

The Science of Intrusion Detection System Attack Identification

Introduction

A. Among the many vendors of intrusion detection systems (IDS), there is marked variation on what constitutes a network intrusion. This has led to many confusing claims by vendors in the IDS market about the best methodologies and solutions. This paper discusses the pros and cons of the various intrusion detection methodologies and explains the Cisco approach for IDS products. The detection methodologies discussed in this paper include simple pattern matching, stateful pattern matching, protocol decode-based signatures, heuristic-based signatures, and anomaly detection. Although addressing each of these analysis methodologies in detail is beyond the scope of this paper, it covers the basic concepts and differences between the approaches.

The term “signature” in this document refers to a set of conditions that, when met, indicate some type of intrusion event. The algorithm used by the signature could be based on any of the five methodologies covered in this paper (for example, an “anomaly detection signature”). It is important to make this distinction because the term signature is generally more closely associated with the pattern match rather than the other methodologies. In fact, this often leads to misconceptions that a signature-based IDS is limited to only pattern matching, so this definition precludes such misconceptions.

Pattern Matching

Pattern matching is based on looking for a fixed sequence of bytes in a single packet. As its name suggests, it is an approach that is fairly rigid but simple to employ. In most cases the pattern is matched against only if the suspect packet is associated with a particular service or, more precisely, destined to/from a particular port. This helps to lessen the amount of inspection done on every packet. However, it tends to make it more difficult for systems to deal with protocols that do not live on well defined ports and, in particular, Trojans, and their associated traffic, which can usually be moved at will.

The structure of a signature based on the simple pattern-matching approach might be as follows:

If the packet is IPv4 and TCP and the destination port is 2222 and the payload contains the string “foo,” fire an alarm.

This example of a pattern match, of course, is a very simple one, but the variations from this point are also simplistic. You could include a specific starting point and endpoint for inspection within the packet, for instance, or you could specify the TCP flags for packets to be considered. In the end, though, this technique remains the simplest and most primitive building block for intrusion detection.

**Pros:**

- This is the simplest method to detect intrusions.
- This method allows for direct correlation of an exploit with the pattern; it is highly specific.
- This method reliably alerts on the pattern specified.
- This method is applicable across all protocols.

Cons:

- This method can lead to high false positive rates if the pattern is not as unique as the signature writer assumed.
- Any modification to the attack can lead to missed events (false negatives).
- This method may require multiple signatures to deal with a single vulnerability to be exploited. Multiple tools lead to multiple signatures.
- This method is usually limited to inspection of a single packet and, therefore, does not apply well to the stream-based nature of network traffic such as HTTP traffic. This scenario leads to easily implemented evasion techniques.

Stateful Pattern Matching

A more sophisticated method is stateful pattern matching-based analysis. This method of signature development adds to the pattern match the concept that because a network stream comprises more than single atomic packets, matches should be made in context within the state of the stream. This means that systems that perform this type of signature analysis must consider arrival order of packets in a TCP stream and should handle matching patterns across packet boundaries.

How does this scenario affect the example that was introduced in the simple pattern-matching discussion? Now instead of looking for the pattern in every packet, the system has to begin to maintain state information on the TCP stream being monitored. To understand the difference, consider the following scenario. Suppose that the attack you are looking for is launched from a client connecting to a server and you have the pattern-match method deployed on the IDS. If the attack is launched so that in any given single TCP packet bound for the target on port 2222 the string "foo" is present, the alarm fires. If, however, the attacker causes the offending string to be sent such that "fo" is in the first packet sent to the server and "o" is in the second, the alarm does not fire. If the stateful pattern-matching algorithm is deployed instead, the sensor has stored the "fo" portion of the string and is able to complete the match when the client forwards the ".o"

Pros:

- This method requires only slightly more effort to employ than simple pattern matching.
- This method allows for direct correlation of an exploit with the pattern; it is highly specific.
- This method reliably alerts on the pattern specified.
- This method is applicable across all protocols.
- This method makes evasion slightly more difficult.

**Cons:**

- This method can lead to high false positive rates if the pattern is not as unique as the signature writer assumed.
- Any modification to the attack can lead to missed events (false negatives).
- This method may require multiple signatures to deal with a single vulnerability to be exploited. Multiple tools lead to multiple signatures.

Protocol Decode-Based Analysis

Protocol decode-based signatures are in many ways intelligent extensions to stateful pattern matches. This class of signature is implemented by decoding the various elements in the same manner as the client or server in the conversation would. When the elements of the protocol are identified, the IDS applies rules defined by the RFCs to look for violations. In some instances, these violations are found with pattern matches within a specific protocol field, and some require more advanced techniques that account for such variables as the length of a field or the number of arguments. Note that pattern matching and protocol decoding are not mutually exclusive, as some would lead you to believe.

For illustration purposes, once again consider the example of the abc attack. Suppose that the base protocol that the attack is being run over is the fictitious BGS protocol, and more specifically, assume that the attack requires that the illegal argument *foo* must be passed in the BGS Type field. To further complicate the situation, assume that the Type field is preceded by a field of variable length called BGS Options. The valid list of options are *fooh*, *mooh*, *torner*, and *buildo*. Using the simple or the stateful pattern-matching algorithm in this case leads to false positives because the option *fooh* contains the pattern that is being searched for. In addition, because the field lengths are variable, it would be impossible to limit such false positives by specifying search start and stop locations. The only way to be certain that *foo* is being passed in as the BGS type argument is to fully decode the protocol.

Not doing full protocol decodes can also lead to false negatives if the protocol allows for behavior that the pattern-matching algorithms have difficulty dealing with. For example, if the BGS protocol allows every other byte to be a NULL if a value is set in the BGS header, the pattern matchers would fail to see *fx00ox00ox00*. The protocol decode-enabled analysis engine would strip the NULLS and fire the alarm as expected, assuming that *foo* was in the Type field.

Pros:

- This method minimizes the chance for false positives if the protocol is well defined and enforced.
- This method can allow for direct correlation of an exploit.
- This method can be more broad and general to allow catching variations on a theme.
- This method reliably alerts on the violation of the protocol rules as defined.

Cons:

- This method can lead to high false positive rates if the RFC is ambiguous and allows developers the discretion to interpret and implement as they see fit. These gray area protocol violations are very common.
- This method requires longer development times to properly implement the protocol parser.



Heuristic-Based Analysis

Heuristic-based signatures use some type of algorithmic logic on which to base their alarm decisions. These algorithms are often statistical evaluations of the type of traffic being presented. A good example of this type of signature is a signature that would be used to detect a port sweep. This signature looks for the presence of a threshold number of unique ports being touched on a particular machine. The signature may further restrict itself through the specification of the types of packets that it is interested in (that is, SYN packets). Additionally, there may be a requirement that all the probes must originate from a single source. Signatures of this type require some threshold manipulations to make them conform to the utilization patterns on the network they are monitoring. This type of signature may be used to look for very complex relationships as well as the simple statistical example given.

Pros:

- Some types of suspicious/malicious activity cannot be detected through any other means.

Cons:

- Algorithms may require tuning or modification in order to better conform to network traffic and limit false positives.

Anomaly-Based Analysis

Anomaly-based signatures are typically geared to looking for network traffic that deviates from what is seen “normally.” The biggest problem with this methodology is to first define what “normal” is. Some systems have hard-coded definitions of *normal*, and in this case they could be considered heuristic-based systems. Some systems are built to learn *normal*, but the challenge with these systems is in eliminating the possibility of improperly classifying *abnormal* behavior as normal. Also, if the traffic pattern being learned is assumed to be *normal*, the system must contend with how to differentiate between allowable deviations and those not allowed or representing attack-based traffic. The work in this area has been mostly limited to academia, although there are a few commercial products that claim to use anomaly-based detection methods. A subcategory of this type of detection is the profile-based detection methods. These systems base their alerts on changes in the way that users or systems interact on the network. They incur many of the same limitations and problems that the overarching category has in inferring the intent of the change in behavior.

Interesting facts can be learned from trending data, and it is possible to detect ongoing attacks based on these algorithms. The information that these systems provide, however, is generally very nonspecific and requires extensive investigation to put it in the proper context.

Another type of anomaly-based detection is protocol-based anomaly detection. This method is closely related to the protocol decode methods mentioned above. Because protocol definitions are well-defined detection of protocol, anomalies do not require learning. An example of a protocol anomaly would be if a field in the protocol had an unexpected value in it. In some instances, the lines blur between methodologies because most protocol decode analysis engines alert the user to the presence of protocol violations that are not directly related to any known attack but are “anomalous” (for example, length-based buffer overflow detection). Therefore, in this example, the engine has attributes of an anomaly-based system.

Finally statistical anomalies may also be identified on the network either through learning or teaching of the statistical norms for certain types of traffic, for example, systems that detect traffic floods, such as UDP, TCP, or ICMP floods. These algorithms compare the current rate of arrival of traffic with a historical reference; based on this, the algorithms will alert to statistically significant deviations from the historical mean. Often, a user can provide the statistical threshold for the alerts.

These systems are generally marketed as the solution for IDS, but systems based on these techniques have been the subject of numerous academic research efforts with results showing limited success, and they are not production proven. The limited success observed in the academic research arena lends credence to the saying "If it sounds too good to be true, then it probably is."

Pros:

- If this method is implemented properly, it can detect unknown attacks.
- This method offers low overhead because new signatures do not have to be developed.

Cons:

- In general, these systems are not able to give you intrusion data with any granularity. It looks like something terrible may have happened, but the systems cannot say definitively.
- The signal-to-noise ratio is very low in most cases.
- This method is highly dependent on the environment in which the systems learn what *normal* is.

Conclusion

So which method is best? The answer is simply that it depends on what you are trying to do. The Cisco philosophy to date for network IDS (NIDS) has been to blend the use of pattern matching, stateful pattern matching, protocol decodes, and heuristic-based signatures. Cisco recommends that you match the appropriate tool to the problem that you are trying to solve or, in this case, the attack you are trying to detect. The mix is fairly heavy in favor of protocol decodes because most of Cisco's signatures are implemented in this manner, with heuristic-based signatures being the next most often used, followed closely by the pattern matches, and finally a number of protocol and statistical anomaly-based methods are used. Cisco will continue to research and monitor developments in the IDS arena and will incorporate new techniques as they become efficient, practical, cost-effective, and commercially viable.



Corporate Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 526-4100

European Headquarters
Cisco Systems International BV
Haarlerbergpark
Haarlerbergweg 13-19
1101 CH Amsterdam
The Netherlands
www-europe.cisco.com
Tel: 31 0 20 357 1000
Fax: 31 0 20 357 1100

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
www.cisco.com
Tel: 408 526-7660
Fax: 408 527-0883

Asia Pacific Headquarters
Cisco Systems, Inc.
Capital Tower
168 Robinson Road
#22-01 to #29-01
Singapore 068912
www.cisco.com
Tel: +65 6317 7777
Fax: +65 6317 7799

Cisco Systems has more than 200 offices in the following countries and regions. Addresses, phone numbers, and fax numbers are listed on the

Cisco Web site at www.cisco.com/go/offices

Argentina • Australia • Austria • Belgium • Brazil • Bulgaria • Canada • Chile • China PRC • Colombia • Costa Rica • Croatia
Czech Republic • Denmark • Dubai, UAE • Finland • France • Germany • Greece • Hong Kong SAR • Hungary • India • Indonesia • Ireland
Israel • Italy • Japan • Korea • Luxembourg • Malaysia • Mexico • The Netherlands • New Zealand • Norway • Peru • Philippines • Poland
Portugal • Puerto Rico • Romania • Russia • Saudi Arabia • Scotland • Singapore • Slovakia • Slovenia • South Africa • Spain • Sweden
Switzerland • Taiwan • Thailand • Turkey • Ukraine • United Kingdom • United States • Venezuela • Vietnam • Zimbabwe

All contents are Copyright © 1992–2003 Cisco Systems, Inc. All rights reserved. CCIP, CCSP, the Cisco Arrow logo, the Cisco Powered Network mark, Cisco Unity, Follow Me Browsing, FormShare, and StackWise are trademarks of Cisco Systems, Inc.; Changing the Way We Work, Live, Play, and Learn, and iQuick Study are service marks of Cisco Systems, Inc.; and Aironet, ASIST, BPX, Catalyst, CCDA, CCDP, CCIE, CCNA, CCNP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, the Cisco IOS logo, Cisco Press, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Empowering the Internet Generation, Enterprise/Solver, EtherChannel, EtherSwitch, Fast Step, GigaStack, Internet Quotient, IOS, IP/TV, iQ Expertise, the iQ logo, iQ Net Readiness Scorecard, LightStream, MGX, MICA, the Networkers logo, Networking Academy, Network Registrar, Packet, PIX, Post-Routing, Pre-Routing, RateMUX, Registrar, ScriptShare, SlideCast, SMARTnet, StrataView Plus, Stratm, SwitchProbe, TeleRouter, The Fastest Way to Increase Your Internet Quotient, TransPath, and VCO are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and certain other countries.

All other trademarks mentioned in this document or Web site are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company.
(0304R) ETMG 203149—BU 11/03