Australian Government
Department of Defence
Science and Technology

Covert Channels over Network Traffic: Methods, Metrics, and Mitigations

Patrick Prendergast

Cyber and Electronic Division
Defence Science and Technology Group

DST-Group-TN-1695

ABSTRACT

This technical note is a review of current open literature regarding covert timing channels over network traffic. The paper consists of a brief background of timing channels, some examples of covert timing channels over network traffic, metrics to define the capacity and probability of a channel, and finally methods to mitigate timing channels. This technical note is intended to provide a basis for analysing and reducing potential timing channels when assessing and designing system architectures.

Produced by

Cyber and Electronic Warfare Division
Defence Science and Technology Group
PO Box 1500
Edinburgh SA 5111

Telephone: 1300 333 362

October 2017
AR-017-008

Conditions of Release and Disposal

Covert Timing Channels over Network Traffic: Methods, Metrics and Mitigations

Executive Summary

This technical note is a study of covert timing channels over computer networks.

Steganography, the art of hiding covert messages in plain sight, is by no means a new concept, and is one that certainly applies to network traffic. Messages can be hidden within files, images, or unused fields of network packets. Data crossing domain boundaries must be scrubbed for any potentially classified or hidden information before release, however information can also be transmitted by the timing of the messages themselves. With the increasing inter-connectedness of computer systems, there is a growing need for systems to communicate across domain boundaries, making timing channels an increased threat.

This report covers the common covert timing channels, as well as ways to define and detect their presence, and finally approaches to mitigate them, based on a review of the open literature. These methods will allow systems to be designed to reduce the presence of potential timing channels, and reduce the risk of data leakage over such channels.

This page is intentionally blank.

Contents

1. INTRODUCTION 1

2. HISTORY AND FOUNDATIONS 1

3. EXAMPLES OF COVERT CHANNELS 2

4. METRICS 3
   4.1 Capacity 3
   4.2 Detection 4

5. MITIGATIONS 5
   5.1 Rate Limiting 6
   5.2 Quantization 6
   5.3 Noise Generation / Random Delay 6
   5.4 Packet Generation 7

6. DIRECTIONS 7

7. ACKNOWLEDGMENTS 7

8. REFERENCES 8

APPENDIX A TIMING CHANNEL TEST BED 10

This page is intentionally blank.

1. Introduction

In 1973, Lampson first noted the risk of covert channels being used to transfer data between confined services on a shared computer system[1]. By 1987, Girling defined the distinction between storage channels, which manipulated the potentially unused information in network protocols, and timing channels, which use the delays between network traffic to transfer information[2]. In the proceeding 30 years this work has been built upon to include several different methods for creating, detecting and mitigating such channels.

The following survey covers work in the timing channel domain, going into particular depth on the papers which we believe are most pivotal to the topic of timing channels. However we also cover work which gives context to the research. The first section covers a brief history of covert timing channels, next we discuss specific methods for transmitting information through a timing channel, metrics and detection methods, counter measures and finally we discuss the current directions and potential future research in the field.

2. History and Foundations

It's impossible to discuss any work on covert channels without acknowledging Shannon's foundational work describing what is now known as information theory. In *A Mathematical Theory of Communication*, first published in 1949, Shannon defines two of the core concepts which are often repeated in the literature: Shannon entropy and Shannon capacity[3].

Lampson's observation that information within a shared system could be difficult to confine due to the ability to manipulate shared resources on a single system as well as the timing of certain events began the academic discussion on covert channels[1]. Lipner further examined this problem, noting the difficulty of a practical counter measure for timing-based covert channels[4]. Padlipsky, et al. argued that covert channels presented a limitation to the security guarantees of encrypted network communication, allowing information to be encoded in the message length and address space metadata of a message, but also the timing of message delays[5]. Although the use of end-to-end encryption has proven significantly more useful than Padlipsky, et al. would suggest, the issue of covert channels remains and continues to undermine efforts to implement high assurance and multi-level secure systems over untrusted components. Indeed, Moskowitz and Kang reasonably argue that covert channels can never be fully eliminated without significant operational costs[6].

Although the distinction is used previously in relation to program confinement[7], Girling describes two types of covert channels over network packets; storage channels and timing channels[2]. The former is any covert channel which is hidden within the protocol data and metadata, while the latter describes information transferred using the timing between packets. Although early papers have various definitions of what exactly a covert channel consists of, we will use this definition for the remainder of the discussion.

3. Examples of Covert Channels

The timing of packet transmissions gives us two factors which can be used for encoding data: the transmission (or lack of transmission) of a packet within a given time window and the relative delay between packets. For our purposes we will call these absolute and relative timing mechanisms, respectively.

The classic example of a timing channel is the presence or absence of some flagging mechanism within a given timeframe. Lampson originally suggests a mechanism which involves manipulating the locks of a shared resource on a single machine[1], and the essential concept of what we will call absolute timing channels is particularly popular in low latency applications[8-10], as well as some network cases[11, 12], where the delivery or lack of a packet is used as a flag.

Shah, et al. introduces a class of timing channels known as JitterBugs, with the popular Keyboard JitterBug example[13]. The method involves first determining a timing window which is used to represent a symbol space (the simplest being a binary symbol space of [0,1]). For example a 30 ms timing window may consider 0-14 ms a 0 bit with 15-30 ms a 1 bit. A delay is then added to the next sent packet such that the resulting delay between packets modulo the timing window will be within the appropriate range. This method is particularly effective on sparse traffic such as traffic generated by user key presses, adding only a small additional delay compared to the natural delay between packets. However, when the channel is saturated the small additional inter-packet delay can significantly affect the throughput of the legitimate traffic.

Sellke, et al. generalised the approach to inter-packet delay based timing channels with their L-bits to n-packets scheme[14]. The scheme improves the ability to calculate the best encoding trade-off for various uses over a network transmission case (average delay, information transmission rate, etc.). They also suggest the use of a cryptographically secure pseudo random number generator with a shared seed between sender and receiver as a masking technique. Alternatively, by determining the cumulative distribution function (CDF) of normal traffic, the sender can generate a mask which will increase resistance to detection based on traffic distribution tests. Although the authors seem to trivialise this process and start with a CDF model as given.

In keeping with the approach of matching covert channel traffic to a statistical model to evade detection, Gianvecchio, et, al. suggest a model-based approach to creating covert timing channels over networks[15]. In their approach legitimate traffic is matched to its closest statistical model which is then used to mask the covert traffic. While the authors claim that the modelling stage adds little overhead, they do require MATLAB for their implementation, implying that they required non-trivial statistical mechanisms for the modelling stage. There's also little detail on how the receiver is implemented (if at all), and the comparison algorithms are designed to maximise capacity over stealth. That said, their approach did fairly well against their chosen tests for regularity and distribution.

Because network traffic usually presents as quite sparse, the majority of network based timing channels are designed to use the relative delay between channels. However, if we

consider that the definition of a network can include intra-device communications[16] or low latency networks used to build composite systems, we must still consider the case of large throughput, low latency, and largely automated (as opposed to user generated) network traffic.

The example "worst case" set out by Gray and James, describes a channel that can periodically produce a large volume of requests, or none over a given interval to represent binary channel[8]. While it's certainly possible that a Trojan on the high side could produce requests, this would make it a fairly loud channel, and the same effect could be achieved by simply jamming or partially jamming the legitimate channel for brief periods. Such methods, which are traditionally a threat for low-level, single machine use cases[17], could present a high bandwidth timing channel where high-volume and low latency connections exist. These channels would avoid the costly delay associated with inter-packet methods.

4. Metrics

Covert timing channels are difficult and expensive to eliminate completely, so there is a strong requirement for metrics which allow us to detect and define such channels. Defining the capacity of a channel allows us to know the maximum data which can be exfiltrated via a covert channel, while the ability to detect the signs of a channel gives us a way to gauge the probability that a channel is active.

4.1 Capacity

A channel is defined as either being continuous (e.g. analogue), or discrete (e.g. digital). Although the time of network packets from a source can be viewed as a continuous channel [3, 18], the minimum precision of the clock of the sender or receiver will determine the interval of a discrete channel over which information can be sent[19], and can be further guaranteed by design[20]. For this reason we consider the timing channel to be discrete for all practical implementations; however it is important to capture the correct granularity. For example if our clock precision is 1 ms, our channel capacity would be 1000 bits/second. The Shannon capacity considers the noise in the channel as a factor, for example we may look at the entropy in the delay for a network channel, by measuring the variance in delays on a given network path between legitimate sender and receiver. However we must assume that the covert receiver may sit between our legitimate end points, allowing them to falsely create the appearance of noise. Therefore any noise measured between legitimate sender and receiver can't be fully trusted, and the capacity of a noiseless, discrete channel should be seen as the ceiling of channel capacity and minimised during design time.

4.2 Detection

Since the total elimination of covert timing channels is considered impractical[6], with any mitigations coming at a cost to the functionality of the legitimate channel, the detection of covert timing channels has become a large focus of study. In order to receive the covert timing channel it must be distinguishable from noise, no matter how well hidden. However general solutions to the detection problem are difficult.

One way to detect a channel is to look at patterns in the regularity of packet delay. Since natural variation in packet delays are due to noise, regularity in the noise would be indicative of a channel. Cabuk, et al. propose two measures of regularity based on the consistency of packet inter-arrival times and ε-Similarity[11]. The first method is based on the standard deviation of inter-arrival variance, with lower variance indicating a possible covert channel, but also a saturated bandwidth. ε-Similarity, looks at the similarity between sorted inter-arrival times, with covert channels representing a lower ε value. Cabuk, et al. found that ε-Similarity was more robust against noise injection countermeasures compared to the standard deviation measure.

Porta, et al. use the corrected conditional entropy (CCE) to measure regularity in biological signals[21]. This method is adapted by Gianvecchio and Wang in detecting the regularity of covert channels via a set of packet inter-arrival delays[22]. Conditional entropy is used to detect (potentially) multi-bit symbols. The correcting step allows us to determine a minimum value based on an incomplete set of data, as would be the case with any practical sample.

The general assumption made by the above authors is that regularity will increase when a covert channel is present, and while this is true for the example datasets presented, a service which has a naturally high regularity will have its regularity decreased by the addition of a channel. For this reason the fluctuation in regularity should be taken against a baseline analysis of the traffic as a relative indicator where a change in regularity either up or down may indicate a covert channel.

In fact, while Porta, et al. seem to think of entropy purely as a measure of noise, we can also think of entropy as a measure of information[21]. Indeed, Shannon uses entropy to measure the maximum compressibility of data[3]. This means that if we account for information which we are sending via the legitimate channel, plus any provable sources of noise, the remaining noise is the combination of real noise and covert channel information. The entropy of this remaining noise gives us an upper bound of the information which is being transferred by a covert channel.

Another indicator of a covert timing channel is the distribution of the traffic delays. A simple example being outlined by Berk, et al., who note that network traffic between two points will tend towards a relatively normal distribution (with a positive skew) around the average delay for the connection[23]. When looking at a simple timing channel, the distribution has peaks around the delays representing symbols used by the timing channel. Indeed, as Figure 1 shows, a simple implementation of the JitterBug algorithm shows two distinct modes for the encoding of binary data. While Berk, et al. suggest that a covert channel with an even symbol set can be detected by comparing the distance

between the mode and mean of the distribution, it's likely that more robust mechanisms to detect a multi-modal distribution would be more effective, particularly against odd symbol sets.

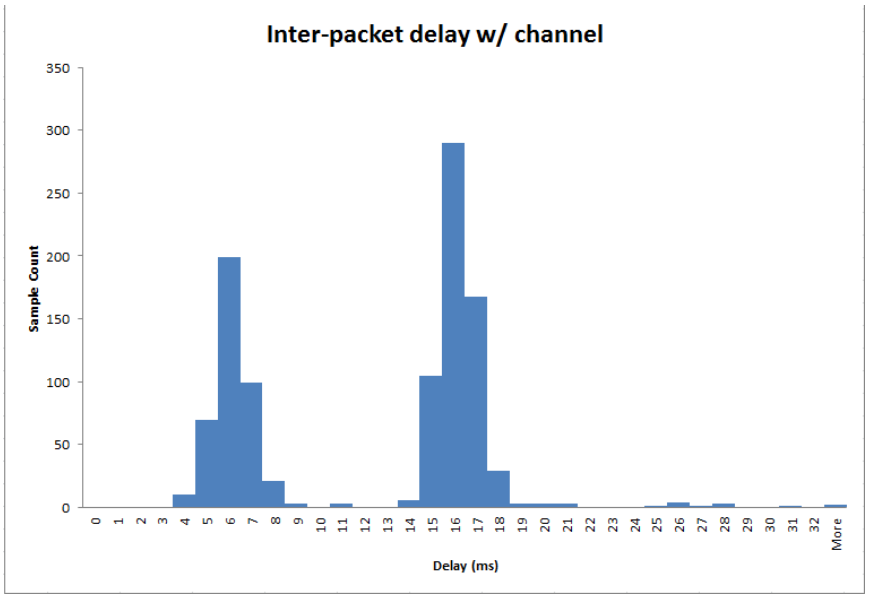

Figure 1 - Simple JitterBug delays over local UDP traffic, as described in Appendix A.

The Kolmogorov-Smirnov test for goodness of fit gives a measure for how close two statistical distributions are to one another[24], and has been used as a detection method for covert timing channels by comparing covert traffic with typical protocol data[15]. However protocol mimicry is a common technique to circumvent this and similar traffic classification approaches[15, 25].

Peng, et al. look at the use of inter-packet delay based watermarking[26]. They find that delays generated by the watermarking scheme tend towards a combination of a normal distribution (for network delay) and a uniform distribution for the artificial delay used to encode the watermark data. They propose the use of the Sequential Probability Ratio Test (SPRT) algorithm, which was able to successfully detect the watermarking scheme by estimating the ratio of packets from each distribution.

5. Mitigations

When considering mitigating a timing channel, we must first consider the capabilities of the attacker. The first case to is consider is one in which the covert receiver can see the content of the legitimate channel, for example when there needs to be some controlled communication between a high and low-side system in a multi-level system (MLS) architecture. The second case is that an encrypted communications channel is sending information over an untrusted channel. Since the former does not allow us to create any false packets we will consider it the worst case, and any mitigations which apply to this

scenario should apply universally. However the latter part of this section will also consider some additional methods for the second case.

5.1 Rate Limiting

Rate limiting is the simplest way to mitigate against a timing channel. Since timing channels are measured by the number of bits which can be encoded per packet of data, reducing the amount of data which can be send over the legitimate channel must also limit the rate at which the timing channel can exfiltrate data. This method is clearly not ideal, however, since any reduction in covert traffic comes at the expense of legitimate traffic due to the fact that there is likely to be no reduction in the proportion of covert traffic versus legitimate traffic. Ideally the legitimate traffic required should be minimised at design time to ensure a lower timing channel capacity, and rate limiting can be used to enforce such restrictions.

The extreme case of rate limiting is to completely eliminate the channel, which would provide perfect security. While most of the time this impractical, alternate approaches and architectures may allow for elimination of communication channels while retaining many of the functional benefits[27].

5.2 Quantization

While we would argue that the limited precision of the system clock is enough to consider any timing channel a discrete source, the actual limit can be difficult to quantify. For this reason some authors suggest that timing channels are naturally a continuous channel, since there is no well-defined quantum of time[18]. The use of a quantizing step can enforce the discrete properties of a channel by allowing messages to be released only at a certain interval. While Kang et, al. argue that quantization is only useful to give us a quantifiable ceiling without necessarily affecting the covert channel[28], quantization can proportionally affect the timing channel more than the legitimate one.

For example we may enforce a 10 ms tick, which means that a packet received at 4 ms must be delayed for 6 ms before it's released. This gives us a guarantee that the maximum rate of the timing channel is 100 bits/second. If we assume a precision of 1 ms for the system clock, an un-quantized channel could encode $\log_2(10)$ bits between packets, or around 332 bits/second given the same average packet rate of 100packets/second.

5.3 Noise Generation / Random Delay

Quantization, while able to guarantee a fixed limit to channel capacity, still reduces the capacity of the legitimate channel significantly. The generation of a random delay would allow us to, at least on a surface level produce the same benefit with an average delay half that of a uniform quantizing delay. However, while the random delay distribution would uniformly cover the defined interval period, a covert channel would have some statistical grouping around its true distribution. This statistical property could allow a covert

receiver to retrieve some information using methods similar to those proposed by Peng, et, al[26]. This means that we have no guarantee of the channel being disrupted completely. Fortunately, since the noise is being generated by a known source, we can use Shannon's capacity calculations to determine an upper limit on the channel. In practice, we would expect better results in the absence of an extremely sophisticated encoding scheme, which in itself would be costly to the covert channel's bandwidth.

5.4 Packet Generation

In the case of connections where the attacker can't distinguish legitimate traffic from generated traffic, for example in protecting an encrypted communications link over an untrusted network, we could also include packet generation as a counter measure. The major advantage to the ability to generate packets is that we can reduce the potential channel with no or minimal delay to legitimate traffic.

The most extreme mitigation is simply to flood the channel so that packets are being sent at the maximum rate in a way that is unaffected by the timing of the input[6]. While this would eliminate any timing channel it would also saturate the shared bandwidth of the network for all devices, with the only practical solution being to limit the rate at which devices can communicate, negating many of the advantages of a packet switched network. As with noise generation through delay (section 5.3), adding random packets can also be used to create noise, disrupting a covert timing channel without significant disruption of legitimate traffic.

6. Directions

With the rise of internet surveillance and counter-surveillance tools, much of the academic conversation around timing channels and associated techniques for detection has moved into this space. Protocol mimicry, for example, is used to disguise Tor traffic using known signatures[25, 29]. Timing channels are also used as a way to watermark anonymous connections as a way to unmask network users and trace back their connections[30]. Overall the discussion seems to be re-framing around traffic flow analysis rather than the building of high assurance and trustworthy systems. Part of this may be because any link across domains will include opportunity for higher bandwidth covert storage channels, which themselves may be difficult to counteract. In the usual case it's considered simpler to either eliminate the channel completely or accept the risk of a timing channel, which has a lower likelihood than other approaches.

7. Acknowledgments

The author would like to thank Samuel Chenoweth for his work on the chaff algorithm.

8. References

1. Lampson, B.W., *A note on the confinement problem.* Communications of the ACM, 1973. **16**(10): p. 613-615.
2. Girling, C.G., *Covert Channels in LAN's.* IEEE Transactions on Software Engineering, 1987. **13**(2): p. 292.
3. Shannon, C.E., *A mathematical theory of communication.* SIGMOBILE Mob. Comput. Commun. Rev., 2001. **5**(1): p. 3-55.
4. Lipner, S.B. *A comment on the confinement problem*. in *ACM SIGOPS Operating Systems Review*. 1975. ACM.
5. Padlipsky, M., D. Snow, and P. Karger, *Limitations of end-to-end encryption in secure computer networks*. 1978, DTIC Document.
6. Moskowitz, I.S. and M.H. Kang. *Covert channels-here to stay?* in *Computer Assurance, 1994. COMPASS'94 Safety, Reliability, Fault Tolerance, Concurrency and Real Time, Security. Proceedings of the Ninth Annual Conference on*. 1994. IEEE.
7. Kemmerer, R.A., *Shared resource matrix methodology: An approach to identifying storage and timing channels.* ACM Transactions on Computer Systems (TOCS), 1983. **1**(3): p. 256-277.
8. Gray III, J.W., *Countermeasures and tradeoffs for a class of covert timing channels.* 1994.
9. Kemmerer, R.A. *A practical approach to identifying storage and timing channels: Twenty years later*. in *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*. 2002. IEEE.
10. Zander, S., G. Armitage, and P. Branch, *A survey of covert channels and countermeasures in computer network protocols.* Communications Surveys & Tutorials, IEEE, 2007. **9**(3): p. 44-57.
11. Cabuk, S., C.E. Brodley, and C. Shields. *IP covert timing channels: design and detection*. in *Proceedings of the 11th ACM conference on Computer and communications security*. 2004. ACM.
12. Bhadra, S., et al., *Communication through jamming over a slotted aloha channel.* Information Theory, IEEE Transactions on, 2008. **54**(11): p. 5257-5262.
13. Shah, G., A. Molina, and M. Blaze. *Keyboards and Covert Channels*. in *Usenix security*. 2006.
14. Sellke, S.H., et al. *TCP/IP timing channels: Theory to implementation*. in *INFOCOM 2009, IEEE*. 2009. IEEE.
15. Gianvecchio, S., et al. *Model-based covert timing channels: Automated modeling and evasion*. in *Recent Advances in Intrusion Detection*. 2008. Springer.
16. Alliance, M., *MIPI Alliance Launches New M-PHY and UniPro Specifications for Mobile Device Applications.* Jun, 2011. **10**: p. 1.
17. Cock, D., et al. *The last mile: an empirical study of timing channels on seL4*. in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. 2014. ACM.
18. Giles, J. and B. Hajek, *An information-theoretic and game-theoretic study of timing channels.* Information Theory, IEEE Transactions on, 2002. **48**(9): p. 2455-2477.
19. Hu, W.-M., *Reducing timing channels with fuzzy time.* Journal of computer security, 1992. **1**(3-4): p. 233-254.

20. Ogurtsov, N., et al. *Experimental results of covert channel limitation in one-way communication systems*. in *Proceedings of the 1997 Symposium on Network and Distributed System Security*. 1997. IEEE Computer Society.
21. Porta, A., et al., *Measuring regularity by means of a corrected conditional entropy in sympathetic outflow.* Biological cybernetics, 1998. **78**(1): p. 71-78.
22. Gianvecchio, S. and H. Wang. *Detecting covert timing channels: an entropy-based approach*. in *Proceedings of the 14th ACM conference on Computer and communications security*. 2007. ACM.
23. Berk, V., et al., *Detection of covert channel encoding in network packet delays.* Rapport technique TR536, de lUniversité de Dartmouth, 2005: p. 19.
24. Massey Jr, F.J., *The Kolmogorov-Smirnov test for goodness of fit.* Journal of the American statistical Association, 1951. **46**(253): p. 68-78.
25. Weinberg, Z., et al. *StegoTorus: a camouflage proxy for the Tor anonymity system*. in *Proceedings of the 2012 ACM conference on Computer and communications security*. 2012. ACM.
26. Peng, P., P. Ning, and D.S. Reeves. *On the secrecy of timing-based active watermarking trace-back techniques*. in *2006 IEEE Symposium on Security and Privacy (S&P'06)*. 2006. IEEE.
27. Beaumont, M., J. McCarthy, and T. Murray, *The cross domain desktop compositor: using hardware-based video compositing for a multi-level secure user interface*, in *Proceedings of the 32nd Annual Conference on Computer Security Applications*. 2016, ACM: Los Angeles, California. p. 533-545.
28. Kang, M.H., I.S. Moskowitz, and S. Chincheck. *The pump: A decade of covert fun*. in *Computer Security Applications Conference, 21st Annual*. 2005. IEEE.
29. Mohajeri Moghaddam, H., et al. *Skypemorph: Protocol obfuscation for tor bridges*. in *Proceedings of the 2012 ACM conference on Computer and communications security*. 2012. ACM.
30. Houmansadr, A., C. Brubaker, and V. Shmatikov. *The parrot is dead: Observing unobservable network communications*. in *Security and Privacy (SP), 2013 IEEE Symposium on*. 2013. IEEE.

This page is intentionally blank.

Appendix A Timing Channel Test Bed

In order to understand the effects of chaff generation on relative timing channels, a simple testbed was developed. The testbed consisted of a pair of Android virtual machines (VMs) running a simple User Datagram Protocol (UDP) based messaging application, which included basic file transfer capabilities. The messaging application sends encrypted UDP packets over an untrusted network to the receiver VM.

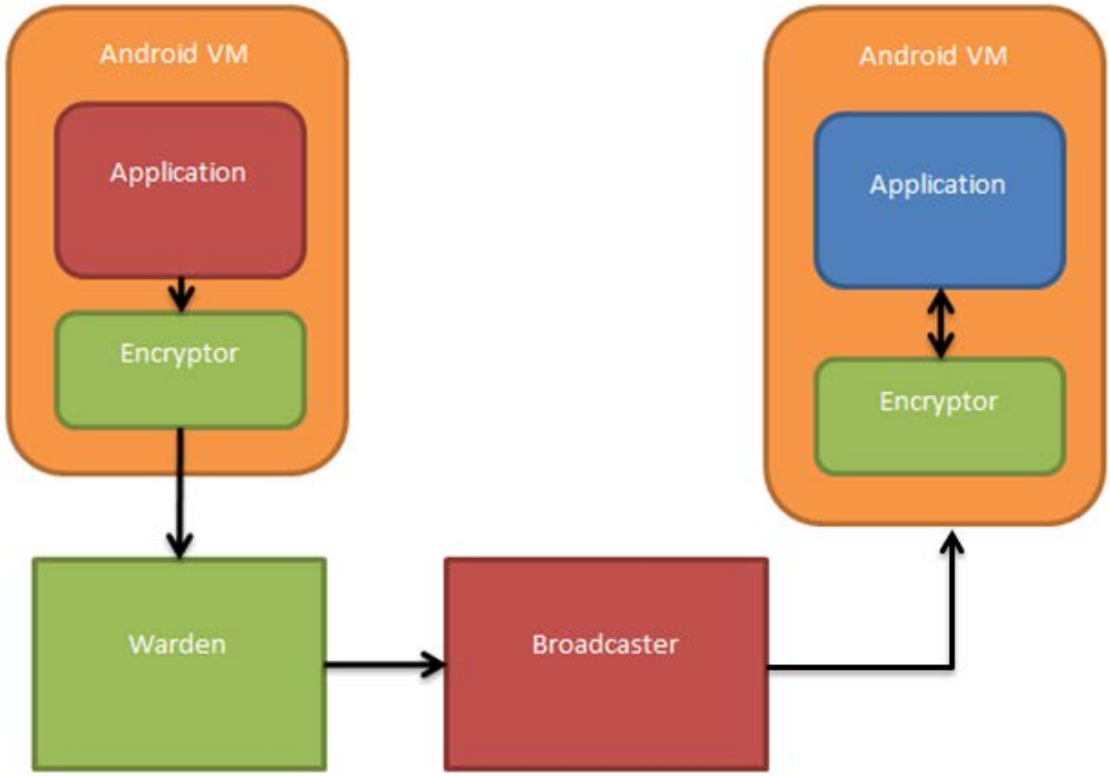

Figure 2- Testbed architecture

Each VM includes a trusted encryptor, ensuring that messages cannot be read by the untrusted network, which in our architecture is represented by the Broadcaster service. A malicious application on the sender VM is able to delay messages so as to implement a covert timing channel between itself and the broadcaster service (both in red). The Warden is placed so that it may examine and modify the traffic from the sender VM.

For the example covert channel we implemented a JitterBug with a 30 ms timing window split into 3 symbols representing 0, 1 and end of message (EOM), allowing us to transmit binary encoded messages over the timing channel. The algorithm was chosen for both its simplicity of implementation and because the timing window is a core mechanism of many relative-delay based algorithms (section 3).

The warden collected a rolling sample of the inter-packet delay for analysis.

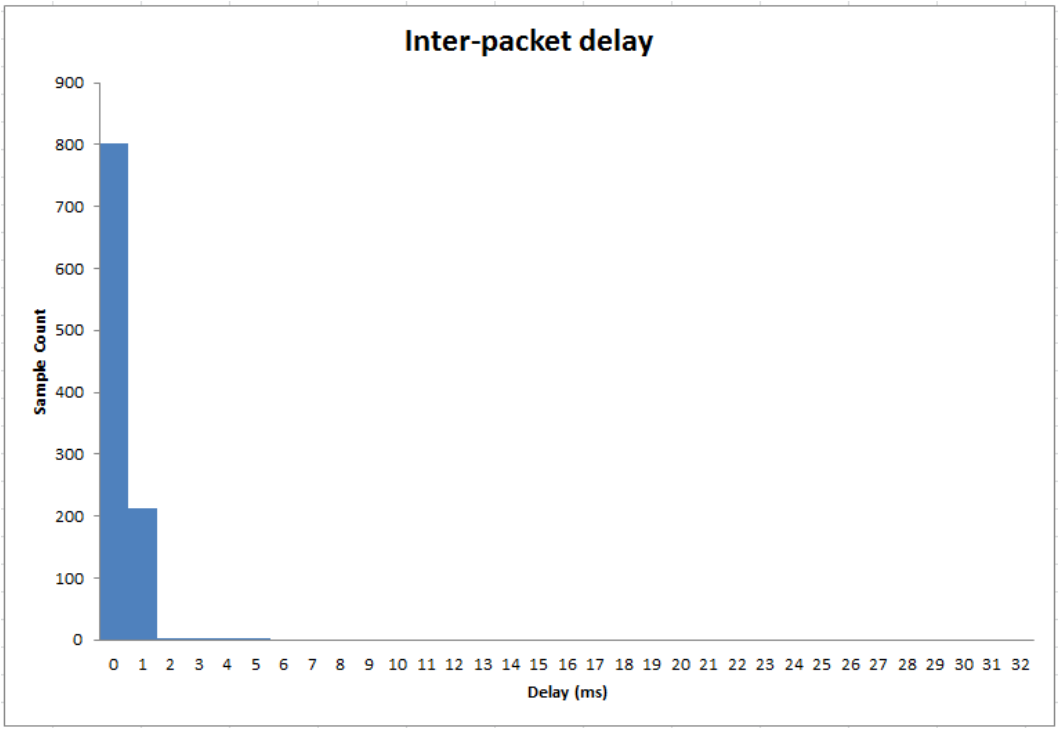

Figure 3- Inter-packet delay for a file transfer with no timing channel

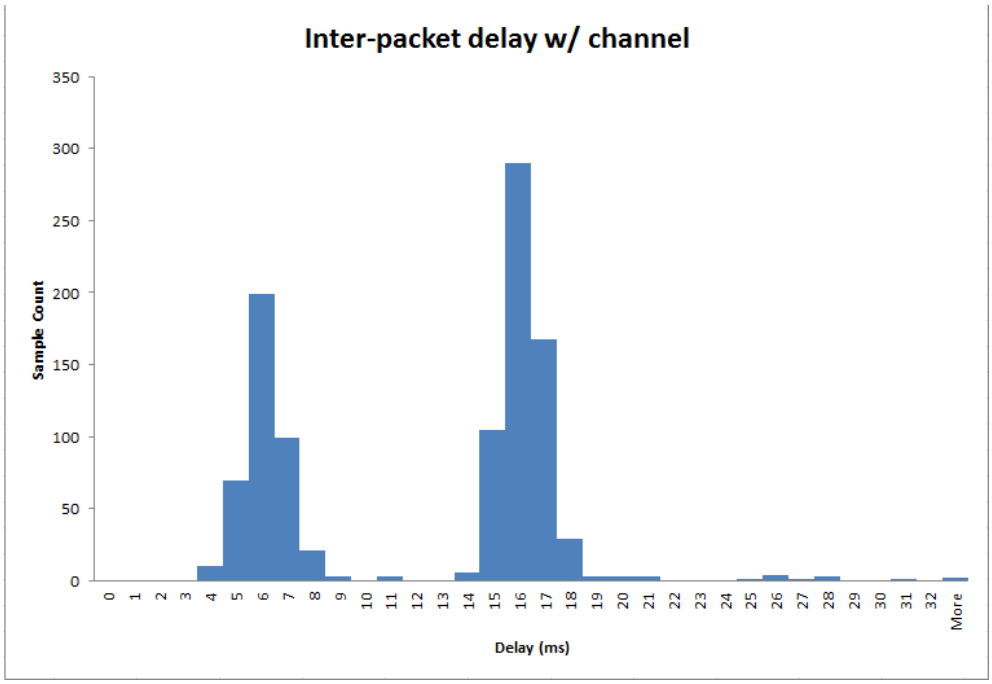

Figure 4- Inter-packet delay for a file transfer with a JitterBug channel

The examples above in Figure 3 and Figure 4 show the inter-packet delays as captured by the warden when transferring a file between the two VMs. As Figure 4 shows, the JitterBug algorithm shows a distinct bimodal distribution (the EOM symbol is used far less often than the 0 and 1 symbols). Since the testbed was implemented on a single machine, the natural delay is extremely low. The covert message data consists of a string encoded latitude/longitude value.

To create a chaff generator a dummy packet was generated by the warden at a time interval calculated using Java's SecureRandom class. The minimum rate is given in milliseconds, and a random value is assigned between zero and the minimum rate, making the effective rate an average of half the minimum rate. The received message is tested against the correct message to determine the error rate, which is measured as the probability that a bit will be flipped.

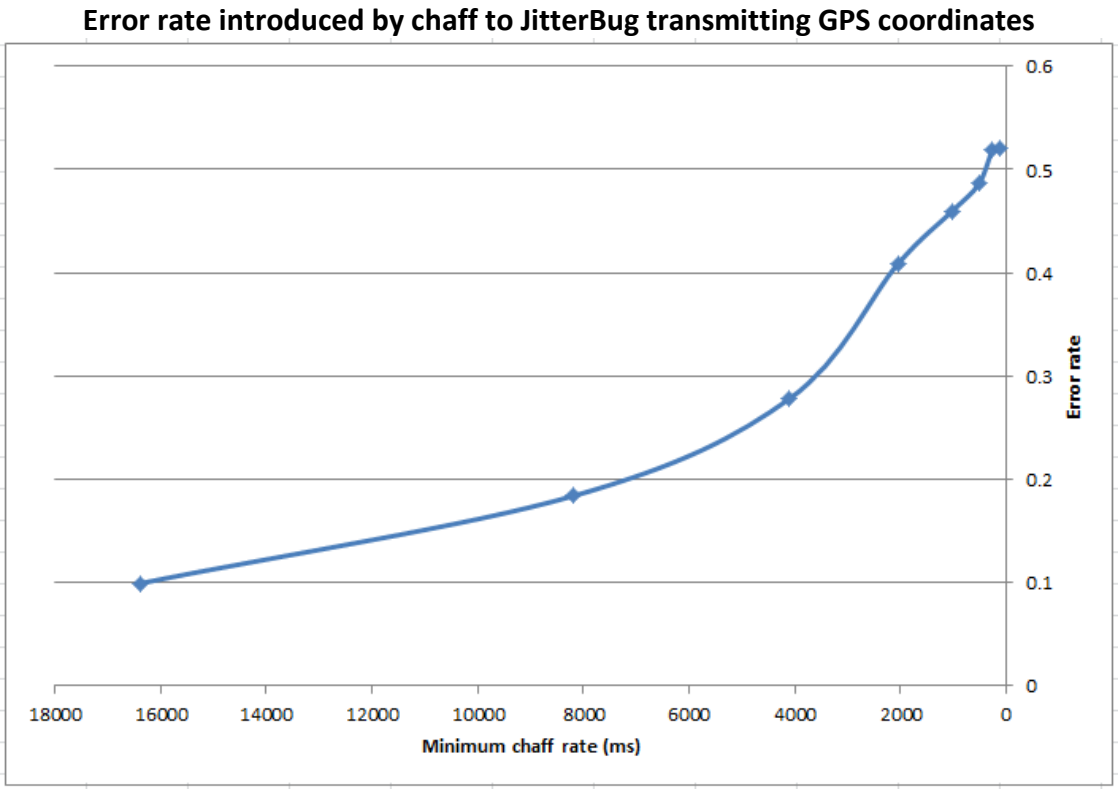

Figure 5 - Error rate introduced by chaff algorithm

As Figure 5 shows, the error rate for JitterBug increases significantly, even when injecting packets at a fairly low frequency. This is due to the fact that the injected packet adds a symbol to the received message, offsetting the remaining bits by one. Clearly this is highly effective against a simple channel, however shortening the message, or correcting for errors will reduce or potentially eliminate this effect.

As a test, we reduced the message length to 8 bits (Figure 6).

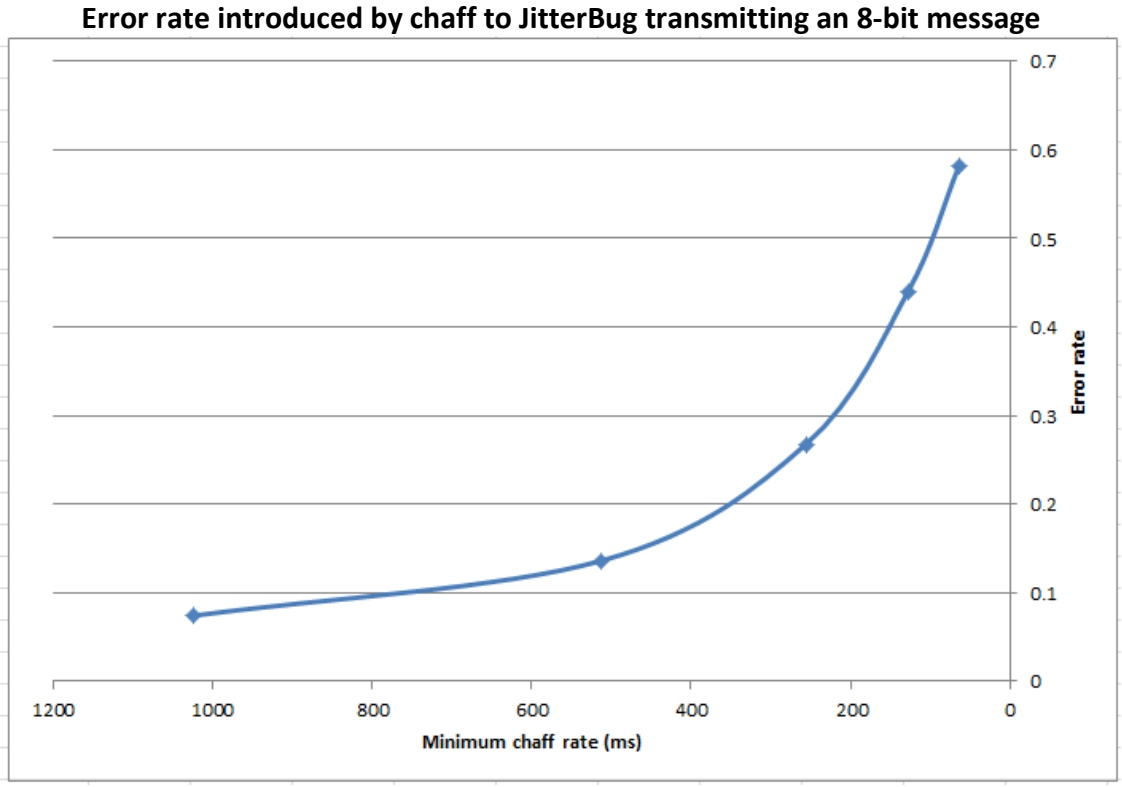

Figure 6 - Error rate with chaff for 8-bit message

As Figure 6 shows, the chaff needs to be sent at a rate high enough to fall within the total message transmission time (which in this case is around 12 ms) to ensure that the message is successfully disrupted in every case.

However, we must also assume that the receiver can re-transmit the message or use another error correction algorithm to reduce the effectiveness of noise. In this case we must interrupt or modify each inter-packet delay through a number of chaff packets relative to the number of legitimate packets sent, or to the symbol window length (in our example case 3 ms).

While our examples use a file transfer to maximise the number of packets transferred, a sparser traffic source would require fewer chaff packets. Additionally, the parameters of the noise could potentially give us a way to calculate the minimum practical symbol set, and thus the reduction in capacity of the covert channel. Even without a formal guarantee, the use of chaff can significantly reduce the reliability of the timing channel with fairly low cost to the legitimate bandwidth.

Unfortunately, time restraints meant we were not able to further examine the effectiveness of this countermeasure against additional, more robust channels; however as a low-cost mitigation the chaff algorithm shows significant promise at disrupting relative timing channels if tuned correctly.

DEFENCE SCIENCE AND TECHNOLOGY GROUP DOCUMENT CONTROL DATA			1. DLM/CAVEAT (OF DOCUMENT)
2. TITLE Covert Channels Over Network Traffic: Methods, Metrics, and Mitigations		3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED LIMITED RELEASE USE (U/L) NEXT TO DOCUMENT CLASSIFICATION) Document (U) Title (U) Abstract (U)	
4. AUTHOR(S) Patrick Prendergast		5. CORPORATE AUTHOR Defence Science and Technology Group PO Box 1500 Edinburgh SA 5111	
6a. DST GROUP NUMBER DST-Group-TN-1695	6b. AR NUMBER AR-017-008	6c. TYPE OF REPORT Technical Note	7. DOCUMENT DATE October 2017
8. OBJECTIVE ID	9.TASK NUMBER N/A	10.TASK SPONSOR Paul Buckland	
11. MSTC Cyber Assurance and Operations		12. STC Active Security Technologies	
13. DOWNGRADING/DELIMITING INSTRUCTIONS		14. RELEASE AUTHORITY Chief, Cyber and Electronic Warfare Division	
15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT *Approved for public release* OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111			
16. DELIBERATE ANNOUNCEMENT No limitations			
17. CITATION IN OTHER DOCUMENTS Yes			
18. RESEARCH LIBRARY THESAURUS Computer Security, Information security, Data protection, Information theory			
19. ABSTRACT This technical note examines the current state of open literature regarding covert timing channels over network traffic. The paper consists of a brief background of timing channels, some examples of covert timing channels over network traffic, metrics to define the capacity and probability of a channel, and finally methods to mitigate timing channels.			