# Uncertainty: An extra layer of security For Unauthorized traffic based Web Services

*Parag Agarwal* *B. Prabhakaran* *Bhavani Thuraisingham*

Department of Computer Science, The University of Texas at Dallas

MS EC 31, PO Box 830688, Richardson, TX 75083

Email{pxa016500,praba,bhavani.thuraisingham@utdallas.edu}

**ABSTRACT**

*Distributed web services are under constant threat of being attacked from nodes, internal or external to the system. Internal attacks may result from hijacking of trusted web servers, resulting in loss/corruption of information, and Denial of Service (DoS) to clients. External attacks can occur from hijacking of trusted clients or malicious nodes leading to DoS to clients. The paper focuses on building an attack resistant framework for web services based on unauthorized traffic. Unauthorized traffic is a consequence of query driven session less HTTP-request/response message based web service applications, such as google.com. Dictionary.com etc. Unauthorized traffic based web service applications are supposed to have low response time. Unfortunately current mechanisms show lack of support for this traffic, since they add extra delay due to processing at intermediate nodes. The paper proposes a framework that optimizes the use of secure overlay services for unauthorized traffic. We add an extra layer of security around the web servers, which introduces uncertainty in the adversary's actions and is achieved by introducing dummy servers to the existing system, which appear as real servers to the clients or adversaries. The dummy servers act as traps if an adversary attacks assuming them to be real servers. Secure strategies have been proposed to implement the dummy servers. These strategies reduce the risk of hijacking and DoS attacks, minimize the changes to external infrastructure, can be easily integrated with existing security systems, do not promote ISP collaboration, and helps in scaling the system.*

## 1. INTRODUCTION

Web services are under constant threat of being abused by attacks such as the following:

- **Denial of service attack:** These attacks can be launched by sending flash crowds to web servers. Such flash crowds can be generated by concatenating distributed hosts across the network; hence the name distributed denial of service (DDoS) attack. DDoS attack can be launched in an unsophisticated way by running "rootkits" and worms [6] on these malicious nodes. The source of the flash crowds can be internal trusted nodes or external trusted/distrusted nodes. The trusted nodes are made to act malicious by hijacking them.
- **Information corruption:** Nodes internal to the system can be hijacked. Once hijacked, these nodes can disrupt the information at the system, and also send malicious information to the client side.

The malicious nodes are either internal or external to the system, hence the name internal and external attacks. In order to subvert such attacks we need to design framework that protects different kinds of web services. We identify web services on the basis of traffic generated by web applications as follows:

- **Unauthorized traffic:** This can be generated from query driven web services such as search engines (google.com), online dictionary (dictionary.com, word.com), map service (mapquest.com) etc. The interaction between client and server are HTTP-request and HTTP-response messages and do not require authentication mechanism prior to any information exchange. Such kind of traffic has low response times since the user expects quick responses.
- **Authorized traffic:** This can be generated from web-applications such as email service (gmail.com), e-commerce website (amazon.com) etc. These applications involve authentication mechanisms prior to information exchange. Such kind of traffic can have delay tolerance since authorization mechanisms involve identification processing.

Recent work on resistance against DDoS [1, 2, 3, 4, 5, 6, 7, 8, and 10] has focused on filtering traffic at the intermediate nodes. The filtering criteria can be traffic from unauthorized/illegitimate clients or absence of expected information in the message sent by the clients. Techniques such as [4] filter illegitimate traffic at the routers whenever the client identifies an attack being done using such a traffic. Similar techniques [5, 6, 7, and 10] have been developed to identify illegitimate traffic as weapons for launching DDoS attacks. Unauthorized traffic is based on short lived HTTP sessions that use TCP connections that tear down after each use. Subjecting the unauthorized traffic to such mechanisms would increase the response time, which is not a valid feature for such kind of applications. However, we observe that such solutions are more suited for authorized traffic based web applications since they are delay tolerant at the cost of security.

Frameworks that are based on Secure Overlay Services (SOS) [1, 2, 3, and 6] are suited for prevention against internal and external attacks. However, it is not suited for unauthorized and authorized traffic, as a result of the following problems:

- **Lack of traffic Support:** SOS-based techniques do not provide support for unauthorized traffic and filter such traffic assuming it to be a weapon to launch DoS attack. Therefore, we cannot rely on methods that support authorized traffic and use unauthorized traffic as a means to filter out an attack.
- **Hop Delay:** We could depend on techniques such as Secure Overlay Services (SOS) [1] to protect web services that support unauthorized traffic based applications. However, SOS has a potential problem for query driven application that requires fast responses. SOS adds hop-delays that will result in increase in response time.
- **Points of failure:** A node belonging to SOS can serve different web services. When such a node comes under attack, it malfunctions and affects all such systems. We need to remove such dependencies and points of failure, in order to secure the web services.

The above rationale motivates the design of secure framework for web services (supporting unauthorized traffic based applications) and is derived from an optimized SOS architecture. This paper contributes by proposing such a framework with the following design scope:

- **Risk Reduction:** Reduce the risk involved in hijacking and DoS or DDoS attacks by increasing the probability of failure of the adversary, which will act as the de-motivating factor for the adversary.
- **Minimal Change to existing Infrastructure:** The change in the system should least impact the external system such as the internet and transport protocols, and the routers. This also includes end-host, inter-ISP and intra-ISP cooperation.
- **Co-existence with existing security features:** The new layer of security should not impact the existing security features.
- **Support Scalability of the existing system:** The addition of the security feature should have minimal impact the scalability of the system
- **Fast Query responses:** The response time of the system should not be affected by the addition of security features.
- **No external cooperation:** The framework should result in increase in traffic due to cooperation between end-host/ISP, inter-ISP, and intra-ISP cooperation [4].
- **Ease of customization:** The framework requires minimal effort in customizing it for existing web service architectures [12], such as cluster-based, virtual clusters and locally distributed servers.

The proposed design introduces a set of dummy nodes/servers that are mapped to the real nodes/servers. The clients communicate either directly with the real servers or indirectly via the dummy servers. However, the clients assume direct communication with the real server. The mapping between the real and dummy servers is a secret and introduces uncertainty in launching internal or external attacks. We have shown the probability of failure of the adversary increases as the number of dummy servers is made equal to the number of real servers. Our design introduces (<) 1-hop delays which is an improvement over SOS design. Moreover it does not have points of failure since we do not use an infrastructure such as SOS.

## 2. DESIGN RATIONALE AND ASSUMPTIONS

The following assumptions are considered for our design:

- Servers within the system can communicate with each other over secure reliable high bandwidth links using their synchronization protocols.
- Any system node is capable of identifying a DoS attack and raises an alarm.
- During a node crash, the system will bring up another identical node with same IP address.

SOS hides the true identities of servers by introducing hidden paths to the target nodes. These paths are like levels in security. In case a web service uses a SOS, the target nodes are the web servers. Therefore, the web servers are hidden behind the SOS. We introduce an extra layer of nodes, similar to SOS, before the web servers. This layer is defined by dummy servers that are not used by the system for processing requests. We assume that the adversary views the dummy servers as real servers and will try to attack them. Therefore the dummy servers will act as traps and as a result the probability of its failure of the adversary will increase.

Figure 1 illustrates the interaction between client and N servers. We introduce K dummy servers to form the extra layer of security. N- K servers communicate directly and considered exposed and vulnerable to attack. Meanwhile K servers communicate indirectly with the real servers via the dummy servers.

We need the following features to implement the extra security layer:

- **Transparency:** The client nodes should visualize the dummy servers as real servers. In order to achieve such transparency we need mechanisms wherein the dummy servers should be able to process the client requests.
- **Secrecy:** In order to break down the system the adversary should be in a state to distinguish between dummy and real servers. In order to counter such an act, we need to maintain the identities of real and dummy servers a secret.

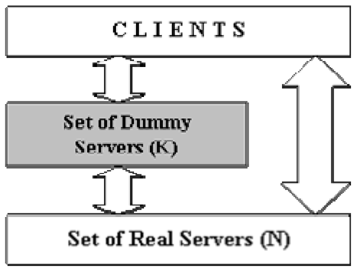

**Figure 1 Client-Server Interaction**

## 3. STRATEGIES TO SUPPORT THE DESIGN

The following section tries to explain strategies that can be used to maintain transparency and secrecy.

### 3.1 Transparency-based Strategy

Each dummy server is mapped to a real server. Clients communicate with the real server via the dummy nodes. In this case, the HTTP requests are received at the dummy nodes and forwarded to the real nodes. Similarly the HTTP response is received from the real server and forwarded to the client. HTTP uses TCP at the transport layer and hence would require two separate TCP sessions, referred in figure 2 as TCP1 and TCP2. Whenever the clients send out requests represented by TCP1-Req, the dummy server establishes a TCP connection with real server represented by TCP2-req. The real server responds with TCP2-reply and this is conveyed to client as TCP1-reply. It should be observed here that the TCP1 and TCP2 are different, and TCP2 represents TCP1 to the real server.

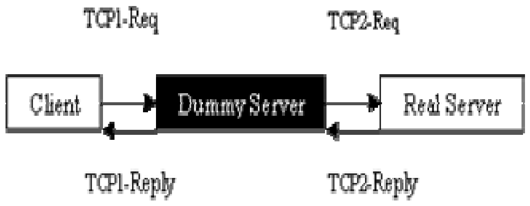

**Figure 2 TCP traffic handling**

The connection is made transparent since TCP2 is hidden from the client and it always assumes interaction with the real server. It is can be observed from Figure 2 that the HTTP requests and response over TCP1 and TCP2 stay invariant and are referred as TCP1-Req/TCP21-Req and TCP1-Reply/TCP2-Reply respectively. The client always visualizes interaction with the real server. We assume that, whenever a dummy server is under attack, all the current connections can be redirected to a back-up dummy server.

### 3.2 Secrecy-based Strategy

We add a set of dummy servers say K (≤ N, number of real servers). Each dummy server is mapped to a real server. The number K and the mapping is kept secret and can vary with time. Therefore the adversary will be unable to distinguish between the real and dummy servers. The IP address of the real and dummy servers is made public and is used by the clients for communication. Since the mapping secret we say that it is an *extra secret layer*. The secret in this layer results in uncertainty in the adversary's decision to attack the system.

In this approach we only have one-to-one mapping between the real server and the dummy server; in case there is a compromise of the dummy server, the identity of at most one server is revealed, leaving rest of the system safe.

.

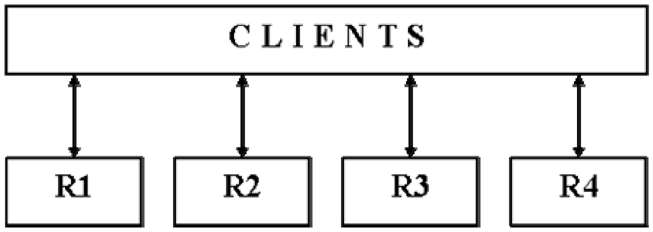

**Figure 3 (a) Insecure System**

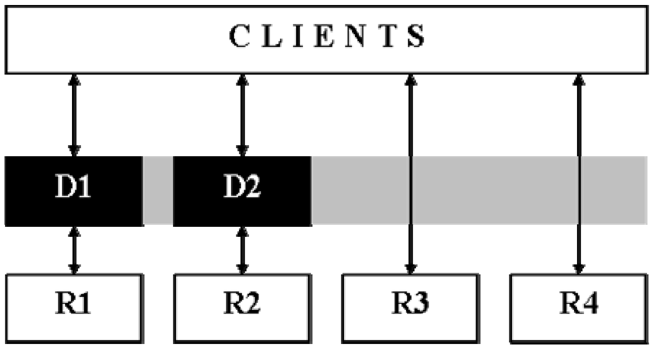

**Figure 3 (b) Secret Extra Layer for Figure 3 (a)**

Figure 3(b) represents a secrecy-based strategy for Figure 3(a). In this case we introduce two dummy servers with mapping (D1, R1) and (D2, R2). The client's perspective remains the same since the number of servers does not change. The following sections try to answer questions related to the use of secrecy-based strategy.

## 4. ATTACKS AND COUNTER MEASURES

The following subsections explain the attacks and their counter measures.

## 4.1 Internal Attack

Adversaries can capture servers and use the information stored at the server to attack other servers. Compromised servers can collude to attack and obtain the identity of other servers. As a result, compromise of servers can lead to attacks that compromise the remaining servers.

In order to make the system robust, we make the real servers incapable of discriminating between real and dummy servers. This can be achieved by following the secret-based strategy mentioned in section 3.2. The clients in this case are the servers themselves. Every real server is given a set S of IP address of servers they can communicate to, for synchronization purpose. The set S contains a combination of IP addresses of real servers or dummy servers. A real server by itself is unable to discriminate between the IP addresses of real and dummy servers. It assumes communication with all the real servers. In case the IP address of the dummy server is being used, communication with the real servers happens via the dummy server. During an attack, an adversary may compromise the real server. However since the real server is incapable of discriminating between the real and dummy servers, it will give no clue to the adversary.

## 4.2 External Attack

Adversaries can capture trusted clients and launch attacks on web servers that trust these clients. The following sub-sections explain as to how the security strategy can help prevent these attacks.

### *4.2.1 Delay Pattern Attack*

The adversary may try to pin point a dummy server by observing the 1-hop delay added due to the routing process. In order to handle this situation we can add random delays to responses to the requests that do not come from the dummy servers. So that the adversary will not get an exact knowledge about the real servers, thus defeating the adversary's purpose.

### *4.2.2 All node Attack*

A consequence of the combination of secret and transparency strategies is a layer of uncertainty that protects the real servers. However, some of the real servers are always exposed to the Internet. In order to overcome the uncertainty, an adversary may attack all the servers. We assume that the dummy servers are capable of recognizing such an attack and will raise an alarm to overcome such an attack. As a result, we will be able to protect the exposed real servers.

As long as the adversary is unable to distinguish between the true and dummy servers, the element of uncertainty always confuses the adversary. Therefore, the adversary is incapable of launching attacks [6] such as probing, adaptive flooding, and request attack. Prevention against external and internal attacks helps reduce the risk of DDoS by unauthorized traffic. The reduction in risk is evident in section 6.

## 5. CUSTOMIZATION FOR WEB SERVICES

A web service can be organized a locally distributed set of servers [12], which can be defined as cluster-based, virtual web-cluster and geographically distributed. The following schemes customize our solution to distributed web servers:

- **Cluster-based:** In such a scheme, we have a web-switch that acts as the router to the incoming and out-going requests/responses. The IP address of the web-switch is the publicly known to the clients. An attack can be launched on such a system by overtaking the web-switch. Since the web-switch has full knowledge of the IP address of all the servers, it will come under attack. Such an attack can be dissuaded by making the web-switch unable to distinguish between the real and dummy servers. It can be easily observed that the dependence on a single web-switch makes the system vulnerable to central point of failure. We can overcome this problem by assuming the web-switch is the target address for SOS. As a result, the clients observe the IP address of these SOS as the contact point to the web service. It has been shown in [1] that SOS offers significant guarantee against failures due to denial of service attack. Therefore in our improved design we observe the use of SOS in combination with our technique.
- **Virtual web-cluster:** In such a scheme, the web servers have the same IP address. We can customize this design to our solution by adding dummy servers with the same IP address. A significant advantage of such an approach can be triangular routing where the connection requests route from dummy to real servers. However the response and further communication takes place with the real server.
- **Locally Distributed servers:** In such a distribution, the IP address of the web servers is publicly known. We customize this scheme to our idea by adding the IP addresses to the pool of publicly known IP addresses. The IP addresses of the real servers that pair with the dummy servers are hidden. So the publicly known IP addresses consist of IP addresses of the dummy and real servers. It should be observed here that our original scheme does not hide the IP addresses of the real servers.

It can be observed that in the above customizations, that we do not change the existing web server systems. This is evident from the fact that the IP address remains invariant. It is the mapping between real and dummy servers that will determine the difference. However, this is not influenced by web service architecture. Thus we induce minimal changes and add ease in customization to the existing system. The following system analyses hop delay and risk of the proposed design.

## 6. ANALYSIS

This section is aimed at analyzing the framework in terms of reduction in risk and average hop delay.

### 6.1 Analysis of Risk

The probability of failure of the adversary indicates the effectiveness in security of our system. The following analysis calculates such a probability, where for a system of N servers with K dummy servers, the adversary attacks J servers without being able to distinguish between the real and dummy servers. Probability of failure of the adversary is $({}^{K}C_{J}/{}^{N}C_{J})$.

| N=5 | K= 1 | K=2 | K=3 | K=4 | K=5 |
|---|---|---|---|---|---|
| J =1 | 0.2000 | 0.4000 | 0.6000 | 0.8000 | 1.0000 |
| J =2 | 0 | 0.1000 | 0.3000 | 0.6000 | 1.0000 |
| J=3 | 0 | 0 | 0.1000 | 0.4000 | 1.0000 |
| J=4 | 0 | 0 | 0 | 0.2000 | 1.0000 |
| J=5 | 0 | 0 | 0 | 0 | 1.0000 |

**Table 1 Probability of failure for (N=5, varying J (1 to 5), K (1 to 5))**

Table 1 illustrates different values of probability of failure, given N = 5 for different values of K (1 to 5) and J (1 to 5). We can observe from table 1 that in order to have probability of failure (> .5) we need higher number of dummy servers in the system. For K = 5, we can guarantee 100% failure of the adversary. It can be observed that in order to attack the system, uncertainty will prevail for probability of failure (> .5). If the number of dummy servers being operated is made a secret, the probability of failure is a secret. Thus, uncertainty will prevail by maintaining such secret and will dissuade adversary. Therefore, when designing the web services framework we need not require the condition N = K. Therefore, we can claim that the SOS based extra layer of security can be *optimized*.

| N =4 | K= 1 | K=2 | K=3 | K=4 |
|---|---|---|---|---|
| J =1 | 0.2500 | 0.5000 | 0.7500 | 1.0000 |
| J =2 | 0 | 0.1667 | 0.5000 | 1.0000 |
| J=3 | 0 | 0 | 0.2500 | 1.0000 |
| J=4 | 0 | 0 | 0 | 1.0000 |

**Table 2 Probability of failure for (N=4, varying J (1 to 4), K (1 to 4))**

Table 2 illustrates risk analysis for Figure 3(b), with different values of probability of failure given, N = 4, K (1 to 4) and J (1 to 4). We can observe that the probability of failure for figure 3 is represented in column K =2. As observed in table 1, table 2 also shows us that for larger values of J, we need higher number of dummy servers to keep the probability of failure (> 0.5).

### 6.2 Analysis of average hop delay

The average hop delay [11] (for K dummy servers, N real serves, Average Traffic) can be calculated as follows:

$$\text{Average Hop Delay} = \frac{\sum_{i=1}^{K} AverageTraffic}{\sum_{J=1}^{N} AverageTraffic} \quad (1)$$

$$= \frac{K * AverageTraffic}{N * AverageTraffic} \quad (2)$$

$$= \frac{K}{N} \quad (3)$$

It can be observed from equation 3, that the average hop-delay is (K/N < 1) for (K < N), and hence is an improvement over SOS where average hop-delay is > 1. For figure 3(b), K = 2, N = 4 and the average hop delay is (2/4 = 0.5 < 1).

## 7. DISCUSSION

This section discusses our design in terms of advantages; the disadvantages with their counter measure, and compare our design to SOS.

### 7.1 Advantages

The advantages are listed as follows:

- **Handling DDoS attack:** Our design is capable of handling DDoS attack as it dissuades the adversary from launching internal and external attacks (see section 4).
- **Reduction in risk:** As we increase the dummy servers in the system, the likelihood of an adversary attacking a dummy server increases (see section 6). The idea of secret strategy will act as a dissuading factor to the adversary. Thus we might be able to avoid external and internal attacks.
- **Co-existence with existing security strategy:** Our idea can work in conjunction with SOS by registering the IP addresses of the dummy and real servers to the SOS. In such a case, our system introduces uncertainty in attack to a compromised SOS node.
- **No change in external system:** We do not make any major changes to the external protocols such as UDP, TCP or routers. Hence our design has minimal impact on the Internet infrastructure.
- **No External cooperation:** The design does not assume any end-host, inter-ISP and intra-ISP cooperation. Thus making the system free of signal exchange.
- **Reduced Response time:** In SOS where hop delay is a problem. However, the introduction of only a single layer adds a 1-hop delay. Since some of the servers are

directly exposed the 1-hop delay is not present. As a result, the response time will be reduced as compared to SOS technique. It can also be observed since no prior authentication is requires, we reduce the response time.

- **Ease of customization:** It has been shown in section 5 that our framework does not require any major changes in existing web service architectures.

### 7.2 Disadvantages and Counter-measures

**Scalability at increased cost**: As the number of requests of the clients increase, the load at the dummy servers will increase. This can be handled by increasing the computation power and memory capacity at the dummy servers. Since companies are willing to invest and the requisite hardware is getting cheaper, scalability is possible.

### 7.3 Comparison with Secure Overlay Services

Secure Overlay Services (SOS) hides the true identities of servers by introducing hidden paths to target servers. These paths are like levels in security. An attack on a node at a given level can be handled by removing the node from the SOS, and rerouting the traffic through another SOS node. In case a web service uses a SOS, the target nodes are the web servers. Therefore, the web servers are hidden behind the SOS. It has been shown in section 5 that the probability of failure of an adversary for SOS is 100%. However, by re-using the SOS ideas of hiding some of the servers in our own system, we introduce uncertainty in the adversary's desire to attack a web service (see section 5). We can compare SOS with our technique in terms of the following:

- **Traffic Support:** Our technique provides this support, and can also work in conjunction with techniques such as SOS to provide support against DDoS attacks.
- **Hop Delay:** Removing the extra layers and leaving certain servers directly exposed can reduce the overall delay. This evident from subsection 6.2 where it has been shown that the average hop delay (< 1).
- **Points of Failure:** The extra layer of dummy servers are system dependent, the points of failure are removed. A failure of a dummy server will not affect another web-application.

## 8. CONCLUSION

In this paper, we develop a secure framework for web service applications that generate traffic which does not require authentication as a pre-condition for communication. The design of the framework is derived by optimizing Secure Overlay Services (SOS). However, it does not suffer from the fundamental drawbacks of the SOS such as hop-delay, and lack of unauthorized traffic support. The paper proves that the framework avoids DDoS and information corruption attacks from nodes (external/internal) that are either acting malicious or have been compromised. The framework can be easily integrated with existing frameworks such as SOS, scalable at increased cost, does not require any change in external (internet protocols), requires no ISP collaboration and can be easily customized for existing web service architectures.

## ACKNOWLEDGEMENTS

The work is supported in part by US Army Research Office grant 48645-MA and National Science Foundation under Grant No. 0237954 for the project CAREER: Animation Databases.

## REFERENCES

[1] A. Keromytis, V. Misra, and D. Rubenstein, "SOS: Secure Overlay Services," in Proceedings of ACM, SIGCOMM'02, (Pittsburgh, PA), August 2002

[2] Angelos Stavrou Angelos, MOVE: An End-to-End Solution to Network Denial of Service. citeseer.ist.psu.edu/718186.html

[3] Angelos Stavrou Debra L. Cook, WebSOS: An Overlay-based System for Protecting (2005) , Web Servers From Denial of Service Attacks. citeseer.ist.psu.edu/720742.html

[4] A. Yaar, A. Perrig, and D. Song. SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks. In Proceedings of the IEEE Security and Privacy Symposium

[5] Drew Dean, Matt Franklin, and Adam Stubblefield, "An algebraic approach to ip traceback," in Network and Distributed System Security Symposium, NDSS '01, February 2001

[6] D. G. Andersen. Mayday: Distributed filtering for Internet services. In USITS, Seattle, WA, 2003. http://citeseer.ist.psu.edu/andersen03mayday.html

[7] J. Mirkovic, G. Prier and P. Reiher, "Attacking DDoS at the Source", in 10 ' IEEE International Conference on Network Protocols, Paris, France, November 2002.

[8] Jelena Mirkovic, Peter Reiher, A taxonomy of DDoS attack and DDoS defense mechanisms, ACM SIGCOMM Computer Communication Review, v.34 n.2, April 2004

[9] M. T. Goodrich, Efficient Packet Marking for Large-Scale IP Traceback, in: Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS), 2002,

[10] P. Ferguson and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks Which Employ IP Source Address Spoofing" IETF, RFC 2267, Janurary 1998.

[11] Robert S. Cahn, "Wide Area Network Design", Morgan Kaufmann, 1998

[12] Valeria Cardellini, Emiliano Casalicchio, Michele Colajanni, and Philip S. Yu. The state of the art in locally distributed web-server systems. ACM Computing Surveys, 34(2):263-311, June 2002