May 2007

Application Denial of Service

Is it Really That Easy?

Shay Chen

Agenda

- Introduction to Denial of Service Attacks
- Application Level DoS Techniques
- Case Study – Denial of Service Testing
- Mitigation
- Summary

About Hacktics

- Security Services Company
- Provides wide range of services with focus on the application security field.
- Relies on vast experience in application level penetration testing and secure development

Hacktics offers unique expertise in the technology and methodology of application security, together with out of the box thinking abilities and a keen understanding of the operational patterns of Hackers.

Introduction

Overview

- Denial of Service (DoS) is the act of performing an attack which prevents the system from providing services to legitimate users
- Denial of Service attacks take many forms, and utilize many attack vectors
- When successful, the targeted host may stop providing any service, provide limited services only or provide services to some users only

The Drive

- Denial of Service attacks are usually conducted by few types of attackers:
  - The “Fun” Hackers (Because they can…)
  - Activists (Anarchists, Anti Globalization, etc.)
  - Terrorists (Aid causes of war)
  - Competitors (Mostly “grey area” industries such as sex, gambling, etc.)
  - Military
- While some other types of DoS hackers exist, they are negligible

Moving to the Present

- Simple, grand scale DoS attacks such as used before are rarely found
- Inherent exploits and simple vulnerabilities are already fixed
- Modern technologies (firewalls, IPS) make DoS even harder
- The desire for DoS, however, has not diminished

DDoS – Distributed Denial of Service

- With improvement in DoS protection, the next step was to simply exhaust the bandwidth of desired hosts
- This is achieved by utilizing a “Distributed Denial of Service” attack (DDoS).
- With DDoS, every member of the attack generates relatively small amounts of traffic. The combined result overwhelms the remote system.

Application Level DoS Techniques

Overview

- Application layer DoS attacks are evolving as part of the evolution of application attacks
- The denied service is the application itself (rather than the host) – effectively preventing usage of the system.
- Take advantage of flaws in the code to perform the DoS
- The benefit for the attacker – does not require the same effort to achieve as a DDoS attack

Overview

- DoS can be achieved in various ways:
  - Application Crashing
    - Memory Access Violation (Buffer Overflow)
    - Various Exceptions
  - Data Destruction
  - Resource Depletion
    - Memory
    - CPU
    - Bandwidth
    - Disk Space

Application Crashing

- Common way of performing a Denial of Service attack
- In many cases, certain types of inputs may yield an error in the application which it did not anticipate, and will cause it to crash:
  - Buffer Overflows
  - Malformed data – causing parser exception
    - Terminating with error
  - SQL Injection (; shutdown --)

Data Destruction

- One way to cause a DoS attack is by tampering with the data instead of the service itself
- If a site is vulnerable to SQL Injection, for instance, it may be possible to DELETE all data from all tables
- Although the Web site will keep being ‘online’, it will actually be useless without the information from the Database

Data Destruction – Example

- Intentional User Lock
  - Any web application login page
  - Taking advantage of the application security mechanisms to cause DoS by abusing the login failure user lock mechanism
  - Intentionally failing multiple login attempts with each possible username, will eventually result in DoS, since all the application users will be locked

Resource Depletion

- Resource Depletion is a technique of performing DoS attacks on any site or application (unvulnerable to trivial DoS)
- Classical Resource Depletion simply utilizes very large amounts of attacker resources
- Sophisticated attacks pinpoint the weak points of the application to achieve maximum effect using minimal resources

Resource Depletion – Example #1

- CPU Consumption
  - A large Forums application
  - Contains millions of messages
  - Allow performing sophisticated regular expression searches
  - An attacker can easily create complicated regular expressions which consume a lot of CPU each time a search is initiated
  - The attacker then writes a script to launch this request over and over again

Resource Depletion – Example #2

- CPU Consumption – The SQL Injection version
  - When SQL Injection is possible – can be used for DoS even without permissions to Shutdown or Delete
  - Creating very intense nested queries does the trick:

```
SELECT A1.*, B1.*
FROM A AS A1, B AS B1
WHERE EXISTS (SELECT A2.*, B2.*
               FROM A AS A2, B AS B2
              WHERE A1.AID = A2.AID)
  AND EXISTS (SELECT B3.*, A3.*
               FROM B AS B3, A AS A3
               WHERE B1.BID = B2.BID)
```

Resource Depletion – Example #3

- Memory Consumption
  - A Web Mail Application
  - Allows uploading files for attachment
  - All attachments are stored in the application's memory until the 'Send' button is sent
  - There is no limitation on the size or number of attachments
  - Assuming the hacker has a lot of bandwidth, the hacker can upload thousands of attachments, consuming all free memory in the machine

Resource Depletion – Example #4

- Disk Consumption
  - Any web application
  - Detailed logging is used for each application error
  - An attacker identifies a light-weight request which can generate a few KB of log
  - The attacker then repeats this until the Disk is full
  - Application behavior once Disk is full is unexpected:
    - Application might terminate when not being able to write to a file
    - If the files are located on the system partitions, the entire machine might crash

Resource Depletion – Example #5

- Network Consumption
  - Any web application
  - Attacker has wide Internet connection
  - Attacker identifies small requests which result in large amounts of data (Display all items in system)
  - Attacker can then launch the request over and over again, causing the database to send large amounts of data back to the web server in each request (potentially exhausting the connection pool as well)

Distributed Application Denial of Service

- Taking application resource consumption attacks to the next level
- Allows extending the effect of application DoS attacks when the resource consumption is slow
- However, DADoS does ***not*** rely on the same magnitudes of normal DDoS attacks
- Normally, up to severl dozens hosts is all that is required.

Case Study – Denial of Service Testing

(Taking Down a Corporate Site with Just Three Laptops)

Denial of Service Testing

- Part of the services offered by Hacktics
- Simulates high-end denial of service attacks
- Allows the organization to estimate the risk of Denial of Service attacks for their internet facing infrastructure
- Performed off-hours to avoid denial of service for real users
- DoS attacks include network, infrastructure and application.

Denial of Service Testing Case Study

- Examination of a recent DoS Test conducted for one of our customers (*Client X*).
- General Information for Client X:
  - Global company with branches in Israel, Europe and the USA
  - Internet site contains both public sites (corporate information, products, etc.) and private sites (users self service portal, users information, etc.)
  - Overall level of security with the customer is high with an active in house security group

DoS Testing Case Study Overview

- Technical Background
  - Internet Connectivity – 3x50Mbps lines with load balancing. ISPs provide Cisco (Riverhead) based Anti DDoS solutions
  - Public Sites
    - ~10-15 Web Servers (Mostly IIS)
    - Databases
    - Mail Relay

DoS Testing Case Study Overview

- Technical Background (Continued)
  - Private Sites
    - Authentication Gateway (Reverse Proxy)
    - ~20 Web Servers (Mostly IIS)
    - Backend Servers
    - Databases
  - Security of sites is high – mostly up to date patches, hardened systems, minimal firewall rules. Additionally, an IPS system monitors the requests

Testing Environment

- The environment was set up to permit as “clean” as possible testing environment
- The entire site was taken off the internet at night time
- A separate dedicated connectivity was set up
- Testing team was equipped with:
  - 5 Laptops
  - 3 ADSL 5M/256k Lines

Initial Tests

- Network/Infrastructure Tests yielded nothing significant:
  - Patching combined with IPS prevents exploitation of known vulnerabilities
  - Utilization of testing infrastructure (768kbps upload) prevented actual DDoS (against 150Mbps)
- Focus was then shifted to the application level:
  - Attempting to find Single-Request DoS
  - Attempting to find Resource Depletion DoS

Resource Depletion DoS Tests

- Test Plan:
  - Identify Resource Intensive Pages (Both public and private sites)
  - Create scripts for generation of requests, overcoming several obstacles:
    - DoS/DDoS Protection Solution (Cisco/Riverhead)
    - Authentication Gateway (Reverse Proxy)
    - Dynamic URLs
  - Execute scripts from several hosts

Identifying Resource Intensive Pages

- Browsing the application as a regular user and as an attacker (using interception proxy)
- Identifying pages/operations which either:
  - Seem to take longer to complete than other pages in the system
  - Perform complicated tasks (data mining, communication with external environments, etc.)
  - Behave slower when provided with invalid input.

Script Generation

- Perl script capable of
  - Generating large number of requests per second
    - Utilizing 90 simultaneous threads
    - Up to several hundred requests/second
  - Authentication
    - Authenticate via Authentication Gateway
    - Maintain Authenticity
  - Request Adaptation
  - Request Mutation

Authentication Gateway Challenge

- Works as a reverse proxy
- Requires authentication information for accessing protected servers
- The script should be able to authenticate, retrieve relevant information (cookie, URLs, etc.) and embed it into the code
- The authentication gateway in use was non standard, including part of the session information in the URL, requiring additional request adaptation

Anti DDoS Protection

- Client X uses Cisco (Riverhead) Anti DDoS protection in all ISPs
- Identifies multiple similar requests from different sources and blocks the request
- To overcome this obstacle, script was fitted with auto mutating capabilities
- Each request sent was different, using a combination of changes irrelevant to the application (redundant parameters, different headers, etc.)

Launching the Attack

- With the script set up it was time to run the attack
- The script was preloaded with relevant resource consuming requests and launched against site
- Initially every server was tested separately
- Later on – scripts were launched against all servers

Results

- DoS was successful to all systems but one
- Two applications crashed completely after a few dozen requests only
- Most other applications stopped responding after 5-15 minutes of script execution from up to three laptops (though with most a single laptop was sufficient)
- Main application DoS cause were CPU exhaustion.

Results (Cont'd)

- Additional results included:
  - Authentication Gateway refused accepting new connections after 2000 SSL handshakes were completed (and never freed)
  - The Load Balancer crashed after its log space was full
- Eventually, using 3 laptops simultaneously, with total upload of 768kbps (256kbps/laptop) all internet sites of the customer (except one) were unavailable
- No need for botnets or significant bandwidth!

Application Denial of Service Mitigation

Mitigation – Code Level

- There are techniques to avoid some DoS attacks at the code level:
  - Perform thorough input validations. Expect for the worst!
  - Avoid highly CPU consuming operations
  - Try to create as little as possible bottlenecks
  - Avoid operations which must wait for completion of large tasks to proceed
    - Split operations to chunks
    - Set timeout timers for unreasonable time

Mitigation – Deployment

- Prepare for performance peaks
  - More Load Balancing
  - Caching
- Always separate the data disks from the System disks

Summary

Summary

- DoS attacks are used by hackers, activists and terrorists to prevent legitimate usage.
- With the evolution of the Internet and its security, network based DoS attacks became rare.
- DDoS attacks may allow DoSing almost every site in the world, but requires significant resources
- **Application DoS attacks allow for efficient DoS with only little resources at hand, and thus pose a serious threat to organizations**

Thank You

Q & A

shay@hacktics.com