On Understanding Normal Protocol Behaviour to Detect the Abnormal

P. D. Smith and D. Hutchison
Computing Department, InfoLab21
Lancaster University
Lancaster, UK, LA1 4WA
{p.smith, dh}@comp.lancs.ac.uk

M. Banfield and H. Leopold
Telekom Austria AG
Lassallestraße 9
A-1020, Vienna, Austria
{mark.banfield, helmut.leopold}@telekom.at

***Abstract*— A network protocol can exhibit abnormal behaviour as a consequence of an attack and still behave according to its functional specification. For example, a protocol engine could be transmitting well-formed messages in the order defined in its specification, but at a significantly greater than expected rate. Abnormal behaviour could occur because a protocol is being used to directly orchestrate an attack (for example, a TCP SYN attack) or as a byproduct of one (for example, the ARP protocol during the propagation of malware). In this abstract, we describe a new research initiative that will investigate how to detect this abnormal protocol behaviour caused by network attacks.**

We believe that an understanding of a protocol's normal operational behaviour, in addition to its functional specification, can be an essential tool when designing, implementing, and specifying parameters for network attack monitoring and mitigation systems. Here, we present an overview of some of the issues related to specifying the normal behaviour of a protocol and suggest how one could be used, for example, with hooks into end-system network stacks, to identify anomalous behaviour.

I. Introduction

Network protocols can be used to facilitate attacks on networked services. One way of carrying out an attack is to manipulate the functional behaviour of a protocol. For example, have a protocol not behave to specification by sending incorrect messages that are out of order or malformed. A TCP Christmas Tree scan is a good example of this, where malformed TCP packets [1] are used to discover open ports on end-systems. In a number of cases, network protocols can be used in attacks and still behave to specification. An example where a protocol is behaving to specification and is being used, indirectly in this case, in an attack is the Address Resolution Protocol (ARP) during the dissemination of malware, such as network worms [1].

For malware to propagate they must discover new vulnerable hosts. The algorithms used to disseminate malware often bias their search for new targets to the same network as the currently infected host. The inference is that other hosts on the network will be under the same administrative control and will therefore be vulnerable in the same way. Consequently, hosts that are infected perform aggressive scans of the local network. A byproduct of these scans can be significant quantities of ARP broadcast traffic caused by a host trying to determine the link-layer address of hosts it intends to port scan for vulnerabilities.

An analysis of the ARP protocol under such conditions is presented in [2]. The analysis shows that infected hosts generate traffic at an alarmingly high average rate (approximately 1639 requests per second), sequentially scan a network's address range, and can potentially cause a significant degradation of service on an affected network. This is not desirable behaviour, yet the ARP protocol is acting according to specification.

Also, in [2] a relatively simple way to control this problem is proposed that involves network switches probabilistically dropping ARP requests from misbehaving hosts. Simulations of the scheme using real ARP traces highlighted non-trivial problems that were associated with the operational behaviour of the ARP protocol. It was found that a network's gateway router can generate a relatively large amount of ARP traffic (in comparison to end-systems), and could be adversely affected by the dropping mechanism if additional measures were not put in place.

To summarise, in the context of network attacks, anomalous behaviour of network protocols can occur because they are being used directly in an attack (for example, a TCP SYN attack [3]) or as a byproduct of an attack (for example, ARP behaviour during malware propagation). There are three main ways in which anomalous behaviour of network protocols can manifest – stochastically (for example, abnormal rates of protocol messages being transmitted), the distribution of values in protocol messages (for example, sequential address scans), and incorrect protocol operation.

A protocol's functional specification (for example, for an Internet protocol its RFC) can be used as a guideline when designing and implementing systems that monitor and control incorrect operation. We propose that a specification of a protocol's normal operational behaviour, in addition to its functional specification, could be an essential tool when designing and parameterising systems, such as the dropping mechanism described earlier, that monitor and control a protocol's non-functional anomalous behaviour.

[1]In a TCP Christmas Tree Scan, messages are sent to all ports on a host with all flags set (*FIN*, *SYN*, *RST*, *PSH*, *ACK*, and *URG*) in order to elicit a *RST* message for every closed port. Consequently, the open ports on a host can be determined.

II. Specifying the Operational Behaviour of Network Protocols

The first step in the process of describing a protocol's normal behaviour, shown in Fig. 1, is to understand its operational context. This can be done during the conception of a new protocol or retrospectively for one that is currently specified. A number of techniques could be used to understand a protocol's operational context – for example, combinations of modelling, simulation, or observation on testbeds or operational networks could be used. The ideal approach is to observe a protocol on an operational network with real usage patterns. However, this may not be possible for technical or organisational reasons.

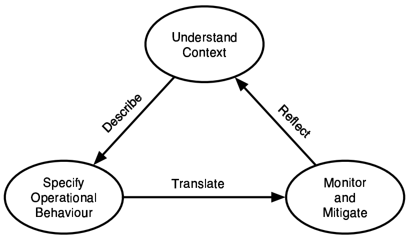

Fig. 1. The processes involved in specifying the operational behaviour of network protocols

It will be important that a protocol is observed from a number of different *viewpoints*. A meaningful viewpoint is a point of observation where different protocol operation can be seen if the *view* is changed. As mentioned in the ARP example earlier, a network's gateway router under normal conditions generates significantly more request traffic than a standard host, which lead to problems with the dropping mechanism. Here, the viewpoint is the device type and the set of views is the different devices. Other viewpoints could include the time of day, the topological location of a device, or the underlying link type. An important consideration for this research is to understand what are the meaningful viewpoints.

Having gained an understanding of the operational context of a protocol, the next step is to describe it. We envisage that a *base specification* of a protocol's behaviour may in the first instance be defined by standardisation bodies, for example. The base specification will define the relevant viewpoints and associated views. It will also define the *features* that make up a view (such as message rate or relevant protocol message fields) that can be used to identify normal behaviour. The base specification may define expected values for features, or describe functions that can be used to determine values. For example, a function could be specified that yields a normal ARP request rate for a network based upon its size. The relationship between viewpoints, views, and features are demonstrated in Fig. 2, with an example for the ARP protocol.

It should be possible to tailor a base specification. Organisations may, based upon policies, wish to define local values for different features, or change the set of viewpoints and views in a way that is relevant to their local context. Furthermore, equipment vendors may wish to extend or adjust a base specification for a protocol based upon the capabilities of their devices. This might, for example, be useful if a new product becomes available that can handle larger data rates or enable traffic monitoring in a way that was previously not possible.

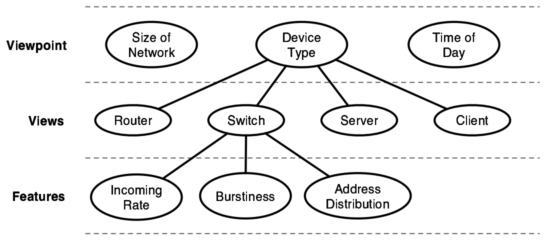

Fig. 2. An example set of viewpoints, views, and features for the ARP protocol

Another way in which a base specification could be extended is through the definition of thresholds for feature values. These thresholds could be used to define the severity associated with deviating from normal behaviour, which could be used to influence the mitigation strategies for a misbehaving protocol. For example, if a set of hosts are sending ARP requests just above the normal rate, an event could be triggered to notify a network administrator. However, if the rate is significantly higher than the norm, such that a degradation of service could occur, the offending hosts could be blocked at their nearest managed switch.

We believe there are a number of ways a specification of normal protocol behaviour could be used for network attack monitoring and mitigation. A clear specification of a protocol's behaviour from a number of viewpoints and views could be used to aid the design and the specification of parameters for novel network attack monitoring and mitigation systems.

Furthermore, we will investigate potential specification formats that are machine readable, which could be compiled into rules for firewalls or packet filters for monitoring systems. For example, a monitoring system developed on the LARA++ programmable edge router [4] could use an operational specification. LARA++ enables packet filters to be registered by active components and ensures packets that match filters are delivered to a component. In this scenario, the types of packets that give a measure of normal behaviour, which are described in a specification, could be filtered to active components that check features such as protocol field values or message rates. If abnormal behaviour is detected, appropriate action could be taken using other active components, for example.

It may also be possible for elements of a specification to hook into modified protocol stacks on end-systems that monitor and control the behaviour of network protocols. The aim would be to enable systems to determine when they are sending messages as a consequence of malicious software (in other words, determine which messages are bad) and perform some form of local control of a protocol's behaviour. For example, it should be possible for a system to determine that it is conducting a sequential scan of a network and locally drop messages that are part of that scan.

One way of implementing such a scheme would be to introduce *aspects* [5] into the protocol stack of end-systems. Aspect-oriented programming techniques allow program segments that address *cross-cutting concerns* (in this case, protocol behaviour monitoring) to be *woven* into parts of a program that implement its core functionality (here, the implementation of the protocol engine). The operational specification of a protocol could be used to provide arguments for aspects that monitor its operation.

For a number of reasons, it should be possible for suitably privileged entities (machine or human) to modify a specification of normal behaviour. For example, new devices, applications, or associated protocols may change the normal operational behaviour described in a specification. Also, a description may be ill-specified from the outset, causing inappropriate control and mitigation actions to be taken. We will investigate how such problems can be discovered and how the necessary changes to a protocol's specification can be modified in an automatic way.

III. Discussion

This abstract describes a new research initiative; there are a number of open issues.

It is not clear at which protocol levels a specification of normal behaviour could be most useful. For example, is the specification of overlay protocol behaviour the most suitable way to detect and mitigate attacks caused by overlay-based applications, or is it sufficient to capture the normal behaviour of lower-level protocols to address these forms of attack? A specification of a low-level protocol could be used to detect the anomalous behaviour of a number of higher-level entities, for example, or vice-versa.

Capturing and specifying the *complete* behaviour of network protocols is a potentially fruitless task. For example, consider capturing the complete behaviour of the IP protocol from every viewpoint and view – this would be practically impossible and probably not very useful. Understanding if there is a minimum set of protocols, viewpoints, and views that are useful for attack detection will be a key contribution of this research. In other words, we will investigate which protocols exhibit abnormal behaviour and in what way when attacks occur.

To what extent stealth attacks, those that attempt to hide their existence, can be identified using a technique such as this will need to be investigated. It is probable that this will depend upon the sophistication of the attack and the fidelity of the protocol behaviour specification. For example, a precise specification could give a stealth attack less room to manoeuvre. However, such a specification could cause a larger number of false positives. A trade-off will need to be made.

There may be a number of side effects associated with detecting anomalous behaviour in this way. For example, it is not clear to what extent this approach could generate false positives and what the subsequent consequences associated with this are. We believe that a system that uses this approach to detect anomalous behaviour should use machine learning techniques to detect and mitigate such side effects. This research is being carried out as part of a larger research initiative that is taking a complete systems approach to building resilient networks [6].

IV. Conclusion

Protocols can exhibit abnormal behaviour as a direct or indirect consequence of network attacks and still behave to their functional specification. We propose that an understanding of the normal non-functional (or operational) behaviour of a protocol could be used to detect such abnormal behaviour and therefore the activity of an attack. For example, a specification of normal protocol behaviour could be used to specify parameters for monitoring systems, such as modified end-system network stacks that perform stochastic and protocol field value distribution checking, to detect anomalous behaviour.

Specifying protocol behaviour is non-trivial and in this abstract we have discussed some of the issues associated with this. As was demonstrated by the example ARP dropping mechanism and its effect on a network's gateway router, it is essential to specify a protocol from a number of different viewpoints. We will investigate what are the meaningful set of viewpoints and which protocols best illuminate attack activity. A key contribution of this work will be a machine readable format that can be used to describe a protocol's behaviour. We will investigate how such a format could be used in network attack and mitigation systems.

V. Acknowledgements

Paul Smith is supported by Telekom Austria AG. The authors are grateful to Linlin Xie and James P.G. Sterbenz for discussions on resilient networking concepts that have contributed to this abstract. Furthermore, we acknowledge the work of Dadi Ármannsson, Gisli Hjálmtýsson, and Laurent Mathy on controlling the effects of anomalous ARP behaviour, which has provided an invaluable case study for this new line of research.

References

[1] D. M. Kienzle and M. C. Elder. Recent Worms: a Survey and Trends. In proceedings of *2003 ACM Workshop on Rapid Malcode (WORM 2003)*, October, 2003, Washington DC, USA, pp. 1–10.

[2] D. Ármannsson, P. D. Smith, G. Hjálmtýsson, and L. Mathy. Controlling the Effects of Anomalous ARP Behaviour on Ethernet Networks. In proceedings of *2005 ACM Conference on Emerging Network Experiment and Technology (CoNEXT 2005)*, October, 2005, Toulouse, France, pp. 50–60.

[3] H. Wang, D. Zhang and K. G. Shin. Detecting SYN Flooding Attacks. In proceedings of *21st Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 2002)*, June, 2002, New York, USA, pp. 1530–1539.

[4] S. Schmid, T. Chart, M. Sifalakis, and A.C. Scott. A Highly Flexible Service Composition Framework for Real-life Networks. *Journal of Computer Networks, Special Issue on Active Networks*, Elsevier, Volume 50, Issue 14, pp. 2488–2505, October 2006.

[5] T. Elrad, R. E. Filman, and A. Bader. Aspect-oriented Programming: Introduction. In *Communications of the ACM* Volume 44, Issue 10, pp. 29–32, October, 2001.

[6] The Resilient Networking Initiative at Kansas and Lancaster Universities http://www.comp.lancs.ac.uk/resilinets