**Anomalous Payload Based Worm and Attack Detection**

Michael J. Kopps

The University of Colorado at Colorado Springs

Colorado Springs, Colorado U.S.A.

mkopps@uccs.edu

*Abstract* — Zero day worms are impossible to detect utilizing the standard signature based Intrusion Detection Systems and virus scanning programs. Any organization or network on the open internet that use a signature based means of detecting malicious software or traffic are therefore vulnerable to any new worm until it has been detected and signatures released, which may be too late to prevent large scale destruction. Worms by their very nature are self propagating, and it is possible to exploit this signature to discover and attempt to prevent the spread of these worms. A sensor that is able to automatically identify malicious traffic is capable of sharing this information with other network hosts and can be quite effective at slowing the spread of the worm before substantial damage has been wrought.

# Introduction

S

ignature based worm detection methods are the most common form of worm detection in use today. Signature based sensors are cheap, efficient, and simple to install and maintain. However, signature based sensors are blind to a type of worm called a zero-day worm. This category of worm encompasses worms that have never been seen before and therefore no signatures have been released to detect their presence on a network. Similarly, they often exploit software vulnerabilities that have just been identified and a security patch has not been created. Zero day worms and attacks are only a serious threat during what is known as the vulnerability window [6]. This window encompasses the time period including the time for the vulnerability to become known, analysis and development of a patch for the problem, issuing the patch to all affected customers, and finally releasing the signature to anti-malware filters. According to Symantec Corp in late 2006, this time period was about 28 days [7]. Worms are the second most dangerous type of malicious code threat on the internet [8]. A thriving underground marketplace exists in the trading and creating of new malicious code that is able to steal private information. Many of the new threats developed are merely variants of existing exploits, but they may not be detectable by the signatures of those existing threats. It is therefore important to take a proactive approach to reducing the threat new worms pose to the security, stability, integrity, privacy, and economy of the world’s organizations and people.

Zero day worms and all worms in general are created with several possible purposes in mind. They could be created to compromise the security on a system to give a black-hat hacker access to valuable data on the system. They are used to create massive bot networks that can be used to generate an attack on a third party for any number of nefarious reasons. The damage resulting from a wide spread worm can reach into the millions if not billions of dollars from lost productivity, wasted computation power, data loss, and theft.

There are several simplistic methods available to guard against zero day worms. However, many of these merely provide a first layer of protection against these worms and do not provide the sophistication necessary to stop the ever increasingly sophisticated worms that may be able to get around these methods. Also, these methods may provide too much of a burden on legitimate uses to be feasible in a high usage environment. Port knocking is a means of opening a closed port on a firewall by creating connection attempts to a specific sequence of port numbers. This acts much like a password to gain access through the firewall. Port knocking does provide a good level of security, but it is overly prohibitive for open access systems such as public web servers, and is therefore unfeasible in many systems. White listing is a very common form of security where only a pre defined set of systems are capable of communicating with a particular computer or service. Once again, however, this is unfeasible in the context of a public facing website or a Virtual Private Network head end box that must be able to be contacted by any IP address on the internet.

Clearly improvements are needed to continue to prevent attacks on the availability and integrity of any internet connected system. These methods must be easy to implement and maintain, and should provide good security with minimal human interaction while in operation. This security should be able to handle changing operating environments while still providing adequate protection.

*Organization:* The rest of the paper is organized as follows: In the next section, we provide an overview some of the work done in the area of payload based worm detection sensors. In Section III, we provide a detailed view of the implementations of how each of these sensors work to provide a broad view of the different methods that can be employed. In Section IV we look at a specific anomalous detection sensor that is a high rate of detection and a relatively low rate of false positives. This system is particularly robust and s capable of responding quickly to new and different threats. Section V provides an overview of the deficiencies of this same system. Finally, we present our conclusions and what work still remains in the ongoing efforts to reduce the threats posed by zero day attacks and worms.

# Overview of Designs

Signature based software is intrinsically blind to zero day attacks. Due to the pervasiveness of this type of solution, there are many networks and organizations that are vulnerable to the ever increasing number of malicious software attacks that are created every day. This paradigm favors those who wish to do harm rather than those who wish to protect and provide a service to customers.

Even if a signature has been created for a particular worm, that worm may be able to modify itself as it propagates, and therefore can change itself enough such that the provided signature does not match new, modified worm code. Also problematic is the ability for worms to fragment the packets used to propagate itself, which can also mask the true nature of the payload.

Some research has been done in the field of software that is able to automatically generate new signatures from new threats. Some of those systems are explained in the next subsections

## Honeycomb

Honeycomb is an extension of the open-source project honeypot honeyd. It inspects the protocol headers of the incoming traffic as well as the payload information to generate signatures for network intrusion detection systems such as the commonly implemented Snort. Since Honeycomb is based on a honeypot design, it is unable to prevent attacks directed at specific hosts, rather it requires the worm to target it before it is able to capture traffic and generate the signature.

## Autograph

Autograph is a system that monitors suspicious flows of traffic and automatically generates signatures based on IP protocol, destination port, and a byte sequence inside of the payload. The analysis occurs on saved packets, and it uses the self-propagation property of worms to identify a unique payload block that grows in prevalence over time. The blocks that are compared are not fixed length blocks, rather they are adjusted using Content-based Payload Partitioning. Autograph is capable of sharing these signatures with other sensors to decrease the time necessary to stop the spread of a new worm. It also utilizes black lists of common patterns to reduce the number of false positives.

## Earlybird

Earlybird is a system very similar to Autograph, however it operates as a centrally managed network, instead of allowing the sensors to communicate with each other. It also eschews the overhead of reassembling the TCP packet flows if they are fragmented at the network layer. As a result, it is blinded by worms that disguise themselves by fragmenting themselves while they self-propagate.

All of these sensors are very capable of detecting new worms; however they are not able to stop the spread of a worm immediately. Rather, they detect the worm after the propagation has occurred, which works in the favor of black hat hackers since worms have an opportunity to spread in the wild. It is important for security systems to be able to stop the worm from spreading immediately after being seen in the network.

## PAYL

The PAYL sensor system was developed by a team at the Intrusion Detection System Lab at Columbia University. The goals of this research is to provide a zero day worm sensor that is capable of detecting worms as soon as they appear to the sensor, or at least at the worms first propagation attempt. It can then notify other sensors of the possibility of an emerging threat and can collaborate to verify if a sensor reading is a true positive or a false positive. The PAYL sensor is unique in its ability to immediately detect emerging threats without the need to wait for the worm to propagate and compromise many potentially high importance systems.

# Anomalous Payload Detection

Anomalous payload detection sensors are capable of working based on the theory that any particular network or server will receive traffic that matches a certain pattern of data. Some servers will expect to see more of a particular set of data than others, such as the differences between a web server and a streaming media server. Web servers send a large amount of textual data, while media servers send much more distributed data within the packet payloads. Furthermore, servers also tend to send and receive data only on certain ports numbers. For example, a web server is expected to have much traffic directed at TCP port 80, but traffic flowing out of this same server will not be destined to port 80.

## Honeycomb Implementation

Honeycomb is a honey pot means of monitoring attack traffic on a network. Honey pots are ideal sensors as any packets destined to them are intrinsically suspicious because they host no useful service to any outsider.



Figure : Horizontal pattern detection as used by Honeycomb. From [2].

Honeycomb keeps state information for all incoming and outgoing traffic flows. The states are kept beyond the end of the flow to aid in creating signatures for previously seen traffic. For each flow, Honeycomb creates a new record containing signature information pulled from the reassembled traffic without regard to previously reported packets. It also tracks anomalies in the packet, such as abnormal TCP flag combinations, invalid IP fragmentation offsets and the like. Once the signatures have been created from the protocol headers, Honeycomb uses a Lowest Common Sequence algorithm to extract signatures from the payloads. If no interesting signature was detecting, processing on this flow stops. Otherwise, the signature is added to the signature pool, which is then periodically logged and new signatures are sent to the signature based IDS system.

Traffic flows can be followed in two directions, in a horizontal direction or in a vertical pattern direction. Certain malicious code will expect to send the same data to a vulnerable system but each vulnerable system could send back may reply with very different traffic. Vertical pattern detection is useful in identifying this threat. Other malicious code may send the same traffic at a certain point in time, and a horizontal detection method could be used in this case.

## Autograph Implementation

As previously described, Autograph segregates traffic into two pools, one pool for non-suspicious traffic and the other for traffic identified as being suspicious. The suspicious traffic flows are saved and reassembled from various fragments for further processing and signature generation by the Autograph algorithm. Performing this initial segregation reduces the incidence of false positives being identified by the system, which increases its usefulness. This suspicious pool is divided further prior to signature generation using the destination ports of the packets. Periodically, the system will initiate a signature generation sequence on the saved flows for a particular port in the suspicious traffic pool.

Figure : Vertical pattern detection as used by Honeycomb. From [2].

The signature generation process first splits the flows into smaller blocks. These blocks are created using the Content-based Payload Partitioning (COPP), which splits the flows based on the content of the payload. By utilizing this method, the worm can add or remove bytes, but the block will remain largely unchanged. The probabilistic nature of COPP means the block sizes can possibly be small blocks which are not specific enough and will generate many false positives. It can also create blocks that are too large, which are too specific and will create many false negatives. To prevent these unacceptable conditions, Autograph imposes limits on the maximum and minimum sizes of the blocks COPP uses.

Once these blocks have been defined, the system will discard any unique blocks that originate from a single source IP address. The self-propagating nature of worms means worm traffic should be present in the suspicious pool of traffic at the network router, so in the lack of such duplicated data, a worm is not likely. In fact, it is common for badly behaving sources to send innocuous, yet suspicious traffic. Such sources are unfortunately common, but this data is sourced from a single address, and can therefore be safely ignored. With the singularities removed from consideration, Autograph then searches the remaining blocks for those that reach a certain threshold of prevalence within the pool. Those blocks that are now selected are sorted based on their prevalence and those with the highest rating are selected as new signatures.

To minimize false positives, an administrator can enter strings that are prohibited from being signatures. It is expected this will be done during a training period, and the administrator will manually monitor the generated signatures for those that are normal traffic, such as normal parts of web requests and web pages. Although traditional blacklists often grow to unmanageable proportions in traditional systems, the researchers have found that modest blacklists composed of less than10 signatures dramatically improves the false positive rate. The signatures that are not filtered by the blacklist are then forwarded on to a traditional signature based IDS.

Autograph also incorporates the concept of cooperation between different sensor installations. Traditionally, security software does not collaborate or cooperate, instead relying upon a centralized top down distribution from a trusted source. Autograph allows a single sensor to distribute the signature from a suspected worm as soon as it identifies such traffic. As a result, dispersed systems can become inoculated before their instance of Autograph has successfully flagged the suspicious traffic as a signature. It is important to note this does not inoculate a system before the worm has a chance to spread; rather the system is inoculated before that specific sensor could provide the inoculation by itself.

Another feature of Autograph is the ability to share suspicious source IP addresses between installations. This aids in generating more sensors monitoring those sources that have been flagged as suspicious. If a source is malicious, this will place its packets into the suspicious flow pool, encouraging all cooperating Autograph installations to perform signature generation on that traffic, increasing the chances of one generating a worm signature for that source, speeding up inoculation time. One obvious limitation of this approach is the existence of source spoofed packets. Since Autograph categorizes packets based on source IP address, this can cause much thrashing in the algorithm. A possible solution is to send the source a SYN/ACK, and if the source replies with the appropriate sequence number, it is not spoofed.

Figure : Sample distribution of 1-grams from three sample servers on port 80, on packets of length 1380. From [1].

Figure : Sample distribution of 1-grams from three smaple servers on port 80, on packets of length 249 bytes. From [1].

## PAYL Implementation

Content sent through the network to a particular server will generally have a similar distribution of data grams in all of its traffic. PAYL operates on the supposition that a worm will have a different distribution of data in its network traffic, and using this difference, novel worms can be detected. PAYL first requires a training period that will generate a nominal traffic profile that will be used by the sensor to compare incoming traffic with.

The processing is comprised of scanning the incoming data with a particular port number and packet length with a sliding window of size n. The frequency of the data contained in each n sized window, called an n-gram, is then computed. Figure 4 shows a sample distribution of 1-grams for three different servers on port 80 and a packet length of 249. Figure 3 shows similar data using packet lengths of 1380 bytes. For the training period, this frequency provides the normal traffic pattern to be compared with the live network traffic during normal operation. During normal operation, each incoming packet has the same analysis performed on it, and is compared with the normal using Mahalanobis distance for its port number and packet length. If the distance resulting from this comparison is larger than a threshold, PAYL will issue an alert and potentially generate a signature for use by a signature based IDS such as Snort. Figure 5 shows the analysis of the payload information of the CodeRed II worm. In all three servers’ nominal payload information, this worm will show an anomalous payload frequency spike.

# PAYL Advantages

According to its researchers, PAYL is completely autonomous. It performs self-calibration, trains the models it uses automatically, and updates itself with new models as required. This lack of human interaction increases speed of developing new signatures and shrinks the vulnerable time frame even further.

## Data Diversity

As previously discussed, many servers will send a specific set of nominal traffic data. Web servers will have a different nominal profile than an E-Mail server and a media server. An obvious shortcoming in scanning network traffic for anomalous payload data is the situation where the malicious data indeed carries the same distribution pattern as the nominal traffic for that server.

In order to solve this problem, the researchers who developed the PAYL system devised a means of collaboration between PAYL sensors. Each sensor is each protecting a system which all have disparate nominal traffic patterns. Figure 2 clearly shows the difference between normal traffic flows across different servers. If one sensor detects a potential new attack, it can share its signature with the other collaborating sensors. In this way, the sensors are inoculating all others against new threats, even if those threats contain the same data distribution as the nominal traffic patterns for that system. A worm or attack that wishes to avoid detection will need to conform itself to the traffic patterns of all collaborating systems. However, even knowledge of what systems are collaborating is difficult to obtain, much less the nominal traffic patterns at each.

## Ingress/Egress Correlation

PAYL is also capable of monitoring outgoing data for similar signatures of anomalous data that was flagged upon entry past the sensor. Worms wish to propagate themselves, and to do so, they must send copies of themselves out through the network to other hosts. The outgoing traffic at a minimum must contain the worm executable code, which must be substantially similar to the incoming data. Unlike the other methods surveyed in this paper, PAYL can identify a worm based on the first attempt at propagation rather than waiting for multiple propagation attempts before flagging and generating a signature.

The authors evaluated three different comparison algorithm techniques. String equality compares the ingress and egress packets directly for equality. This technique is fast to execute and will produce a very limited set of false positives, but it will be completely ineffective against worms that modify themselves even in the smallest manner. Longest common substring, like that used in Honeycomb, searches for similarities in the payload, and identifies the longest string within the payloads that are equivalent. Longest common subsequence searches for the longest sequence of n-grams that may or may not be contiguous in the payload. This helps solve the shortcoming of worms inserting bytes into their machine code. This method requires the highest processing overhead to perform, and it can produce many more false positives than the other two methods.

Figure : Payload distribution of 1-grams in the CodeRed II worm. From [1].

## Collaboration

Most current malware scanners and anti-virus scanners that can generate their own signatures do not collaborate with other similar sensors to inoculate each other or to corroborate results. Similar to the way PAYL can leverage multiple installations to provide data diversity to locate and inoculate against new worms, it can also share information about payloads to confirm the presence of potentially dangerous payloads. If two different sites independently identify a new threat, then the likelihood that this payload is a credible threat is greatly increased. This corroboration of potential threats amounts to voting on what should be considered malicious, and as a result, it also reduces the false positive rate.

An important factor to consider when sharing data between sites is privacy of the data. The two sites may be two different companies who wish to keep their data proprietary, but wish to efficiently corroborate their anomalous payloads. To solve this, the authors propose using the Manhattan distance or a lowest common substring of the “Z-string.” The Z-string is “a string of bytes whose frequency in the data is ordered from the most frequent to the least, serving as representative of the entire distribution” [1]. It is impossible to determine the original content of the packet using the Z-string, but performing the lowest common subsequence will provide enough information to corroborate the two alerts, even if the payloads seen by each sensor is different due to polymorphism by the worm.

# PAYL Issues

As with any new technology, there are problems that need to be overcome before the system can be put into place.

## Training Data

The PAYL system relies upon the normal traffic flow information to compare incoming data against. Obviously, this normal traffic information must represent the true information about the traffic, and any spurious or anomalous data in this training dataset will introduce erroneous data and possibly cause false negatives. The sheer size of the training data makes manual identification and removal of anomalous and attack packets tedious and difficult. Doing so can leave the training data with an un-specific set or with a set that is too specific. These situations will reveal either holes worms can exploit or an unacceptable amount of false positives.

Again, the authors [4] propose using the natural behaviors of both normal traffic flows and of worms and anomalies to cleanse the training data set. In a dataset that spans a large period of time, abnormalities will appear to be confined in small time periods, and the normal traffic will be seen across all arbitrary length time periods. However, not all abnormal traffic is truly abnormal. HTTP cookies can contain random data, which are irregular and therefore abnormal; however they are an expected part of the normal data flow. The first step in sanitizing the data is to split the single large training dataset into smaller ‘micro-models’ of the traffic flow. The reason for doing so is the assumption that normal traffic will be normal across all of these micro-models, and abnormal or attack traffic will be present in a sparse number of these models. The software will then scan the packets in each micro-model and mark each packet as normal or abnormal. The packets are then separated into two sets, one set containing packets that were voted by the majority of models to be normal, and the rest of the packets, used to generate a model of abnormal packets. The set containing normal packets can be used by anomalous payload detection sensors in a live environment.



Table : Sanitation impact on PAYL sensor performance. From .

Cleaning the training data of anomalous and attack data has been shown to increase the true positive rate and decrease the false positive rate in the PAYL sensor [4]. Table 1 shows three servers and the false positive and true positive rate for each when used with both un sanitized data and sanitized data. The list server rate of true positives goes from 64.19% to 86.54% while at the same time the false positive rate went from 64.14% to 2.40%.

It is still possible to incorporate attack data into the normal model if an attacker sustains an attack during the training period. The authors propose a collaborative effort in sanitizing the data will help reduce this problem. If one sites anomalous payload set that resulted from the sanitization process contains matches with another sites data, the malicious data can be removed. A consequence of this process is an increase in the false positive rate.

Another growing concern for sanitizing data is polymorphic worms infecting the training data. The authors performed an experiment where they injected samples of a polymorphic shell code into training data. Specifically, they placed 20 samples within each micro-model. When they ran this modified set of data through the sanitation process, they observed the resultant clean dataset was identical with the training dataset that did not contain the polymorphic shell code. This test shows the data sanitization architecture used by the researchers is capable of identifying and removing polymorphic attacks from training data.

# Conclusion

Computer worms are becoming bigger and more significant threats every year. New threats are being commissioned and created every day to attack the internet as whole or even specific targets. The existing scheme of signature based detection of security threats is unable to react fast enough to counteract all of the new threats that are constantly emerging. Fortunately, a new class of malicious traffic sensors is becoming available for use. These new sensors analyze the payload data contained in the traffic and search for unusual patterns that stray from the nominal traffic usually seen by the sensor. Although these sensors are not perfect, they are a great leap forward in the effort to neutralize the threat of new attack vectors.

Anomalous payload sensors are nimble enough to identify an unseen threat and take immediate action, without the need for human involvement. This can significantly reduce the damage inflicted by an attack, especially if it is a self-propagating worm. By taking a cue from the attackers playbook, security administrators can set up collaborations between sensor sites which will notify each other in real time of emerging threats. The privacy of sharing this data is preserved by using a format that cannot be used to make any guess as to the original content of the suspicious message.

There are also other options for identifying new threats. Honeycomb is a novel system that uses a honey pot to attract unusual traffic and generate signatures to provide a layer of protection against emerging threats. This has an advantage of a simple network traffic analyzer in that the honey pot should not have any traffic directed towards it under normal conditions, and therefore any such traffic is suspect.

In all, anomalous payload sensors add a powerful new tool to the security professional’s toolbox, and can be implemented to provide a significant addition to the existing security of any organization. It requires minimal administration once configured, so even administrators that do not have an extensive background in network security will be able to deploy and reap the benefits of the additional security.

# Future Work

The PAYL system matches incoming anomalous traffic with outgoing traffic, and any correlation seen is automatically suspected to be a self-propagating worm. Although in most cases this is an accurate assumption, some scenarios this could be expected. With peer to peer file sharing, it is normal to see a particular payload incoming to the network, and then exiting the network repeatedly a short time later. Work is required for handling this situation and others like it.

Another area of further research is in the area of combining the many security paradigms and allowing them to share resources. It is not productive to keep security information proprietary, rather the entire internet community will benefit with the sharing of real time information about emerging threats. By quickly sharing suspicious traffic in real time, it could be possible to relegate worms to the same fate as the smallpox virus. Antivirus software could communicate with intrusion detection systems and vice versa to provide a network wide sensor against threats.

As always, a goal should be to reduce the number of false positives and false negatives and increase the number of true positives. It is unlikely for all attacks to be identified and secured before damage is wrought, but more research is always useful.

##### References

1. K. Wang, G. Cretu, and S.J. Stolfo. Anomalous Payload-based Worm Detection and Signature Generation. In *Proceedings of the Symposium on Recent Advances in Intrusion Detection (RAID)*, September 2005.
2. C. Kreibich and J. Crowcroft. Honeycomb-Creating Intrusion Detection Signature Using Honeypots, *In Proceedings of the 2nd Wrokshop on Hot Topics in Networks (HotNets-II)*, November 2003
3. K.-A Kim and B. Karp. Autograph: Toward Automated Distributed Worm Distribution, *In Proceedings of the USENIX Security Syposium*, August 2004.
4. Gabriela F. Cretu, Angelos Stavrou, Michael E. Locasto, Salvatore J. Stolfo, Angelos D. Keromytis "Casting out Demons: Sanitizing Training Data for Anomaly Sensors"*In the Proceedings of the IEEE Symposium on Security & Privacy. May 2008, Oakland, CA.*
5. Wikipedia contributors, "Port knocking," *Wikipedia, The Free Encyclopedia,* <http://en.wikipedia.org/w/index.php?title=Port_knocking&oldid=327540500> (accessed December 13, 2009).
6. Wikipedia contributors, "Zero day attack," *Wikipedia, The Free Encyclopedia,* <http://en.wikipedia.org/w/index.php?title=Zero_day_attack&oldid=326940743> (accessed December 13, 2009).
7. "Internet Security Threat Report" Symantec Corp, Vol. X, Sept. 2006
8. "Internet Security Threat Report" Symantec Corp, Vol. XIV, April. 2009
9. *UCCS CS591 (2009) Retrieved Dec 10, 2009 from* <http://cs.uccs.edu/~cs591/studentproj/projF2009/sjahnke/doc/>

[SELinuxSLIDEsjahnke.pptx](http://cs.uccs.edu/~cs591/studentproj/projF2009/sjahnke/doc/SELinuxSLIDEsjahnke.pptx)