the questions in the breach report.

It is clear that these reports to be completed by the attackers were quite extensive. The experimenter needed to continually monitor the attackers' reports to make sure that these reports were complete, particularly with respect to the most important data required. Most of this data (particularly the time report data) is not recoverable after the event, even though the experimenter has the option of questioning the attackers afterwards.

3.10 Reporting for the Owner

Unlike the attackers, there were no reports for the system owner but we wanted to have continuous information from him on any attacking behaviour he observed together with any event which he observed and considered to be a breach. We also wanted a record from the owner of which activity he stopped, how he stopped it and why he stopped it. In addition we wanted a record of changes that he would have made to the system in order to stop attacker activity that he observed. What we required was something like a continual daily time report from the owner which includes all this information and when, in real time, all these events took place, so that we can relate the owner's observations and activity to the attackers' activity. Since the owner's usual monitoring and activity relating to security is fairly low level it was considered that such an activity report would be sufficient to get the data required. The experimenter could question the owner during the experiment if information from him was incomplete.

After the experiment we shall question the owner about his view on breaches made by attackers which were not detected by the owner. And finally, we may have some concept of significant events in the system which were overlooked by both the attackers and the system owner, i.e., that may be considered as a breach event to the owner (i.e., associated loss) when the occurrence of such events is pointed out afterwards by the experimenter.

3.11 Disposition of Time

In the pilot study, we initially planned that the attackers would need to spend at least 3 to 5 days of effective working time during a 2 to 4 week calendar period in order to get enough data. If, after the experiment began, it became apparent that this was too low an estimate of the required effort (both effective and calendar), it would have been possible to extend the duration of the experiment.

There was initially no planned deadline or exactly defined final date for completion of the experiment. Instead, the experimenter continually tried to motivate the attackers to continue to participate in the experiment. This was because we wanted to get an effective time-slice of real attacker behaviour: the presence of an end date could have caused the attackers to behave non-typically just prior to such a date since they could have decided, for example, to reallocate their efforts according to the short time available. It was decided that, when it became apparent that the attackers were no longer participating very actively in the experiment (i.e., they ceased putting much effort into attacking the system), the experimenter would declare that the experiment was over.

4 Results

4.1 Overview

The experiment commenced with a briefing for the attackers in the evening on February 4th 1993. As previously discussed, at that stage the attackers were given no deadline for the end of the experiment. After about 5 weeks, attacker activity was so low that, in order to try to motivate the attackers to become more active, it was decided that the attackers *would* be told that they had only 3 weeks left in which to participate. On March 5th the attackers were told that the experiment

would end on March 26th; this date was chosen so that it would not interfere with the examination period (15th-20th March) for the attackers, most of whom were regular undergraduate students at Chalmers. A few weeks after the end of the experiment, on May 3rd, a summing-up meeting was held, during which the results of the experiment were presented to the attackers and a prize was awarded to the most successful attacker. This also gave the experimenter the opportunity to get the attackers' opinions on what was good and bad about the experiment and their suggestions on how it could be improved. The attackers were asked to complete a final report at this meeting: an English translation of this report (originally in Swedish) is shown in appendix C.5.

The initial briefing was attended by 22 potential participants but it was clear, even at this early stage, that some of them would not have time to work on the project. It turned out that, out of those who had initially expressed an interest, only 13 participated in the experiment in the sense that they submitted at least one attack or breach report. The summing-up meeting was attended by only 6 attackers and, perhaps not surprisingly, these attackers had been the most productive during the experiment. As a consequence only a limited number of final attacker reports (appendix C.5) were collected, and unfortunately, this limits the value of this part of the investigation.

The automatic logging system recorded more than 73,000 commands and programs that were executed by the attackers (out of a total of about 800,000 programs that were executed on the system during this time). Most of these commands were executed automatically, i.e. by software guessing passwords, searching the system, etc. In theory some types of data, lost due to incomplete data reporting by the attackers, should be recoverable from this automatic log, but in practice this task is non-trivial. In addition to this, many commands were executed on other systems out of reach of our logging system.

4.2 Attack Methods Used and Breaches Achieved

4.2.1 Summary

As already mentioned attack and breach reports were received from 13 attackers, and these attackers sent 37 attack reports and 25 breach reports in total during the 7 weeks duration of the experiment. Almost half of these reports were submitted during the first days of the experiment.

When we studied the data from the attackers it was evident that the different attackers had different notions of what constituted single attacks and breaches. For example, some attackers regarded a successful single-user boot-up sequence (for details see below) *plus* the privilege transfer to the server as one attack and one breach. They accordingly filled in one attack and one breach report, while others filled in two separate attack and breach reports corresponding to these two activities and successes. In some cases attackers filled in only one attack report for a number of clearly unconnected attack methods which they tried. In addition to this there was attacker activity which was either informally reported to the experimenter and for which no reports were sent or activity which was not reported at all. There were some instances were the attacker sent part 1 of an attack report stating their intentions to try a particular attack method but sent no subsequent related reports so that we know neither whether they continued with this method (and the effort expended), nor whether they were successful in their attack. Therefore, due to these various reporting problems, the total *numbers* of attack and breach reports are not in themselves

particularly meaningful. These reporting issues are discussed in more detail below, in section 4.5, while the remainder of this section discusses details of the attack methods tried.

The majority (approximately two thirds) of the methods used in the attacks were "standard" methods, making use of vulnerabilities in the system which seem to be generally already known to the attackers prior to the beginning of the experiment, and/or using already existing software. The remaining attack methods tried were more innovative, but even here ideas had in some cases been taken from textbooks and other external resources had been used.

4.2.2 Standard Attack Methods

Broadly speaking there were 3 different types of standard attack methods: single-user boot-up (sometimes followed by using the SUID-facility to achieve root on the server), running a password cracking program and using an xkey snooping program [Richie], [Grampp, Morris 1984], [Curry 1990] and [Garfinkel, Spafford 1991].

The most common of these attack methods was to carry through a single-user boot-up sequence on a work-station to which the attacker had access; 9 attackers tried this and all of them were successful. This attack is only possible if the attacker has physical access to a node in the system. By means of issuing a reset on the machine and requesting a single-user boot, he will be the super-user of that machine, i.e. since there is only one user, he is the super-user and he has all privileges on that machine. As a follow-up, these privileges can be transferred, e.g. to the server, by means of using the SUID-facility (Set User ID on execution) in UNIX. The attacker has then gained super-user privileges over the whole system. 7 attackers made this follow-up successfully.

The next most common attack method was to use a crack program, which is a publicly available password guessing program. Most UNIX systems store passwords in an encrypted form in the /etc/passwd file. The principle of this attack is simply to "guess" a password, encrypt it, and compare the result with the contents of the /etc/passwd file. This attack is possible since the encryption is performed using a publicly known one-way encryption algorithm. The guesses can be based on the contents of an electronic dictionary and it is only the size of the dictionary and the speed of the system that limits the number of passwords that can be tried. The used system permitted a guessing rate of approximately 10,000 guesses per second. However, one has to bear in mind that each original password has to be tried in a number of different versions. 6 attackers made use of crack programs during the experiment with 2 of these attackers trying 2 variations of this method and reporting them as two distinct attacks. 4 of the attackers reported success from using crack and these successes consisted of 52 instances of password cracks in all (although many of these are repeated cracks of the same password). For example 7 different passwords were revealed during the execution of one single crack program, running for circa 24 hours on 22 SUN work-stations in parallel. These passwords were kalhus (two Swedish three-letter words), ascona (Opel Ascona is a car name), oaxaca (which turned out to be a Mexican county), romanus (Family name of a well known Swede), siddhart (Siddharta is Buddha's second name), parvin (the name of the user's Iranian wife) and bol (means nothing, but is obviously far too short). It is worth noting that most of these passwords, even if they are quite uncommon, could be found in a dictionary or encyclopedia, a fact that should disqualify them as passwords.

The third standard attack method used was to monitor the key strokes made by ordinary users of X Windows in the hope that they would login, thus revealing their password. As for crack the attackers obtained some publicly available software in order to carry out this attack method. 4 attacks reported were of this kind and this includes 2 attack reports from a single attacker. The only success reported from this method occurred in the first attempt by this attacker; 28 passwords were obtained and one of these was actually the system owner's password.

4.2.3 Other Attack Methods

In addition to the standard attack methods described above, there were 2 attempts to use the security-enhancing program "cops" [Curry 1990]. This method has not been classified as a standard attack method since in both cases the attackers did not really seem to know how to use this program as intended for revealing security problems; little effort was expended on this method by either attacker and in neither case was the attack reported as successful.

There were also a number of less straightforward attacks. These latter included using Trojan Horses in order to cheat another user into executing a faked version of the "ls" command (ls = list files in current directory), thereby creating an executable file with the SUID-flag set: 2 attacks were of this kind, 1 failed (the experiment ended while this attack was on-going) and for the other a breach was reported but in fact this attacker just proved that this method would work effectively (by testing it) but never actually caught anyone out. Also, a few attackers expended some effort on searching through the file system for writable files and checking various potentially interesting directories for possible vulnerabilities. Most of these attacks were claimed to be successful in the sense that they learnt something useful about the system relating to security: only for some of these cases, though, were breach reports actually filled in. The results varied from finding writable files so that they could potentially cause damage to getting root on the server. One attacker found that the device buffer for the Sun work-station was by default readable (and writable), meaning that it was possible to monitor all output on the screen. This attacker found that he could do this on machines used by ordinary users, and gave an associated breach report, but that it was not possible to do this on the machine usually used by the system administration.

4.3 Interaction during the Experiment

4.3.1 User Interaction

There was no significant interaction with other users of the system reported during the course of the experiment. Only the users who got their passwords cracked were informed of this after the experiment by the system owner, and were requested to change their passwords.

4.3.2 System Owner Interaction

For the system owner the experiment also passed almost unnoticed. The system owner took one action that was definitely related to attacker activity; changing back the root password which had been changed by an attacker. The owner also had to shut down the system due to excessive loads a few times. This may have been due to the execution of password guessing programs and in some instances it was known that this was the case. Indeed, some attackers

reported in the relevant parts of the attack reports that their activity had been stopped by the system owner when they were running crack. This was the only reporting from attackers that any of their activity had been stopped by the system owner. One attacker, though, reported a couple of security related changes in the system, which were made during the experiment, which they felt could have changed attackers' chances of breaching. Unfortunately, one of these changes, although not made intentionally by the owner as a result of attacker activity, did indeed make it significantly harder to breach the system using one attack method.

Disk crashes, enforcing a subsequent reload of the system from back-ups, occurred twice during the duration of the experiment. Since disc crashes normally occur very seldom this is surprising, but we have no reason to believe that the cause was attacker activity; a more likely explanation is correlated faults on some new disks which occurred at this time.

4.3.3 Experimenter Interaction

The experimenter did not have to interact with the attackers very much during their attacks. He (or sometimes a back-up) had offered to be available on a 24 hour basis but it turned out that there was not much need for his services. He mainly just had to take care of incoming reports and send electronic mail messages to try to encourage and motivate the attackers to work more.

4.4 Summary of Attackers' Final Evaluation

This section summarizes the opinions of the attackers collected in the attackers' final reports (see appendix C.5) and during the summing-up meeting and informal discussions. As stated above, this meeting was only attended by the 6 most productive attackers, limiting its usefulness. In particular it was not possible to ask the absent attackers why *they* had not been more active and how they might have been better motivated.

The purpose of the final report was threefold. Firstly we wanted to get information on what the attackers thought about the structure and layout of the experiment, for example the adequacy of the pre-experimental information provided, the report forms and the methods for awarding points and prizes. Secondly, we wanted to get some idea of the nature and quality of the collected data, for example to find out whether the attackers had co-operated, how many attacks were unknown to the attackers before the experiment started, whether they believed that their reported effort data was correct, whether they had reported all relevant activity and so on. Finally we wanted the attackers' opinions on how they could have been motivated to be more active and how they thought improvements might be made for future experiments. The subsequent discussions with the attackers were aimed at clarifying and expanding the information received in the final reports.

The attackers found the presentation of the project and the introductory information to be generally good. In spite of this, some of the comments in the attack and breach reports suggested that some of the attackers had in fact not completely understood the objectives of the experiment. Also, it seems likely that the most productive attackers may have understood the introductory material the best.

The report forms were judged to be alright, but a few adverse comments were given. The most important of these was that there was too much to fill in and that some information was asked

for more than once. Furthermore, the attackers said that they found it hard to estimate the consequences of their attacks/breaches, whether for themselves, i.e. what they had gained, or for others, i.e. possible damage. Several of them stated that they found it hard to estimate the effective elapsed time as well as the CPU time. They also found it hard to allocate time to specific attacks, since they often worked on several attacks in parallel. It was evident from comments in the attack and breach report forms from some of the attackers who did not attend the final meeting that they held similar opinions on the difficulty of filling in these reports.

Most of the attackers said that the methods for awarding points (and the prize) did not affect their activity and that their own self-motivation and interest was the over-riding factor which influenced their attacking behaviour.

Only one attacker reported having been helped by another with one of his attacks. In general then, it seems that the attackers did not co-operate but there were some comments in the attack and breach reports which suggested that some attackers may have got their ideas from other attackers' activities. The majority of the attack methods tried were claimed to be known of in advance by the attackers. Comparison of their initial ideas on how to attack the system (from the attacker background reports - appendix C.1) with the attack methods tried seem generally to support this. There is evidence to suggest that a substantial amount of activity, in particular time spent thinking or planning and general searching around the system, was not reported by the attackers as effort expended.

Most attackers that we consulted strongly favoured working in small groups over working alone. There were several reasons that they put forward in support of this - working in groups is closer to real-life, it would allow exchange of ideas stimulating more activity and more of a variety of attacks and breaches, it would allow optimisation of effort (i.e., more efficient), and so on. Other suggestions by the attackers for motivating more activity included using a more secure system so that it was more of a challenge to break in. Some comments in the attack and breach reports supported this; some attackers seemed surprised and disappointed that they had found it so easy to get root access on the server and ceased to participate in the experiment after this had been achieved.

As a whole the attackers who attended the summing-up meeting said that they found the experiment to be well organized and interesting, and stated that they would be quite happy to participate in some future similar activity.

4.5 Adequacy of Data Collection

Our ultimate objective in conducting this type of experiment is to gather data on the effort expended, and breaches (and rewards) obtained, by attackers in order to gain empirical data to aid in developing probabilistic models that can be used for forecasting the security of systems. In this pilot it was anticipated that the data obtained would be deficient for the purposes of this ultimate objective and the primary aim was to move towards a successful experimental methodology. The data was indeed deficient and these deficiencies (discussed in the remainder of this section) led to pointing out some ways (discussed in section 5) in which the experimental methodology could be improved for future experiments.

4.5.1 Measuring Effort

To summarise, much of our data collection was concerned with collecting data which we thought was relevant to investigating the possibility of finding an appropriate measure of the effort expended by the attackers.

Firstly we had collected information relating to attacker experience before the experiment started (see appendix C.1) and, although we had chosen attackers which we hoped would be fairly similar in their experience, some differences between them were suggested by the attacker background reports; for example some attackers seemed to know quite a lot about how to attack the system already, while others had few or no initial ideas. It is not at all clear, however, how this qualitative information on these differences could be quantified in order to measure their impact on probability to breach. In addition to this it is likely that attackers with the same previous experience may simply have different abilities, and even qualitative data on attacker ability seems very difficult to collect. No attempt to collect such data was made in this pilot.

This problem of quantification is also present for most of the effort related data which we collected for the attacks actually done during the experiment (see appendix C.2). There were two problems: how to get a quantified measure of the different factors which we think are relevant to effort and how to combine these into a single quantified measure of effort expended by the attackers. These problems of quantification can probably be more easily discussed by dividing the observed attack methods into different types of attacking activity, identified by the nature of the effort expended.

Probably most easily quantifiable, are those parts of the attacks where the effort expended was just the time spent working by the attacker, for example, learning about the system (on- or off-line), manually searching files for vulnerabilities or the single-user boot-up sequence and the SUID follow up. In some cases attackers reported having used external resources (e.g. documentation, friends, ...) in order to learn about possible ways to break into the system and it is not at all clear how to take into account the use of such external resources in addition to the learning time spent by an attacker. Next, there are those attacks where the attacker executed some software for a significant period of time. In such cases there tended to be working time expended directly by the attacker in preparing the software for execution, and later in monitoring the results intermittently during execution, but, in addition to this we have to take account of the resources used in actually *executing* the software, for example, duration for which it is executing and/or CPU time; some attackers executed software on many machines in parallel and use of this additional hardware also needs to be taken into account. In many of these attacks the software used by the attackers was publicly available (e.g. crack, xkey). In such cases the attackers spent time retrieving, learning about and setting up this software, in preparation for a direct attack on the target system, but measuring the attacker working time (and software execution time) alone does not seem sufficient for taking into account the impact of this kind of additional resource on the chances of success. Finally we have "dormant" attacks (e.g. Trojan horses) where, after the attacker has spent some initial time in setting up the attack, it lies dormant waiting for some action by another user to invoke it. This attack may also involve the use of publicly available software (e.g., xkey) and some further working time by the attacker monitoring results, but the special property which distinguishes this type of attack from others is that activity in the normal system operation (not under the control of the attacker) affects whether this type of attack is successful or not at a particular instant in time. In cases such as

these perhaps some measure of the normal system operation needs to be taken into account in constructing an appropriate measure of the exposure of the system to this kind of attack.

In most cases carrying out a single attack method involved the attacker in a combination of these different kinds of activity, e.g. some learning and planning, followed by writing some software, followed by executing the software and so on. In many cases the attack reports received from the attackers allowed us to identify these different types of activities and to recover the different time data and additional resources discussed in the previous paragraph but it is certainly not clear how to combine them into a single quantified measure of effort expended by the attacker.

In general, examination of time data reported in the attack reports for particular attack methods revealed quite a lot of variability, for different attackers, of working time (if appropriate) to breach. Sometimes this variability really reflected some clearly identifiable difference, for example, different initial expertise or ability. For example, even for the most common attack, the single-user boot-up sequence, one attacker at least reported that, although he *did* know that such a method existed beforehand, he had to do some digging around in manuals in order to find out exactly how to do it; as a result this attacker spent much more time before he achieved a successful breach using this method than most of the other attackers, who simply needed to go to a work-station and try it out to see if it worked. In fact, it was clear that even though attack methods tried by the attackers were mainly claimed to be known about before the experiment began, there was usually quite a bit of learning and other preparation needed by the attackers for these methods and the amount of work needed generally varied from attacker to attacker. In other cases inaccurate reporting by the attackers of the time that they spent working was a significant contributory factor to this variability. There was clear evidence that some attackers tended to underestimate the working time which they had spent. In particular, although some attackers reported time spent planning an attack, others only reported time when they were directly interacting with the system.

Other likely sources of inaccurate or incomplete time reporting by the attackers were also evident. In many cases the attackers did not give the CPU time used (unfortunately, some just commented that we should retrieve it from our automatic logging), or even the duration, for execution of software in an attack method, but it should be possible to retrieve some of this information from the automatic logs (providing, of course, that execution was on the target system). For "dormant" attacks it was sometimes not possible to identify from the reports the duration for which these attacks were on-going. Often, time reports were not filled in at the time of the attack (and were certainly not mailed to the experimenter until some considerable time afterward) and it seems likely that the attackers may have only roughly estimated the time that they spent. In one case more than one (unconnected) attack method was reported in a single attack report, and as one activity, and it was not possible to recover the data relating to the different methods. In addition to this some activities were not reported at all or else initial reports were sent indicating that an attack had begun but no follow up reports (containing effort data) were sent.

There were also some very basic problems with the time report itself. For example, even when planning time *was* reported, much of this data was lost since it was often not possible to tell from the attack reports whether this time had been included as part of the first activity in the time report or not. In addition to this units were often unspecified, dates were wrongly filled for the time period, or no date was given at all making it impossible to examine the effort expended as a function

of calendar time. This absence of reporting of calendar time duration of activities sometimes made it difficult to identify how long software was left running and "dormant" attacks were on-going.

It is clear that before we can address the quantification problems for effort expended discussed earlier we need to sort out some of these more basic problems with recording of data which *can* be quantified.

4.5.2 Breach Events and Rewards

As mentioned earlier, it was apparent from examining the data, that not all attackers filled in breach reports (see appendix C.3) for the occurrence of the same consequences of their attacks. It is interesting to note that, in many cases, attackers reported in the last part of the attack report (see appendix C.2) that they *had* gained some reward (e.g., local root, some learning about the security of the system) even if they had not filled in a breach report associated with this, or this part of a, particular attack. In some reports comments were made like "this is not really a breach but ...", " ... to me a breach is the root prompt ...". It is important to note that this problem is not related to the issues discussed earlier in the paper, of the subjective nature of attacker reward, but is merely to do with the different preconceived interpretations which the attackers had of the term "breach". Instead of reporting all events with which they associate reward as breaches, broadly speaking each attacker only reported breach events as those with which they associated large or particularly significant rewards.

In spite of these problems, taking into account comments on rewards reported in both the attack and breach reports, it was possible to learn quite a lot about the nature of the reward processes for the attackers. Thus, although asking for similar reward related data on both the attack and breach reports may have been one of the sources of their complaints that they were asked for the same information more than once, as far as our data is collection is concerned this was useful, since otherwise much data relating to attacker reward would have been lost.

Firstly, it was quite clear that each attacker generally associated very different rewards with the achievement of different successes. Fairly obviously, attackers associated a very high reward with getting root on the server ("the ultimate goal"[2]for many of them) while, for example associating a lesser reward with getting root on a local machine. Again, fairly obviously, in cases where one (*partial*) breach was used to achieve some further goal, for example local root to achieve root on the server or discovery of some other vulnerability in the system which later led to getting root on the server, the attackers attached higher reward to the ultimate breach than to the partial breach. The main other type of success with which the attackers clearly associated substantial rewards was that of obtaining passwords. Other events with which the attacker associated rewards were generally to do with learning about security issues in the system. In some cases the attackers seemed to attach reward to having proved that a particular attack method would work even though no higher objective was achieved (e.g., proving their implementation of a Trojan horse could catch passwords, but not subsequently actually getting any passwords from this method).

---

[2] Although some attackers appeared to view this as the *ultimate* goal, there are circumstances under which this is not always obviously the case; not all information or rewards can necessarily be obtained through having super-user access.

Even though it is sometimes possible to rank different successes in order of the amount of subjective reward to the attacker, this is generally not the case, particularly when successes are of different types. For example, for ordinary user accounts where an attacker is indifferent between these users, it is probably true that most attackers would rate getting many passwords for different accounts more highly than getting just one. On the other hand how would they rate getting many passwords of ordinary users against getting the system owner's password? Ranking of even more disimilar successes (e.g., local root versus some users' passwords) seems even more of a problem.

Most of the previous discussion on reward refers to the description by the attackers of what they had actually *gained*. In addition to this the attackers were asked questions about potential *loss* (damage that could be done) to the owner and/or users, and potential *loss* to the attacker due to the risk of detection (whether they could be, or had been, detected by the owner and so on). Some of the attackers gave quite a lot of information about what the loss to the owner or users of such a system is likely to be as a result of such a successful attack, and whether their actions would have been likely to have been detected if this had been a real attack on a real system. Unfortunately, due to the artificial constraints that we had to impose on the attackers, the information given here is probably of limited relevance. Since they were not actually allowed to damage the system, and since they were not really going to suffer any consequences from detection by the system owner, it seems that these issues are unlikely to have effected their attacking behaviour very much. It also seems that asking leading questions about malicious damage, for example, may result in them giving us information on how much they *could* do, but does not really give us any idea about what the attackers *would* do in reality. Thus, since rewards are subjective, it seems likely that those factors which in real life could generally be expected to be part of attackers' subjective rewards, were not part of the attackers' rewards in this experiment.

A second significant feature about the nature of the attackers' subjective reward processes which was apparent from examination of the reward-related data collected, was that distinct differences could be seen in the nature of the rewards gained against the effort expended, dependent on the attack methodology being used. For example, for some attack methods (e.g. seeking a particular vulnerability by searching files in the system, single-user boot-up) the aimed at reward (e.g. finding the loophole that was being searched for, getting local root) could generally be considered to have been gained at a single distinct instant, usually at the end of the effort expended in using this method. At this point the attacker generally stops since he has achieved their goal, and proceeds with another attack (which may be based on the results of this one, or may be unrelated). On the other hand, for on-going attacks such as crack or key-snooping, it is not so clear when the rewards are obtained within the effort process. For example, with crack is it correct to view the reward of each password as being obtained when it has been cracked by the software or is it more accurate to view the reward as having been obtained when the attacker later monitors the results. Since rewards are subjective maybe the latter is more appropriate, but, wherever we decide to place these events, different rewards are generally being obtained via this type of method at different instants within the process of effort expenditure, as opposed to all at once. Notice that in the examples considered so far the reward process is *discrete*; in all cases there are *events*, within the effort expended, and each event has an attached reward. No attacks were reported where the reward was clearly *continuously* accrued as effort was expended although we cannot rule out the possibility that this *may* have occurred in unreported activities were the attacker was learning in some continuous way.

A final aspect of the attacker's reward processes which was noticed was that there was some evidence to suggest that different attackers *did* seem to have different rewards associated with the occurrence of the same successes (in other words different attackers had different motivation driving their attacking behaviour). For example, it was apparent that some attackers seemed to get more personal satisfaction from being in a position to do something malicious than others, some found learning particularly important and attached more reward to this than other attackers, some seem to get a lot of satisfaction from having done something clever, and so on. Most of the data which suggested that attacker reward processes are subjective is data which by its nature is particularly difficult to collect.

Most information collected on the attackers' rewards was in its nature qualitative. Even where it is possible to rank successes, as discussed above, most of the information given by the attackers about reward did not give us any idea on how we might actually *quantify* such rewards. Even in password guessing, the one case where we obtained numbers (of passwords obtained), these numbers are not particularly helpful. Leaving aside the fact that in reality an attacker might not attach equal gain to getting different passwords, it seems likely that there is a stage at which attackers would simply not feel they were achieving any more gain however many new passwords they obtained. Even if such gains *could* be quantified combining other aspects (for example loss due to detection by the owner) with these reported gains into a single quantified reward measure is likely to be very difficult.

To summarize, although some interesting insights were gained in this pilot relating to attackers' reward processes, quantification problems predominate, as with effort. Most of the reward-related data was qualitative and it is not at all obvious how to achieve a quantitative measure of reward. The purely subjective nature of the reward process for the attackers makes this problem of quantification of rewards particularly hard. Again, before these quantification problems can be addressed, there are some more basic problems with the reports used in our pilot, concerning collection of data on rewards, which need to be resolved for future experiments.

4.5.3 A Note on Changing Beliefs

To a certain extent, we tried to collect data on how attackers' beliefs about effort and reward changed as their activities progressed. We did this by trying to get them to report before the attack (see appendix C.2, the first part of the attack report) about what they thought they would get, in terms of rewards, from the attack and about the effort that they thought they would have to expend, so that this could then be compared with the actual later reported results of their attack. This attempt at collecting data before the attack turned out to be particularly deficient. One reason for this is that most of them were very reluctant to fill in forms before they had begun the attack since they just wanted to proceed with attacking the system. As a result many of the reports were filled in after attacks had been done and so we cannot expect that the data is representative of their beliefs beforehand. Another reason was that, usually their beliefs about the reward that they would obtain, given that they succeeded in their attempt, did not really change that much. For both of these reasons the attacker would feel that they are being asked for the same information twice (before and after the attack) and this is almost certainly the main source of their complaint about the reports repeatedly requesting the same information.

On the other hand it was quite clear that the attackers learnt quite a lot (ie., their beliefs *did* change quite a lot) during their attacks. One attacker's comments suggested that he did not like giving an account of what he thought beforehand, because he quickly changed his mind as he then proceeded with the attack. Many attackers expressed surprise that an attack method that they had tried did work. This was true even for many of the standard methods that the attackers tried and highlights the point that, although the attackers claimed that they knew these methods beforehand, that did not mean that they knew that these methods would work on our target system. Since attackers generally had different expectations about what would work and what would not, it seems clear that they would allocate their effort differently between different attack methods. In some cases the attackers found other things which clearly surprised them, for example, they sometimes stumbled across a vulnerability in the system which they had not been seeking or expecting. These were the only circumstances where beliefs about rewards (as opposed to effort required or probability of the success of a particular method) were substantially changed, although attackers could not know, for example, how many passwords they may expect to get from running crack for a particular length of time, say.

5 Conclusions and Recommendations for Future Experiments

The outcome of this pilot experiment was much better than we had initially anticipated. More than a dozen attackers performed real attacks on our target system and many of these attacks resulted in breaches; initially we had been worried that no breaches would be achieved by the attackers at all, and this was our reason for selecting a system with only modest protection.

The most important experience from this pilot lies in the field of experimental methodology. We have now gained a deeper insight into the type of problems that we have to deal with when planning for the next experiment, and new ideas of how to solve these problems. In addition, we *did* learn quite a bit about the *nature* of the attacker reward process versus effort expended, even if the pilot did not give us enough data to address directly the detailed *quantitative* modelling issues.

One of the main problems encountered was that many attackers were not motivated to use more effort in attacking the system and into trying more of a diversity of attack methods. Encouragement from the experimenter was not very effective in solving this problem. Some of those who initially expressed an interest, did not attack the system at all. Very few attackers actually tried more methods after breaching once or twice. Many could not see the point of trying anything else after they had root on the server. Generally, attackers did not try attack methods which were novel to them in the sense that they were initially (when the experiment started) completely outside their imagination. This is not to say that they were not learning anything new about the target system and about security issues in general, but their ideas usually started from some attack method which they had known of a priori and some attackers initially had quite a lot of ideas while other attackers had few or no initial ideas.

There is an interesting dichotomy here. The behaviour of many of these attackers (and in particular those who stop when they achieve the first significant breach and even those who try nothing because they don't have any ideas) is probably quite *realistic* for *this* particular system and *this* particular attacking population. But, even though we want our experiments to be representative of reality, unless we encourage attackers to try harder and to try new methods even after they consider that they have already succeeded with some sufficiently high gain, we would not get very

much interesting data. In a sense here we are trying artificially to simulate fixes in the system, after an attacker has exploited a particular vulnerability: in other words, the attackers need to go back and behave as if this system did not have that particular vulnerability, even after having discovered it. Another way to look at this is that, instead of investigating just one fixed system, we are investigating a number of systems with varying levels of protection. It is clear that we need to find ways in which the attackers will be more motivated in order to get enough useful empirical data.

Some attackers were not motivated since the security of the target system was so low, with fixes not in fact ever implemented, and they wanted more of a challenge. If a system which was better protected was used then these attackers may be more motivated. Unfortunately, this may also result in fewer breaches occurring, so again we may be left with a paucity of data relating to the reward processes (even though in this situation we would be likely to have more data relating to when attackers finally give up on a failed attempt). In addition, this might have the effect of more attackers finding it so hard that they do not participate at all, again resulting in less data. An alternative way to make it more of a challenge for the attackers is to start with a system which is not very well protected and then to fix *for individual attackers* as they discover vulnerabilities. This is probably the ideal experimental set-up but almost certainly far too difficult to implement in practice. Fixing (assuming fixes are possible) in response to a breach by one attacker would have to only effect that attacker, or we would have to accept that fixing for one would effect the chances of another. It is difficult to see how to stop fixes for one attacker affecting another except by only allowing the attackers to attack the system one at a time. Fixing resulting in interaction between attackers may be an interesting consideration for future experiments, when we wish to consider attackers viewed as a population, but it will not help us with investigating the initial more simple problem of single attackers. In either case, any fixing is likely to involve an awful lot of work on the part of the experimenter and the owner and so this is probably not a practical solution.

Another way in which attackers themselves suggested that they would be more motivated would be to allow them to co-operate by working in small groups Having the system attacked by small teams of two or three rather than by individuals would provide some mutual reinforcement and encouragement to continue with new tactics, which may lead to more activity. On the other hand it is not necessarily true that this will result in more data. One problem, of course, is that working in groups would probably result in smaller sample sizes. Also, it is not clear whether working in groups might result in attack methods being tried by a group which were initially unknown by all the attackers in that group. Further, we cannot assume that co-operation would result in those attackers which were not active becoming more active.

There are a number of further considerations with respect to working in groups. Firstly there is the issue of how representative of reality this would be; do real attackers usually co-operate in this way or do they usually work as individuals? The extent of the similarity between the behaviour of a single co-operating group and the behaviour of a single attacker will obviously effect whether or not data from one could be useful for modelling the other. In addition there is the problem of deciding on how to divide the attackers into groups. If they were allowed to choose their own groups, and they would probably prefer to do this, then this may result in them dividing themselves into groups of similar ability, for example, which may result in less data rather than more. Thus it may be better for the experimenter to enforce some grouping on them but it is not at all clear how this grouping should be done. Finally, the reporting may have to be changed in order to cope with the fact that they are working in groups rather than as individuals.

One final possibility for increasing the attackers' motivation may be to improve the mechanism for awarding points and prizes. In this pilot these attackers seemed little motivated by the existence of this award mechanism. Even the very active attackers who were behaving as we wished them to were still more motivated by their own sense of self-satisfaction (ie. that they were being clever in finding new ways) than by our award mechanism. If we decide to increase the awards in order to motivate the other attackers into behaving similarly, there is a delicate trade-off to be made: if the awards are too big, the attackers might have a tendency to cheat in order to maximize their own benefit; if they are too small, it will not have the intended motivating effect. One further alternative mechanism to consider is to give awards to all attackers in accordance with the successes which they achieve, since having only one winner may have the effect of other attackers giving up as soon as they feel that they are out of the game. The practicalities of such awarding mechanisms, and their probable impact on attacker behaviour, need to be considered for future experiments.

The reporting system generally worked fairly well. Using the electronic mail for all reporting (from attackers and between experimenter, owner, and attackers) was a very convenient way of collecting information and is obviously much easier than having to collect papers from all participants. Even though the attackers had the option of reporting manually, all of them preferred to use e-mail. One disadvantage though was that the attackers chose when to report and when not to, and in many cases e-mailed their reports much later than we would have prefered.

With the reporting forms themselves there are many improvements that need to be made. Clearly, they need to be made much shorter. Using the term "breaches" was generally misleading; it might be better to just use some format which would ask when they had obtained some reward and so on, and incorporate all these questions on one report instead of having a separate breach report. In addition, asking for beliefs before they have started the attack is probably not necessary. These changes should result in questions relating to consequences of successes really only being requested once, which would substantially shorten the length of the reports.

Examination of the attackers reports revealed that some of the questions were misinterpreted and need to be made much clearer. In addition more detailed changes, relating to the time reporting in particular, is required in order to get more accurate data relating to effort.

For many reasons it may be beneficial to consider the use of some more interactive mechanism for reporting rather than asking the attackers to fill in reports for particular attacks or breaches. Obtaining instead some kind of on-going activity report may be a more accurate reflection of the kind of data with which we are dealing and may help the attackers to report more accurately. In particular, it may help them to report more accurately different methods which are being done in parallel and encourage them to report activities which they do not necessarily associate with a particular attack (eg. planning and investigative work). In addition it might be possible to tailor the questions being asked about time data in order to accommodate the particular attack method which is being used, in order to allow more accurate collection of this data. One way to implement such a reporting mechanism may be to build a software tool which they can run interactively during their on-line attacks, but that also incorporates reporting of off-line activity. This would allow time records to be made internal to the software itself as a back-up for incorrect time reporting by the attackers. Some considerable thought is required to work out the feasibility of the development of

such a tool for future experiments. Clearly any such automatic data collection tool would have to be very easy for the attackers to use.

The automatic logging system worked satisfactorily, except for a few incidents where data was lost. For future experiments it may be useful to develop a tool in order to extract the data that we want from the resulting logs, since, due the large amounts of data involved, doing this manually is a non-trivial task.

In this pilot we have clearly concentrated very much on the *attackers'* subjective reward processes. Since predictions of interest of operational security are probably more related to the owner's loss process, collection of data on the owner's subjective views should be considered for future experiments. In addition to this it may be interesting to use, in future experiments, better protected target systems with populations of attackers who are more expert, since, in general, this may be an area which is of more interest to the security community. On the other hand, it may be that models constructed from empirical data on less secure systems, with less expert attackers, can also be used for more secure systems.

References

[Allwood, Bjorkhag 1990] C.M. Allwood and C.G. Bjorghag, "Novices' debugging when programming in Pascal", *Intl. J. Man-Machine Studies*, vol. 33, Academic Press Ltd., pp. 707-724, 1990.

[Curry 1990] D.A. Curry, *Improving the Security of your UNIX System, SRI International*, Final Report, ITSTD-721-FR-90-21, April 1990.

[Garfinkel, Spafford 1991] S. Garfinkel and G. Spafford, *Practical UNIX Security*, ISBN 0-937175-72-2, O'Reilly & Associates Inc., 1991.

[Grampp, Morris 1984] F.T. Grampp and R.H. Morris, "UNIX Operating System Security", *AT&T Bell Laboratories Technical Journal*, vol. 63, no. 8, Oct. 1984.

[Littlewood, Brocklehurst et al. 1991] B. Littlewood, S. Brocklehurst, N.E. Fenton, P. Mellor, S. Page, D. Wright, J.E. Dobson, J.A. McDermid and D. Gollman. "Towards Operational Measures for Computer Security", PDCS Project Second Year Report (ESPRIT Project 3092), vol. 3, chapter 2, 1991.

[NCSC 1985] NCSC. "Department of Defense Trusted Computer System Evaluation Criteria", DOD 5200.28.STD, National Computer Security Center, Department of Defense, 1985.

[Olovsson, Jonsson, 1992] T. Olovsson and E. Jonsson, "Security Forms for Protection against Vulnerabilities in Computer Systems", *Proc. IASTED Intl. Conference Reliability, Control and Risk Assessment*, Washington DC, USA, ISBN: 0-88986-171-4, 1992.

[Richie] D.M. Ritchie, "On the security of UNIX" in UNIX Programmers Manual, section 2, AT&T Labs.

Appendix A Brief for the Attackers (and the Owner)

Pilot experiment in quantitative security evaluation: briefing for participants

Chalmers University, Göteborg

City University, London

1 Introduction and background

This experiment arises from some work carried out in an ESPRIT research project, PDCS (Predictably Dependable Computing Systems), in which City and Chalmers universities are partners. The overall aim is to develop a quantitative theory of *operational security*, similar to that which now exists for reliability (for hardware and, more particularly recently, for software).

In reliability, the important point is that we express our confidence in a system probabilistically in terms that reflect naturally its ability to operate successfully. In particular, there is an acceptance that there is a suitable "time" variable, and it is assumed that failures in time occur in a random process (albeit perhaps a complex one). This natural randomness of the failure process arises in reliability for several reasons. In the case of software reliability, for example, we cannot be certain when the next failure will take place because of the natural unpredictability of the operational environment. The input space is typically very large and complex, and we never know with certainty which inputs will be selected in the future: in particular, we shall not know when a failure-prone input will be selected and so trigger a design fault. We take account of this natural unpredictability in probabilistic measures of reliability such as the *reliability function* (the probability of failure-free operation for a specified time t), the *rate of occurrence of failures*, the *mean time to next failure*, etc.

Computer security, on the other hand, has not so far been expressed quantitatively in terms that acknowledge a similar uncertainty and a dependence upon the nature of the operational environment. Instead, security *levels*, such as those of the Orange Book, are concerned with the nature of the development process, and of the design architecture. They are similar in some respects to the general arguments about the quality of the development process that preceded the availability of techniques for direct measurement of operational reliability. Yet security clearly *is* an operational concept. Intuitively, when we say that one system is more secure than another, we are asserting a belief that it is less likely to exhibit a violation of the security policy than the other - for example, it is less likely to be breached by a deliberate attacker. Although a high security level may tend to give higher "operational security" in this sense, this cannot be guaranteed, just as the use of good development practices cannot be guaranteed to result in high reliability.

Our aim, then, is to provide the means whereby the operational security of a system can be measured. In thinking about this problem, the example of reliability was foremost in our minds. Reliability theory can be thought of as the description of processes involving *failure events* taking place in *time*. In the case of security, a naïve view would be that we could similarly model the process of *security breaches* in time as well. However, it quickly became clear that, although there were parallels, there were some important distinctions.

In the case of security, time alone does not seem to be an appropriate variable. In particular, elapsed time when a system is not under attack is irrelevant for the evaluation of resistance to deliberate attack. Instead, we need to consider an variable which captures the *effort* expended upon attacking the system - then informally a system that requires more effort to be expended until it is successfully breached is "more secure". We might then define operational security in terms analogous to reliability: the *security function* would be the probability of resisting a security breach for the expenditure of effort *e* by an attacker; other measures such as the rate of security breaches per unit effort, mean effort to the next security breach, etc, have obvious meanings.

Although the notion of security breach itself - essentially a security "failure event" - is often sufficient, it is also sometimes too simplistic. Firstly, it is clear that the value of security breaches to attackers can vary enormously[3], and this notion of *reward* should be incorporated into the measures. Secondly, whilst attackers of a system may certainly gain something of value from these breach *events*, they can also acquire reward continuously - for example by gradually learning about the system. For these reasons we believe that, instead of the failure events in time that characterise the reliability process, the most general description of the security process would be in terms of a *reward process*, involving both discrete and continuous increments, against *effort expended*.

Another area where reliability and security differ somewhat is in the notion of "viewpoint". Consider, as an example, the "value" of a security breach: clearly the system owner's view of his/her loss will usually differ from the attacker's view of his/her reward (and this last view might differ from attacker to attacker).

2 Overview of experiment

The purpose of the present experiment is to investigate the feasibility of this new approach to security evaluation by trying to measure the key quantities, effort and reward, from different viewpoints. The experiment will comprise several student *attackers* of a real distributed computer system (which will have other real users), a system *owner* (the real system administrator), and the *experimenter* (controller of the experiment). The experimenter is the final arbiter of all issues that arise during the experiment: attackers, in particular, should consult the experimenter if they have any doubts about the validity of any of their activities within the experiment.

Whilst it is inevitable that such an experimental set-up will be somewhat lacking in realism in some areas, our intent is, of course, to seek as realistic behaviour as possible from all participants.

---

[3] This is also true in reliability: the cost of different types of failure can vary a great deal. However, not all reliability models take account of this.

Our main demands of both attackers and system owner, therefore, are that they should behave as they would if they were carrying out their activities in real life, *restrained only by the rules of the experiment* (see later for details), and that *they report their actions as accurately, honestly and completely* as the experiment requires (see later for details of reporting requirements).

The attackers will be expected to attempt to carry out security breaches, in their own time in the duration of the experiment, and document all their activity on the forms that will be provided. It is important that this activity is as realistic as possible, and the only constraints are those spelled out in the rules of experiment. The constraints mainly arise from the fact that the system under attack will have real users, and it is our intention that these should not be disturbed by the existence of the experiment. Neither will it be allowed to cause actual damage to the system. However, attackers will not be forbidden to carry out attacks that might have these effects; instead, they will be expected to consult the experimenter at an early stage in such an attack. The experimenter will make a judgement about how, or whether, the attack should proceed.

When filling in the forms that report on the activities that they have carried out, it is inevitable that much of the information we require from participants will be very subjective. This is to be expected, and does not detract from the objectives of the experiment. For example, we expect that there will be interesting differences in the perceptions of the rewards associated with different breaches between attackers. Attackers are encouraged to describe these subjective views as accurately and completely as they can, as well as providing the more objective data on the effort they have expended.

To provide some motivation to the attackers to participate fully in the experiment, there will be points awarded by the experimenter at the end of the experiment for the reward arising from successful breaches (and for activity by attackers which, whilst falling short of actual breaches, nevertheless provides information that may be of assistance in achieving a future breach). At the end of the experiment there will be prizes awarded to the highest point-scorers. Attackers should bear in mind that the experimenter will award points according to the following principles:

- The points that will be awarded will be chosen by the experimenter to reflect his/her view of the rewards that would be obtained by an "ideal" attacker.
- The points system will reflect the gain to the attacker due to a breach. This gain will include potential harm to the system owner and/or users, information acquired, ingenuity of attacker, ...
- The points system will reflect the consequences to the attacker of detection (negative reward) by the owner.

However, there are some exceptions to these principles which reflect the fact that this is an experiment and not real life:

- The attacker must not unacceptably disrupt the system. If the attacker gets into a position of being able to do so, he must stop and contact the experimenter, but his potential gain in being able to so disrupt the system will be reflected in his points.
- Each attacker will only be rewarded once the first time they carry out a particular breach, and not for subsequent repeated breaches. Reward will be given, though, for subsequent

(new) breaches to which this breach might lead, even if this breach is being repeated in order to achieve the new breach.

- The points system will not penalise attackers who are detected only because they are using the target system for experimental reasons (i.e. in normal circumstances a different system would have been used)

In addition to points gained, each attacker will be given a fixed gratuity for taking part in the experiment, and obeying its rules, regardless of whether they achieve effective breaches or not. This gratuity will not be given to any attacker who knowingly does not obey the experimental rules. In particular, no gratuity will be given if they knowingly cause unacceptable disruption to the system, if they collaborate or communicate with other attackers, or if their reporting is inaccurate or incomplete. Attackers who have violated the rules and are thus not eligible for this gratuity will also not be eligible for prizes.

Attackers should note that for the purpose of experimental realism, they should regard only the system owner as their opponent and *not* the experimenter. In particular, the attackers may attempt to conceal their activity from the owner but not from the experimenter. There will be no penalties in any circumstances merely for reporting things to the experimenter.

The attackers will be given an indication by the experimenter of the type of behaviour that the owner would consider unacceptably disruptive. This should not deter attackers from attempting such breaches, since they will be rewarded accordingly in the points system (see above), but they must understand that they should not actually go ahead with disruptive actions.

Attackers participating in the experiment and obeying the rules will not incur any real-world penalties for their activities within the experiment.

Appendix B Experimental Rules

B.1 Rules for the Attackers

General rules

1. The attackers must have a complete understanding of the objectives of this experiment and must be willing to cooperate fully with the predefined experimental rules and with other demands which may be made by the experimenter during the experiment.

2. If, at any stage, an attacker is unsure whether he/she is complying with the experimental rules, he/she must immediately consult the experimenter.

Rules to Limit the Scope of the Experiment

3. All reports will be kept in confidence. After the experiment, no names should be associated with the reports.

4. The attackers must work independently of each other; no cooperation (or communication) between them is allowed.

5. The attackers must not make any attempt to monitor other attackers' (on-line or off-line) activity or reports or to break into other attackers' accounts.

Rules for Realism

6. Apart from the rules and demands made by the experimenter, the attackers must not let the existence of the experimenter influence their behaviour. They must not, for example, devise attacks in an effort to conceal them from the experimenter.

7. With the exception of a number of initiatives listed below, all possible help and resources are allowed, for example:

    - The attackers can get help in technical issues; they will be provided with the system documentation (e.g., technical and programming manuals).

    - Within our own economical constraints, the attackers will be able to borrow equipment (for example, PCs).

8. To motivate the attackers, they will be given points. The points system will reflect the value of breaches to the attacker as realistically as possible.

Rules to Prevent Unacceptable Disruption of System and other Unacceptable Activities

9. In certain cases, the experimenter has to determine whether actions that can have disastrous consequences for system operation or for the owners of the system should be performed. Whenever an attacker determines that a certain attack may fall into this category, he/she must contact the experimenter to verify whether the attack should be performed or not. Examples of such actions are:

    - The attackers must not perform any activity which physically harms any part of the system, e.g., cut wires, etc.
    - Since the system lacks protection against availability breaches and it is necessary to protect other users of the target system, these breaches must be treated with care. In general, availability breaches will be regarded as disruptive behaviour and will be forbidden. If an attacker believes that he/she could carry out an availability breach undetected, he/she should consult the experimenter to determine how far this activity should be allowed to proceed.
    - Active network attacks are not allowed unless permission has been granted by the experimenter since erroneous actions may affect many other computer systems on the network as well. However, if the attacker has enough knowledge, he/she will be allowed to continue with the attack.
    - Parallel execution of software on the machines is not allowed since it may cause an overload of the system.

10. The attackers are not allowed to get active help from the Internet network, for example by asking questions to news groups or other attackers on the network. This is to prevent people from other sites from attacking our systems.

11. For legal reasons, the attackers are not allowed to break into other systems in order to reach the target system, but exceptions may be granted by the experimenter with the agreement of the owner of this other system.

12. When attackers have gained super-user rights on the file-server (the ultimate goal where they can do anything they want with the system), they must immediately cease all activity and contact the experiment leader. The experiment leader will then explain the circumstances under which he/she can continue with the experiment. Violation of this rule may preclude further participation in the experiment.

Rules for Good Reporting

13. The attackers must complete a questionnaire relating to their prior experience and expertise before the experiment begins (see appendix A[4]).

---

4 Appendix of document handed out to the attackers. Appendix C in this document.

14. Throughout the experiment, the attackers must document (see appendix B[3]) all their actions, observations and motivations, and when (in real time) these take place. Note that when time is concerned, both on-line as well as time spent elsewhere must be recorded.

15. When a breach occurs, the attackers must complete a detailed report describing exactly when and how the breach was conducted (see appendix C).

16. The attackers should use the target system for as many activities as possible to facilitate monitoring, even in those instances where, in reality, due to the risk of detection by the owner, they probably would have chosen to use another system. When an attacker is using the target system only to satisfy this rule, this activity will be allowed even if the owner of the target system discovers it.

17. The attackers must be willing to cooperate after the experiment in order that the experimenter can question them about their activity. (For example, if data is incomplete etc.).

18. In order to facilitate system monitoring, the following should be considered:

    - Log out from the system when no activity is going on.
    - If two attacks are performed at the same time, use different windows for the attacks if possible.

B.2 Comments on Rules for the Attackers

Rule 3 was included to motivate the attackers to give accurate reports to the experimenter without any fear of retribution from the system owner, for example.

There were two distinct ways in which the experiment deviated from reality. Firstly, we voluntarily restricted the scope of the experiment by limiting the kinds of questions that we addressed (see section 2.4) - essentially to those that we believed were more easily addressed in such a preliminary investigation. Rules 4 and 5 essentially limit the scope of the experiment to addressing the issue of single non-cooperating attackers attacking a system, with no others attacking simultaneously.

Rules 6, 7 and 8 related to ensuring that they behaved, as much as possible, realistically within the bounds of the chosen scope of the experiment and within the other limitations, imposed for practical reasons, which are described below. Rule 7 was an attempt to encourage the attackers to use other resources which we may include as a factor in the effort variable. There are clearly gaps between the pilot set-up and reality. For example, the attackers were not given and were not able to borrow any money for their attacks. This means that any equipment they borrowed was not equivalent to expending some of their own resources. This also excluded certain kinds of attacks such as bribery of other users or system administrators.

The second way that the experiment deviated from reality was due to restrictions imposed upon the experiment for practical reasons. Rules 9 to 12 were applied to stop certain activities by the attackers which simply could not be allowed. There were two main types: those attacks which would have caused unacceptable disruption to our target system (which was in a real environment

with real users) or other systems, and certain other activities that would have ruined the experiment itself.

It was clearly crucial that the experimenter kept the attackers from doing activities which were unacceptably disruptive to the owner and/or users (see rule 9). The experimenter had to try to ensure that such activities were not done but, in the event that such activity did occur, the experimenter had to try to minimise the damage by responding immediately either to observation of that activity or to requests from the owner/users. As stated previously, it was important that the owner was able to distinguish, in such cases, between activity which was genuinely damaging or just potentially damaging. We wished the attackers to be able to carry out activity which was potentially damaging from the owner's viewpoint providing we could ensure that no actual damage to the system occurred. This was not meant to deter the attackers from attempting breaches in order to be able to potentially do such actions; they would be rewarded in the points scheme in accordance with potential disruption that they could have caused, but they had to understand that they should not actually go ahead with disruptive actions. Clearly, failure to prevent unacceptable disruption to the system may not only jeopardize this pilot experiment but may also eliminate the possibility of doing future experiments.

The attackers were given an indication of the type of behaviour that the owner would consider unacceptably disruptive. As previously stated, all three security aspects, confidentiality, integrity and availability are normally targets for attackers. In this experiment, we excluded the availability aspect. This was mainly for practical and economical reasons: the simplest way to disrupt the service from a system is to remove or to cut cables, to disconnect the power or to destroy the hardware and this was unacceptable in our target environment. Security can also be divided into three different forms (or areas), where each form represents a specific form of security: hardware security, information security and administration security. Hardware security is an area that deals with how to protect the system from physical threats, such as theft and physical damage. Again, for economical and practical reasons hardware security issues were excluded from our pilot experiment. Administration security depends largely on the local organisation maintaining security, but we believe that it was still useful to allow security breaches within this area as well as in the "traditional" area of information security.

Rule 10 was imposed since we did not want to let other people from other sites know about the experiment since this may provoke them into attacking the target system and give them information on how to do this.

Rule 11 was required since we wished to make sure that the attackers understood that they were not being given permission to attack any system other than our target system.

If an attacker gains super-user rights on the file server it is clear that, with these rights, they can do actions, or make observations, that will result in them being able to easily achieve future breaches with little effort. We did not want them to exploit such possibilities under these circumstances. If an attacker did this he/she would have effectively been lost from the experiment since no more useful data would have been obtained from that attacker. So, rule 12 was imposed so that instead of exploiting such possibilities, they stopped and retried new attacks. They had to understand, though, that they would be rewarded, in the points scheme, for having obtained super-user rights and for the potential actions they could have taken, in this position, even though we did not want them to actually take these actions.

Finally there were rules imposed to ensure that we got the required data from our experiment. Some of these rules (16 and 18) related to facilitating monitoring on the computer system by the experimenter. For rule 16 it was important that the attackers understood that they would not be penalised for following this rule, either by the owner stopping activity that would normally have been performed elsewhere, or in the proposed points scheme (i.e., no loss due to detection/risk of detection by the system owner). The remaining rules were so that the attackers gave complete reporting on their own actions, observations and motivations. The details of the contents of these reports are discussed in section 3.9 and appendices D.1 and D.2.

The attackers had to be made to understand that they would be penalised in the points scheme, the gratuities and prizes if these rules were not followed.

B.3 Rules for the Owner

General rules

1. The owner must have a complete understanding of the intentions of this experiment and must be willing to cooperate fully with the predefined experimental rules and with other demands which may be made by the experimenter during the experiment.

2. If, at any stage, the owner is unsure whether he is complying with the experimental rules, he must immediately consult the experimenter.

Rules to Limit the Scope of the Experiment

3. Apart from the inevitable interaction through normal monitoring of activity on the computer system the owner must avoid communicating directly with the attackers. All communication between them should when possible be done through the experimenter.

Rules for Realism

4. The owner must continue with his usual monitoring activities and other behaviour and must not let the fact that he knows the experiment is going on influence this or other behaviour. For example, the owner must only allow his monitoring of activity to be increased by observation of attacker behaviour which he actually makes while monitoring as usual.

5. If the owner sees an activity that he would normally decide to stop, he must do so. He should preferably notify the experimenter who will contact the attacker, but if necessary he may stop the activity himself and notify the experimenter as soon as possible.

6. The owner is not allowed to modify the system as a result of observed attacker behaviour. If he sees an activity that would normally prompt a system change to prevent future such activity, he should only prevent continuation of such activity from that attacker.

Rules for Good Reporting

7. The owner must document all his relevant (to security and observed attacking behaviour) actions, observations and motivations and when (in real time) these take place. This includes

    - o When the owner decides to stop an activity, he must document the reasons for this action together with a description of the mechanisms used.

    - o In addition to this, if the owner normally would have changed the system in order to prevent attacker activity from being repeated, he must document how and when this modification would have been done.

8. The owner must make this documentation available to the experimenter throughout the experiment.

9. The owner must be willing to cooperate after the experiment in order that the experimenter can question him, for example, about actions he would have taken or loss due to attacker behaviour that he did not observe during the experiment.

B.4 Comments on Rules for the Owner

Rule 3 was imposed to limit the scope of the experiment to attackers with no interaction: it seemed quite possible that communication with the system owner may have caused a second order interaction between the attackers.

Rules 4 to 6 were basically just stating that the system owner should behave as he normally would, for example stopping attacker activity he sees as he normally would and so on. An exception to this was rule 6, where it is stated that the owner must not make changes to the system as he normally would in response to attacker activity. This was in order that a change made in response to one attacker did not affect the possibilities for any other attackers: we wished to limit the scope of the experiment to single attackers with no others attacking simultaneously. In general we did not want the owner to let any one attacker's activity influence his responses to any other attacker. We have listed rule 6 under rules for realism since, although it is a departure from realism for the owner's behaviour, it is imposed in order to that we have realistic behaviour of each single attacker.

Note that if there were changes that the owner would have made in response to attacker activity, we wished to record such changes and possibly modify the data from that attacker after the experiment in order to simulate the situation where the change had been made. The points scheme should also reflect the situation that such a change had been made by the owner for that attacker. Since the owner activity is fairly low level we did not envisage that this situation would arise very frequently.

The remaining rules related to good reporting by the owner. Unlike the attackers, the owner was not actually filling out reporting forms but we wished accurate reporting from the owner, the details of which are discussed in section 3.10.

Appendix C Reporting

C.1 Attacker Background Report

1. Current date and time: Attacker ID:

2. Education/ percentage completed?

3. Estimate number of computer-related courses taken?

4. Have you taken:

Operating systems?	Yes	No
Unix internals?	Yes	No
Distributed systems?	Yes	No
Cryptography?	Yes	No
Data communication?	Yes	No

5. Have you been working professionally with computers (i.e., not related to school)? Yes No

   Where? When? For how long? Your major tasks?

   With Unix or with security issues?

6. Are you using computers on your spare time? Yes No

   Estimated number of hours a week?

   What kinds of system(s) are you using?

   What do you mostly do?

7. Do you own a computer system? Yes No
What kind? What OS?

8. What are your motives for doing these experiments? Low Average High
the gratuity
the prize
personal interest
scientific results
the challenge
for fun
lack of other projects
have spare time
other reasons (specify!)

9. When did you first hear about this experiment?
How much did you know 3 days ago?

10. Have you ever been interested in security issues? Yes No
Why? What have you been doing?
Have you ever tried to break into a computer system before?
When? How (what method)? How hard did you try?
How much time did you spend? Did you succeed?

11. What possible ways of breaking into the system can you think of right now?

12. Other relevant information (for example additional experience or skill in computer security):

C.2 Attack Report

Report to be completed before, during and after each attack:

Fill out before the attack has started:

1. Report number (ATTACK_id_*nnn*):

2. Date and current time:

3. Name of attack:

4. Is this a continuation of an old attack or is it in any way related to older attacks?
   old new
   Give report number(s) and activity(s) for old attack:
   How are they related?

5. Description (of proposed) method (if new attack)?

6. What do you think the consequences (rewards or damages) of this attack will be to:

   a) you

   b) the system owner

   c) other users?

7. Might a success (i.e. a breach) lead to further breaches? How?

8. How big is the risk of being detected? None Small Big
   Can the attack be traced back to you? Yes No

9. How much working time are you willing to spend on this attack?

   How much effort do you think is needed?

10. How do you estimate the chances for a success?
    Very small Small Fair Large

11. How did you get the idea for this attack? When?

12. How much time and effort have you spent in planning this attack? Enter preparation time in the time report!

Fill out during the attack :

Give separate activity numbers for on- and off-line time. Number the activities and give details below!

13. Time report (time period is time when you or a computer works with the problem):

Time period	Est. effective working time	On-line time (user @ host)	CPU time used (hh:mm:ss)	Activity number
-----	----------	----------	----------	------

14. Details about activities: For each activity, tell what was done and effort not included above.

15. If an activity was based on results from other attacks, specify activity and report numbers:

Fill out after the attack :

16. Current time and date:

17. Time and date when the attack was stopped:

18. Why did you stop the attack?

19. Did the attack end with a breach? If so, during which activity did it occur? What is the breach report number(s).

20. What did you gain from the attack except from a possible breach?

21. a) Which activities could have been detected (i.e. may have looked suspicious)?

    b) Which could be traced back to you?

22. a) Which activities do you know was detected? (If not by the system owner, by whom?)

    b) Which activities were stopped? (If not by the system owner, by whom?)

23. Which activities would normally not have been executed on the target system if this would have been a real attack?

24. What software was used? Did you write it? If not, how and from where did you get it?

25. Did you use any additional hardware? What?

26. Did you use any other resources (for example books, documentation, money, etc.)?

27. Have you been working on (or thinking about) other attacks or has CPU time been used for other attacks during any of the activities above? Which? If so, give the report numbers for the other attack(s)?

28. What other relevant information can you think of, for example with regard to effort? To other things?

29. Do you have any future ideas about how to attack the system (a future attack plan):

C.3 Breach Report

Report to filled in <u>immediately</u> after a successful breach:

1. Report number (BREACH_id_*nnn*):

2. Date and time when breach occurred:

3. During which attack and activity did the breach occur (give report and activity number)?

4. Was other breaches necessary to achieve this breach? Which?

5. Was this breach expected? Yes No
   Was it the primary goal for the attack? Yes No
   Was this breach detected? Yes No Don't know

6. What are the consequences from this breach for you? What did you gain?

7. What are (the possible) consequences for other users?

8. What are (the possible) consequences for the system owner?

9. Can this breach facilitate further breaches? How?

10. Details about the nature and how the attack was performed (so it can be repeated):

11. Do you have other relevant information?

C.4 List of Attacks Report

This form should be kept by the attacker during the experiment:

Attacker ID: **Date:**

Attack number:	**Description:**	**Start date:**	**Start time:**
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			

C.5 Attacker Evaluation Report. Completed after the Experiment

Report to be completed after the experiment is over

Date: Attacker id (voluntary):

Introduction to the Experiment

1. What do you think about:

   a). the theoretical (English) introduction?

   very good OK bad

   comments: _____

   b). the practical (Swedish) introduction?

   very good OK bad

   comments: _____

   c). the written instructions?

   very good OK bad

   comments: _____

   d). Other comments?

The Reports

2. What do you think about:

   a). the attack reports (before the attack)?

   very good OK bad

   comments: _____

   b). the attack reports (after the attack)?

   very good OK bad

   comments: _____

   c). the breach report?

   very good OK bad

   comments: _____

   d). the reporting system using e-mail?

   very good OK bad

   comments: _____

   e). Other comments?

3. What question/questions were hardest (or impossible) to answer?

4. What do you think about the reward system?

   How does it affect the experiment?

   In what way was the time and effort you actually spent affected?

The Attacks

5. During how many attacks were you helped by other attackers? What attacks?

6. How many of your attacks were unknown to you when you started the experiment (give a number "m of n" or a percentage)?

7. How many and what (kind of) attacks were known?

8. What activities were never reported (for example investigations or searches for something)? Why?

   Estimate the number of hours spent on such tasks!

9. Would you prefer to work in a team instead (for example in groups of two)? Why?

   Would it affect the total time/effort you would spend on the experiment?

10. How do you think it is possible to motivate an attacker to spend more time before giving up?

11. What is your overall impression of the experiment? What can we do to make it better next time?

12. Other comments about the experiment:

Appendix D Detailed Comments on Reporting Forms

D.1 Comments on Attack Reports (Appendix C.2)

The first part of the attack report relates to the attacker's beliefs before he/she has begun each attack, and to make sure that this is genuine, it was required that this part of the report was sent to the experimenter before the attack in question had begun. Much of this report (questions 6, 7, 8 and 10) covers different aspects of what they think the attack will achieve (i.e.,. the potential rewards) and should reflect the driving force behind the attack. Question 9, in some sense, should be a reflection of the attacker's total value which he/she attaches to this attack which includes his/her perceived chances of success (i.e., a breach) together with what value he/she attaches to such success. In addition question 12 asks about effort that they have already spent off-line in preparing for this attack. This is because we want to include this in our effort measure and it was anticipated that they would inevitably have spent some time thinking about each attack before they fill in part 1 of the attack report. We also wished to know (see question 4) whether this is a completely new attack or whether it is a continuation of an attack previously tried and for some reason terminated.

The second part of the attack report, to be filled in while the attack was being done, mainly relates to the effort expended by the attacker during the attack. Most of the information required relating to effort should be contained in the time report (question 13) which should give on- and off-line time spent by the attacker on this attack; it was thus crucial that the attackers gave accurate and complete time reports. In order to complete this time report the attackers were asked to divide the attack into distinct activities. Apart from the requirement that the attackers separate activities by on- and off-line time it was a matter of their choice as to how the attackers made this division of each attack into activities. There is an issue of granularity here (which also exists for how the attackers decide to divide their attacking actions into distinct attacks and so fill in an attack report for each) and for the data we require, it is desirable to have the finest granularity possible for divisions into activities; providing any resources used can be connected with the appropriate windows of effective working time from the time report, the contribution to the effort measure can be calculated.

The final part of the attack report was to be filled in immediately after the attack had finished, and the completed report sent to the experimenter. Presumably the attack will terminate due to either a breach being achieved, the attacker giving up, or the attacker being stopped by the system owner or the experimenter and questions 18, 19, 20 and 22 may cover these points. Questions 20, 21, 22 and 29 relate to rewards that the attacker may associate with the attacking activity but for which they may not have actually reported breach events. This is an attempt to get empirical evidence on continuous rewards achieved, for example via learning, or any other rewards which have not been reported as a breach event by the attacker but for which they do actually have some subjective reward attached. Question 23 is required so that the experimenter can simulate the situation where the attacker is not actually using the target system when he/she would choose not to. For example, the experimenter did not want to allow the owner to stop activity in these circumstances and also the risk of detection in these circumstances should not be reflected in the points scheme. Questions 24 to 26 were intended to cover those factors of the effort measure, for example resources other than time used, which were not included in part 2 of the report. Question 27 relates to simultaneous activity from another attack since we need to know if more than one

attack is going on at the same time; appropriate adjustments may have to be made to the time variable contribution to the effort measure if this is shared amongst different attacks.

D.2 Comments on Breach Reports (Appendix C.3)

We were interested in knowing (see question 4) whether each breach was the result of a succession of partial breaches or not. Questions 5 to 9 relate to the attacker's subjective view of the reward associated with each breach that they achieved. Question 10 was included so that the experimenter can easily get an overview of how each breach was done.

So that the attackers could fill in questions 4 and 15 of the attack reports (see appendix C.2) and also as a double check on their activity for the experimenter, it was useful for each attacker to keep a record of all their attacks during the experiment (see appendix C.4).

Appendix E Accounting and System Logs

The system which was used in the pilot experiments, SunOS, has several data collecting facilities which are intended to be used when doing user accounting. In addition to this, the system records when users log in and out from the system and when unusual events such as failed login attempts, disks being filled up and unexpected network problems, occur etc. These information "databases" can be used for data collection in the experiments. The following information is available:

The Command History Database (/var/adm/pacct):

Command	User	tty	CPU time	Real time	Memory	I/O chars	I/O blocks	Flags	When
man	tomas	ttyp1	3.15	17.42	2054	123	0		Jan 15 11:08:36
ditroff	tomas	ttyp1	0.77	6.33	7264	5403	12		Jan 15 11:09:49
finger	sync	-	0.23	1.42	1412	4041	0		Jan 15 11:11:34
in.finge	sync	-	0.12	1.92	562	1311	0	S	Jan 15 11:11:34
cpp	tomas	ttyp2	0.20	0.87	1662	5891	1		Jan 15 14:51:17
ccom	tomas	ttyp2	0.35	0.63	3902	1778	1		Jan 15 14:51:18
as	tomas	ttyp2	0.25	0.67	2805	1337	5		Jan 15 14:51:19
ld	tomas	ttyp2	0.27	1.28	2845	1684	7		Jan 15 14:51:19
compile	tomas	ttyp2	0.13	4.08	979	256	0		Jan 15 14:51:17
a.out	tomas	ttyp2	1.20	1:40.72	6100	1059	0		Jan 15 14:53:03
more	tomas	ttyp2	0.60	1:43.23	1422	1526	0		Jan 15 14:53:03
vi	tomas	ttyp2	3.55	52:06.40	8976	10492	28		Jan 15 14:20:42
breach	*tomas*	*ttyp2*	*0.08*	*0.80*	*467*	*8912*	*1*		*Jan 15 14:55:03*
sendmail	root	-	0.05	0.13	198	1024	0	F	Jan 15 14:43:11

This file contains one record for each command (i.e. program) being executed. It contains a summary of CPU, I/O and memory usage for each command together with a time-stamp when the command was issued. For example, the line beginning with "cpp" until the line "vi" shows a session where a user (tomas) had been using the "vi" editor, compiled a program (its name is not recorded), executed it together with "more" and finally terminated the editor.

The User History Database (/etc/utmp):

User	tty	From	When
tomas	ttyp3	triffid	Thu Jan 14 21:01 - 21:52 (00:51)
tomas	ttyp3	triffid	Thu Jan 14 20:59 - 20:59 (00:00)
jones	ftp	ada.sisu.se	Thu Jan 14 20:29 - 20:38 (00:08)
tomas	:0		Thu Jan 14 15:37 still logged in

This file contains records about when each user logs in and out from the system. Together with the command history database, it is possible to collect information about user sessions and to detect when commands are executed automatically after users have logged out from the system. (When two parallel activities are going on, it can be difficult to separate them from each other).

The System Error Log (/var/adm/messages):

```
Jan 13 12:29:46 triffid su: 'su openwin' failed for tomas on /dev/ttyp2
Jan 13 17:32:33 triffid login: ROOT LOGIN ttyp4 FROM lilja
Jan 14 00:39:20 triffid vmunix: le0: Receive: giant packet from 55:55:55:55:55:55
Jan 14 15:56:55 triffid yppasswdd[143]: john: password incorrect
Jan 14 12:29:46 triffid login: REPEATED LOGIN FAILURES ON ttyp1 FROM 192.70.3.86, dg
```

This file contains error messages from various programs in the system, for example the login program when it detects repeated login failures or when other anomalies are found in the system. This file was monitored during the experiments to give us as much information as possible about our attackers' actions.