情報保護學會論文誌
第17卷 第4號, 2007. 8

블록 기반 파일 결함 주입 기법을 이용한 소프트웨어 보안 테스팅

최 영 한,[1†‡] 김 형 천,[1†] 홍 순 좌[1†]
[1]국가보안기술연구소

Software Security Testing using Block-based File Fault Injection

Young-Han Choi,[1†] Hyoung-Chun Kim,[1†] Soonjwa Hong[1†]
[1]National Security Research Institute

요 약

본 논문에서는 파일에 결함을 주입하는 기법을 이용하여 보안 테스팅(security testing)을 수행하는 방법론을 제안한다. 본 논문에서 제안한 방법론은 파일 내의 여러 필드(field)들을 묶어 블록(block)으로 처리하는 파일 포맷을 대상으로 필드를 고려하여 결함 주입 기법을 수행함으로써 소프트웨어의 취약점을 발견한다. 해당 방법론은 파일 데이터의 변경으로 발생할 수 있는 메모리 처리 관련 취약점에 초점을 맞추고 있다. 파일에 결함을 주입할 때 필드를 고려하면 파일을 파싱하는 과정에서 발생할 수 있는 파일 포맷 불일치의 에러 처리를 줄일 수 있는 장점이 있다. 본 논문에서는 블록으로 처리하는 파일 포맷 중 대표적인 파일 포맷인 이미지 파일에 대해 해당 방법론을 적용하였다. 이와 함께 이미지 파일에 대해 자동으로 결함을 주입할 수 있는 도구인 ImageDigger를 구현하였다. ImageDigger를 이용하여 WMF, EMF 이미지 파일 포맷에 대해 결함 주입을 수행하였으며 10 종류의 서비스 거부 취약점을 발견하여 원인을 분석하였다. 해당 방법론은 블록을 기반으로 파일을 처리하는 대표적인 파일 포맷인 MS Office와 이외의 파일 포맷에 대해서도 적용 가능하다.

ABSTRACT

In this paper, we proposed the methodology for security testing using block-based file fault injection. When fault is inserted into software, we consider the format of file in order to efficiently reduce the error that is caused by mismatch of format of file. The Vulnerability the methodology focuses on is related to memory processing, such as buffer overflow, null pointer reference and so on. We implemented the automatic tool to apply the methodology to image file format and named the tool ImageDigger. We executed fault-injection focused on WMF and EMF file format using ImageDigger, and found 10 DOS(Denial of Service) in Windows Platform. This methodology can apply to block-based file format such as MS Office file.

Keywords : *Software testing, fault injection, file fuzzing*

Ⅰ. 서 론

소프트웨어는 유저인터페이스(User Interface), API (Application Programming Interface), 파일과 같이 다

접수일: 2006년 12월 20일; 채택일: 2007년 6월 2일
† 주저자, yhch@etri.re.kr
‡ 교신저자, yhch@etri.re.kr

양한 곳에서 입력을 받아들인다[1,2]. 이들 입력 엔트리 중 파일은 소프트웨어에서 생성된 데이터를 저장하기 위해 사용된다. 파일의 경우 일단 열어보아야 내용을 확인할 수 있으므로 이러한 점을 악용하여 파일 데이터에 악성코드를 삽입한 후 사회 공학적 기법이나 파일 자동 로딩과 같은 방법으로 시스템을 공격한다. 예를 들어 이미지 파일인 경우 화려한 문서를 제공하는 인터넷 환경에서 없어서는 안 될 필요 파일이기에 대부분의 경우 차단되지 않고 자동으로 메모리에 로드되어 파싱된다. 그래서 이미지 파일에 대해 이러한 점을 역이용한 공격이 늘어나고 있는 추세이다[3].

본 논문에서는 파일 결함 주입 기법을 이용하여 보안 테스팅(security testing)을 수행하는 방법론을 제안한다. 본 논문에서는 보안 테스팅 중 비정상 입력으로 인해 발생하는 보안 문제에 초점을 맞춘다[4]. 본 논문에서 제안한 방법론은 파일 내의 여러 필드(field)들을 묶어 블록(block)으로 처리하는 파일 포맷을 대상으로 필드를 고려하여 결함을 주입하여 테스팅을 수행함으로써 소프트웨어의 취약점을 발견한다. 블록은 블록 내 길이 필드가 영향을 미치는 영역을 지칭하며, 파일은 하나 혹은 그 이상의 블록으로 나누어진다. 파일에 결함을 주입할 때 필드를 고려하면 파일 파싱 과정에서 발생할 수 있는 파일 포맷 불일치의 에러 처리를 줄일 수 있는 장점이 있다. 해당 방법론은 파일 데이터의 변경으로 발생할 수 있는 메모리 처리 관련 취약점에 초점을 맞춘다. 데이터가 변경된 파일을 입력으로 받아들이는 소프트웨어인 경우 파일 파싱을 수행할 때 발생할 수 있는 예외를 적절히 처리하지 못하면 이상현상을 유발할 수 있으며, 해당 이상현상이 보안상 악의적인 목적으로 이용될 수 있다. 해당 이상현상은 대상 소프트웨어가 더 이상 서비스를 할 수 없도록 하므로 서비스 거부 취약점(Denial of Service)으로 분류된다[5,6,7]. 본 논문에서는 블록으로 처리하는 여러 파일 포맷 중 이미지 파일에 대해 해당 방법론을 적용하였다. 이와 함께 이미지 파일에 대해 자동으로 결함을 주입할 수 있는 자동화 도구인 ImageDigger를 구현하였다. ImageDigger를 이용하여 WMF, EMF 파일을 대상으로 결함 주입 기법을 수행하여 10개의 서비스 거부 취약점을 발견하였다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 결함 주입 기법에 대한 기존의 연구에 대해 논하고, 3장에서는 블록 기반 파일 결함 주입 방법론을 제안한다. 4장에서는 앞 장에서 제안한 기법을 블록 기반 파일 포맷 중 이미지 파일에 대해 적용하며, 해당 방법론을 구현한 ImageDigger에 대해 논한다. 5장에서는 결론을 맺는다.

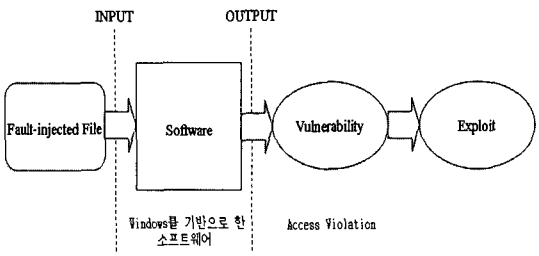
Ⅱ. 관련연구

결함 주입 기법[8]은 소프트웨어 테스팅 중 블랙박스 테스팅의 일종으로 주로 COTS(Commercial Off-The-Shelf)를 대상으로 프로그램의 내부 구조 및 코드에 대한 사전 지식 없이 테스트를 수행하는 방법이다. 결함 주입 기법은 프로그램의 외부에 노출된 인터페이스를 대상으로 하며, 대상은 API, 파일, 커널, 유저 인터페이스(User Interface)가 있다[1].

API를 대상으로 수행되는 결함 주입은 API의 파라미터에 비정상적인 값을 주입하여 대상 소프트웨어의 이상현상 발생 여부를 테스팅하는 방법이다. 대표적인 프로젝트로 Ballistar[9,10]가 있다. 결함이 주입된 데이터가 자체 에러처리 메커니즘에 의해 에러로 간주되는 것을 방지하기 위해 함수 파라미터의 데이터 타입을 고려하여 입력을 주입한다.

파일을 대상으로 하는 결함 주입은 파일 데이터에 이상현상을 유발시킬 수 있는 데이터를 주입하여 파일을 생성한 후 대상 소프트웨어가 읽게 하여 이상 현상을 유발시키는 방법이다. 대표적인 것으로 FileFuzz[11]와 SPIKEfile[12]이 있다. FileFuzz는 정상적인 파일에 대해 임의의 위치에 임의의 값을 주입하는 프로그램이다. 비정상으로 생성한 파일을 자동으로 실행을 시킴으로써 이상현상을 발생시킨다. SPIKEfile은 네트워크 퍼징을 수행하기 위해 구현한 SPIKE을 변형하여 파일에 초점을 맞추어 퍼징을 수행할 수 있도록 구현한 프로그램이다. 사용 방법은 SPIKE와 유사하며 파일 포맷에 대해 직접 바이너리 값들을 입력으로 사용한다. 본 논문의 ImageDigger는 파일을 대상으로 결함 주입을 수행하는 것으로 임의의 위치에 대해 테스팅을 수행하는 FileFuzz와 SPIKEfile과는 달리 분석한 파일의 필드를 대상으로 결함 주입을 수행하여 소프트웨어의 에러 처리 메커니즘에 의해 에러로 처리되는 비율을 줄이도록 하였다.

커널에 대해 결함 주입을 수행하는 방법은 사용자 레벨에서 직접적으로 커널 영역으로 접근할 수 없기 때문에 커널에 접근할 수 있는 시스템콜을 이용한다. sysfuzz[13]는 시스템콜과 해당 파라미터의 값을 임의의 값으로 변경시킴으로서 이상현상을 유발시키는 프로그램



〔그림 1〕 결함 주입 기법 전체 흐름

이다. 시스템콜을 통해 전달된 값들이 커널 영역에서 사용되므로 이상이 발생하였을 경우 커널 패닉(kernel panic)에 빠지게 된다.

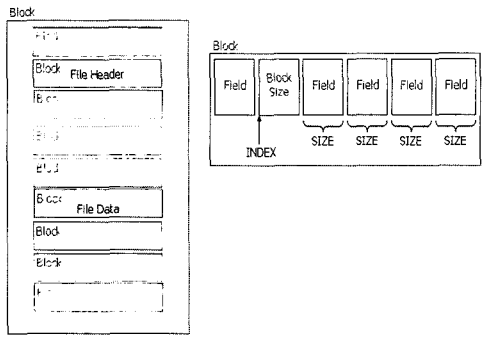
유저 인터페이스(User Interface)는 사용자의 입력을 받아들이는 부분으로 다양한 값들이 설정될 수 있으며 위험이 가장 많이 노출된 부분이다. Windows의 GUI(Graphic User Interface)에 대해 결함 주입을 수행하는 방법도 있다[14]. GUI에 대해 자동으로 결함 주입을 수행하기 위해 GUI가 수행될 때 호출되는 다양한 메시지 관련 함수에 대해 메시지 값과 여타의 다른 값들을 변경시킴으로써 취약성을 분석한다.

Ⅲ. 블록기반 파일 결함 주입 기법 방법론

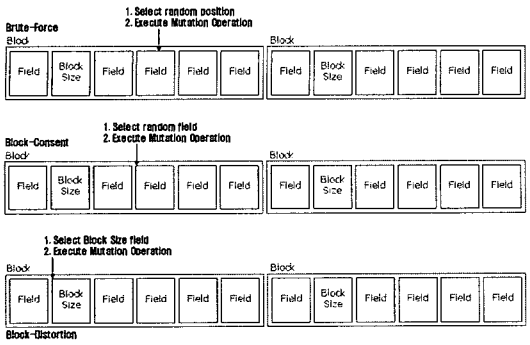
〔그림 1〕은 본 논문에서 수행하는 결함 주입 기법 전체 흐름을 나타낸다. 결함 주입 기법은 결함을 포함하고 있는 입력을 주입하여 출력으로 발생하는 이상현상을 모니터링 하는 방법이다. 결함을 유발하기 위해 본 논문에서는 입력으로 블록 기반 파일에 대해 블록을 고려하여 결함을 주입하였다. 대상 소프트웨어가 Micosoft사의 Windows를 기반으로 하는 경우 출력으로 Access Violation(AV) 예외를 모니터링한다. AV는 버퍼 오버플로우(buffer overflow)나 Null pointer 역참조와 같은 메모리 처리 관련 예외에 대해 발생하며, 해당 AV는 보안 관련 취약점 중 서비스 거부 취약점으로 분류된다[5,6,7]. 서비스 거부 취약점으로 분류되는 이유는 AV가 가 발생하면 대상 소프트웨어는 더 이상 자신의 역할을 수행할 수 없기 때문이다. AV는 Vulnerability 단계이며 해당 단계에서 악의적인 목적으로 시스템을 이용하기 위해 Exploit을 생성한다. Exploit이 가능한 Vulnerability는 여러 가지 분석으로 판단된다. 해당 단계는 많은 경험을 필요로 하며 자동화 및 일반적인 방법론을 적용하는 것은 불가능하다. 따라서 본 논문에서는 결함을 주입한 파일을 소프트웨어에서 실행을 시킨 후 AV 관련 Vulnerability를 탐지하는데 초점을 맞춘다[15,16]. AV는 길이와 관련된 필드가 관련 많으며 본 논문에서 제안한 블록 기반 파일 주입 기법도 필드 중 수와 관련된 부분에 대해 결함을 주입하였다. 블록의 필드를 기반으로 결함을 주입하게 되면 소프트웨어에서 파일 처리 시 발생할 수 있는 에러를 줄일 수 있는 장점이 있다. 이와 같은 이유는 일차적으로 파일이 블록으로 나누어질 때 파일 포맷과 일치하므로 파일 불일치 에러 메커니즘은 통과할 수 있기 때문이다. 결함을 주입한 후 Output 모니터링은 WinDBG[17]을 이용하여 AV가 발생하였을 때 Vulnerability를 탐지하도록 하였다.

본 논문에서 결함 주입 기법의 입력으로 사용될 파일은 전체적으로 파일 헤더(file header)와 파일 데이터(file data)로 나누어진다. 파일 헤더는 파일 자체의 정보를 저장하는 부분으로, 파일 크기나 파일 생성 일자 같은 정보가 저장된다. 파일 데이터는 파일의 실제 데이터가 저장되는 부분으로 파일의 내용이 저장된다. 예를 들어 이미지 파일의 경우 그림에 대한 정보가 저장된다. 파일 헤더는 파일 자체의 정보를 나타내기 때문에 고정된 길이를 가지는 경우가 많으나, 파일 데이터의 경우 다양한 정보를 저장하기 위해 다양한 데이터 구조체가 포함된다. 파일 데이터를 저장하는 방법 중 블록을 사용하는 방법은 역할이 비슷한 데이터를 블록으로 묶어 처리하는 블록 기반 파일 포맷이 있다. 예를 들어 PE 파일 포맷[18]은 여러 개의 섹션으로 나누어져 처리되며, Microsoft Excel 파일[19]은 셀을 표현하기 위해 수많은 레코드로 나누어져 파일로 저장된다.

〔그림 2〕는 파일을 블록으로 나눈 것이다. 블록이란 블록 내 길이 필드가 영향을 미치는 영역을 의미한다. 따라서 파일은 하나 혹은 그 이상의 블록으로 나누어진



〔그림 2〕 일반 파일 포맷에서의 블록 나누기.



〔그림 3〕 블록 기반 결함 주입 기법의 3가지 방법

다. 블록 내의 필드는 개수는 다양하나, 항상 블록의 크기를 나타내는 블록 사이즈 필드는 항상 포함하고 있어야 한다. 블록 내 필드의 수와 길이는 파일 포맷에 따라 정해진다.

본 논문에서 제안한 블록 기반 결함 주입 기법 방법론은 〔그림 3〕과 같다. 결함을 주입하는 위치와 길이에 따라 3가지 방법으로 나누어진다. Brute-Force는 파일 포맷을 고려하지 않는 결함 주입 기법을 적용하는 일반적인 방법이다. Block-Consent와 Block-Distortion은 본 논문에서 제안한 블록 기반 결함 주입 기법이다.

- Brute-Force : 결함 주입 위치와 길이를 임의의 값으로 설정하여 수행하는 방법으로 파일 포맷의 필드에 대해 고려하지 않는다. 파일 포맷에 대한 고려 없이 결함 주입을 수행하기 때문에 테스팅을 쉽게 할 수 있는 장점이 있으나, 파일 포맷 불일치로 발생하는 에러가 발생할 가능성이 높다.
- Block-Consent : 결함 주입 위치를 필드의 시작 위치로, 길이를 필드의 길이로 설정하는 방법으로 파일 포맷의 타입을 고려하여 결함 주입을 수행한다. 단 해당 방법에서는 블록의 길이 필드에 대해서는 결함을 주입하지 않는다. 블록 길이 필드에 대해 결함을 삽입하지 않으므로 일차적으로 블록에 대한 파싱은 수행되는 장점이 있다.
- Block-Distortion : Block-Consent 방법과 동일하나 블록의 길이에 대해서만 결함 주입을 수행한다. 해당 방법은 블록의 길이를 변경하여 소프트웨어가 이상 블록에 대해 파싱을 수행하도록 한다. 블록 길이 필드를 변경하기 때문에 예상치 못한 블록을 생산하여 보안 테스팅을 수행할 수 있는 장점이 있다.

Ⅳ. Case Study: 이미지 파일에서의 블록 기반 파일 결함 주입 기법

본 장에서는 블록 기반 파일 중 취약점이 많이 발생한 이미지 파일 포맷에 대해 본 논문에서 제안한 방법을 적용하였다. 이미지 파일의 경우 웹브라우저에서 자동으로 로딩되어 파싱되므로 취약점이 지속적으로 보고되는 파일 포맷이다. PE 파일 포맷과 MS Office 파일 포맷은 대표적인 블록 기반 파일 포맷이며 본 논문에서 제안한 방법론은 해당파일 포맷들에 대해서도 적용 가능하다.

4.1. 이미지 파일 선정

이미지 파일은 웹브라우저에서 자동으로 파싱되는 대표적인 파일 포맷이다. 따라서 이와 관련하여 많은 취약점이 발표되고 있다. 〔표 1〕의 취약점은 CVE에서 2006.08까지 발표된 취약점으로 윈도우즈와 관련 있는 것을 분류한 것이다[20]. 이미지 파일의 경우 〔표 1〕과

〔표 1〕 이미지 파일 관련 취약점(~2006.08)

파일 포맷	관련 취약점	
WMF	CVE-2006-4071, CVE-2006-0143, CVE-2005-4560, CVE-2005-2123, CVE-2004-0209, CVE-2002-0340	CVE-2006-2376, CVE-2006-0020, CVE-2005-2124, CVE-2005-0954, CVE-2003-0906,
EMF	CVE-2006-2376, CVE-2005-0803, CVE-2003-0906	CVE-2005-2123, CVE-2004-0209,
JPG	CVE-2006-4066, CVE-2004-0200,	CVE-2005-3312, CVE-2001-0712
PNG	CVE-2006-4066, CVE-2006-0025, CVE-2004-1244,	CVE-2006-0033, CVE-2005-1211, CVE-2002-1185
GIF	CVE-2006-0007, CVE-2005-3975, CVE-2005-3310, CVE-2003-1048	CVE-2005-4426, CVE-2005-3312, CVE-2005-0562,
BMP	CVE-2006-0006, CVE-2004-1922, CVE-2004-0566,	CVE-2005-3310, CVE-2004-1049, CVE-2000-0415

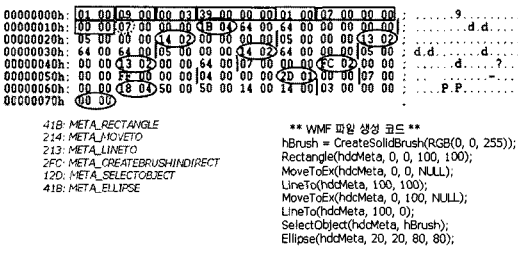
같이 지속적으로 취약점이 보고 되고 있으며, 웹브라우저에서 사이트 접속시 자동으로 로딩되므로 보안상 위험한 파일 포맷이다. 따라서 본 논문에서는 제안한 방법론을 적용하기 위한 파일 포맷으로 이미지 파일 포맷을 선정하였다.

이미지 파일은 크게 벡터(vector) 방식과 래스터(raster) 방식이 있다[21]. 벡터 방식은 이미지를 표현할 때 벡터로 표현하는 방식으로 이미지를 표현하기 위한 그리기 정보로 이미지를 표현하는 방식이다. 예를 들어 WMF 파일 포맷의 경우 함수와 관련 파라미터가 파일 내 레코드로 저장된다. 이와는 반대로 래스터 방식은 이미지를 표현할 때 점에 대한 RGB 값을 저장하여 표현하는 방식이다. 대표적인 이미지 파일 포맷은 다음과 같다.

- 벡터 방식 : WMF(Windows MetaFile), EMF (Enhanced MetaFile)
- 래스터 방식 : BMP, GIF, JPG, PNG

래스터 방식의 이미지 파일은 파일 헤더와 파일 데이터가 하나의 블록 혹은 제한된 수의 블록으로 묶여진다. 하지만 벡터 방식의 이미지 파일은 파일 데이터가 함수 정보를 저장되는 관계로 수많은 블록으로 나누어 질 수 있다. 따라서 본 논문에서는 제한된 수의 블록으로 묶여지는 래스터 방식의 이미지 파일이 아니라 많은 수의 블록으로 나눌 수 있는 벡터 방식의 이미지 파일에 대해 블록 기반 파일 결합 주입 기법을 수행하였다.

본 논문에서는 윈도우즈에서의 대표적인 벡터방식 이미지 파일 포맷인 WMF, EMF에 대해 블록 기반 파일 결합 주입을 수행하였다. WMF[22]는 Windows MetaFile의 줄임으로써 Microsoft사에서 제안하여 사용하고 있는 파일 포맷이다. WMF는 16비트에서 구현된 관계로 완전한 32비트 이미지를 지원하기 위해 EMF가 지원되었다. EMF는 WMF의 확장이며, 많은 부분이 WMF와 유사하다. 큰 차이점은 WMF의 기본 데이터 타입은 16비트이며, EMF는 32비트라는 점이다.

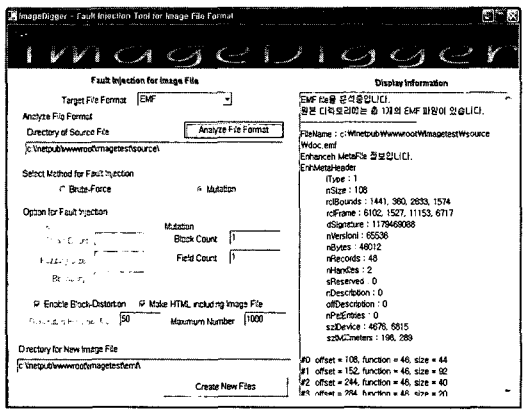


〔그림 4〕 WMF 파일 포맷 분석예

〔그림 4〕는 WMF 파일 포맷을 분석한 것으로 WMF 파일 생성 코드와 생성 파일의 데이터를 나타낸다. WMF 파일은 이미지 관련 함수당 하나의 레코드가 할당이 된다. 예를 들어 Rectangle(hdcMeta, 0, 0, 100, 100) 함수의 경우 파일에서 12h 위치의 07 00 00 00 1B 04 64 00 64 00 00 00 00 00 으로 저장된다. 07 00 00 00 은 레코드의 길이를 나타내는 필드로 지금 레코드의 크기가 7*2byte = 14byte임을 나타낸다. 1B 04 는 Rectangle 함수를 나타내는 함수 번호이며 이 후의 데이터는 해당 함수로 전달되는 파라미터인 0, 0, 100, 100을 나타낸다.

4.2. ImageDigger 설계 및 구현

ImageDigger는 그래픽 파일 포맷에 대해 블록 기반 퍼징을 수행할 수 있는 프로그램이다. WMF, EMF에 대해 블록기반 결함 주입을 수행하며 C#으로 구현되었다. 테스팅 대상 소프트웨어는 웹브라우저로서 Internet Explorer, Fire Fox Opera와 같은 모든 웹브라우저에서 수행 가능하다. ImageDigger는 Brute-Force, Block-Consent, Block-Distortion의 세가지 모드로 동작을 하며, Block-Consent와 Block-Distortion을 수행하기 위해 검증 대상이 되는 파일 포맷에 대해 분석을 해야 한다. 결함이 주입된 파일을 저장할 때 해당 이미지 파일을 파싱할 수 있는 HTML 문서를 같이 생성한다. 웹브라우저가 대상이 되기 때문에 HTML 문서를 생성하는 것이며, 생성된 파일이 자동으로 수행되게 하기 위해



〔그림 5〕 ImageDigger Interface

META 태그의 reflesh 속성을 사용한다.

〔그림 5〕는 ImageDigger의 인터페이스(interface)를 나타낸다. ImageDigger는 Brute-Force와 Mutation의 총 2개의 결함 주입 방법을 선택할 수 있으며, Mutation에서 본 논문에서 제안한 Block-Consent와 Block-Distortion이 구현되어 있다. Block-Distortion을 사용하기 위해선 Mutation과 Enable Block-Distortion 옵션을 선택하면 된다. Brute-Force 방법으로 결함 주입을 선택하면 3개의 선택 사항이 있다. Offset Count는 결함 주입 데이터의 총 수의 최대값을 나타내며, Fuzzing Size는 결함 주입 데이터의 길이의 최대 값을, Boundary는 결함을 주입할 총 파일 데이터의 범위를 나타낸다. 최대값을 설정함으로써 해당 범위 내에서 값을 임의적으로 선택하도록 하였다. Mutation 방법으로 결함 주입을 수행하면 2개의 선택사항이 있다. Block Count는 ImageDigger에서 분석한 WMF 파일의 레코드에서 결함을 주입할 레코드 수의 최대값을 나타내며, Field Count는 결함을 주입할 레코드 필드 수의 최대값을 나타낸다. 이렇게 설정한 값을 이용하여 ImageDigger는 Directory of Source File에서 설정한 디렉토리에 있는 대상 파일에 대해 Generated Files per file의 값만큼 결함을 주입한다. 결함을 주입한 파일과 해당 파일을 로드할 HTML 문서를 생성한 후 HTML 문서를 실행시키면 자동으로 결함을 주입한 파일을 로드 시킨다. 만약 취약점이 발생하면 디버거가 이상현상을 발견하고, 분석은 디버거를 이용하여 수행한다.

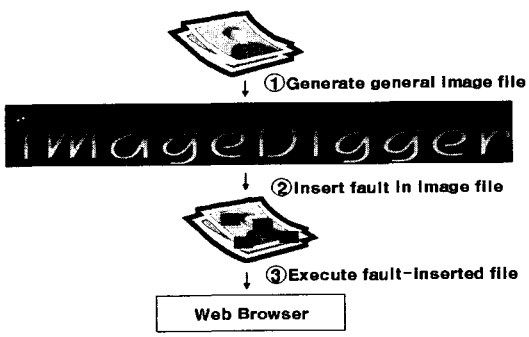
4.3. ImageDigger를 이용한 검증 실험

웹브라우저의 취약성을 검증하기 위해 ImageDigger를 이용하여 WMF, EMF 파일 포맷에 대해 결함 주입을 수행하였으며, 실험환경은 다음과 같다.

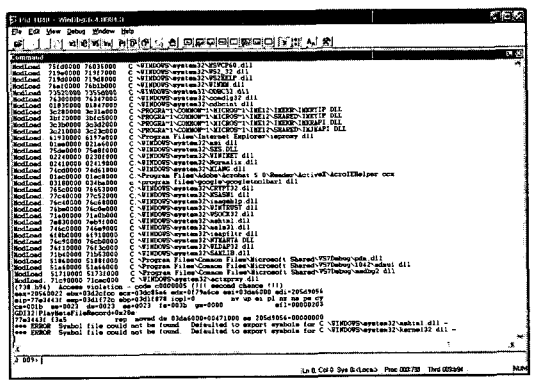
- 소프트웨어 보안 테스팅 대상 웹브라우저 : Internet Explorer 6.0
- 대상 설치 운영체제 : Windows XP SP2
- 대상 하드웨어 : Intel Pentium 4 CPU 3GHz, 2GB RAM

〔그림 6〕은 ImageDigger를 이용한 결함 주입 메커니즘을 나타낸다. 우선 WMF와 EMF 파일 포맷으로 이미지 파일을 생성한다. 생성한 파일에 대해 ImageDigger를 이용하여 결함을 주입한다. 생성된 파일은 결함이 주입된 이미지 파일과 해당 파일을 웹브라우저로 로딩 할 HTML 문서 파일이다. 결함을 주입한 파일을 Internet Explorer에서 자동으로 로딩하면서 소프트웨어의 결점을 탐지한다. 결함을 주입하는 방법으로 ImageDigger는 Brute-Force, Block-Consent, Block-Distortion의 세가지 방법 중 하나를 선택한다.

ImageDigger에서 생성하여 검증을 수행한 파일은 Brute-force와 Block-consent & Block-distortion 방법으로 각각 10만개이다. 이들 파일을 실행 시킨 후 이상현상을 실험한 결과 서비스 거부 취약점이 발견되었으며 이들 취약점의 원인을 분석한 결과, Brute-force 방법으로 7 종류, Block-consent & Block-distortion 방법으로 10 종류의 서비스 거부 취약점을 발견하였다. 서비스 거부 취약점이 발생하였을 경우 그림 7과 같은 화면을 볼 수 있으며, 이상 현상은 WinDBG[13]를 이용하여 분석을 하였다. 해당 10 종류의 취약점은 WMF 파일 포맷 중 레코드 길이 필드에 큰 값이 삽입되어 발생하였다. Brute-force의 경우 임의의 위치에 결함을 삽입



〔그림 6〕 ImageDigger를 이용한 실험



〔그림 7〕 서비스 거부 취약점이 발생한 IE

하기 때문에 레코드 길이 필드에 정확히 결함이 삽입되는 경우가 적어 7 종류의 서비스 거부 취약점이 나온 반면, Block-consent & Block-distortion 방법은 삽입되는 필드 중 레코드 길이 필드도 포함되므로 서비스 거부 취약점이 10 종류가 발견되었으며, Brute-force 보다 3 종류 더 발견됨을 알 수 있었다.

V. 결 론

본 논문에서는 블록 기반 파일에 대해 결함 주입을 하는 보안 테스팅 방법론을 제안하였다. 본 논문에서 제안한 방법론은 파일 내의 여러 필드들을 묶어 블록으로 처리하는 파일 포맷을 대상으로 필드를 고려하여 결함을 주입하여 테스팅을 수행함으로써 소프트웨어의 취약점을 발견한다. 파일은 하나 이상의 블록으로 나누어 질 수 있으며 이들 블록들의 형식에 어긋나지 않게 결함을 주입함으로써 결함이 주입된 파일이 에러로 처리되는 것을 줄일 수 있도록 하였다. 본 논문에서는 제안한 블록 기반 파일 결함 주입 방법론을 여러 블록 기반 파일 포맷 중 하나인 이미지 파일에 적용하였다. 이미지 파일은 인터넷에서 자동으로 다운을 받아 메모리에 로드되는 파일이므로 해당 파일로 인해 소프트웨어가 큰 영향을 받을 수 있다. 이미지 파일 중 많은 수의 블록으로 나누어 질 수 있는 WMF, EMF 파일 포맷에 대해 결함 주입을 수행하였으며, 이들 파일 포맷에 대해 자동으로 결함을 주입할 수 있도록 ImageDigger를 구현하였다. ImageDigger를 이용하여 이미지 파일에 대해 결함 주입을 수행하였으며, 실험 결과 총 10 종류의 서비스 거부 취약점을 발견할 수 있었다. 본 논문에서 제안한 블록 기반 결함 주입 방법론 및 ImageDigger는 소프트웨어의 취약점 중 메모리 관련 취약점을 찾는 영역에 효과 있다. 이와 함께 ImageDigger는 이미지 파일 이외의 PE 파일 포맷과 MS Office 파일 포맷 같은 블록 기반 파일에 대해서도 적용 가능하다.

참고문헌

[1] James A. Whittaker, "Software's Invisible Users", IEEE Software, 18(31):pp84-88, 2002

[2] Perter Oehlert, "Violating Assumptions with Fuzzing", IEEE Security & Privacy Magazine, 3(2):58-62, 2005

[3] Microsoft Security Bulletin, http://www.microsoft.com/technet/security/current.aspx

[4] James A. Whittaker, Herbert H. Thompson, How to Break Software Security, chapter 3, Addison Wesley

[5] Microsoft Windows GDI WMF Remote Denial of Service Vulnerability, http://www.securityfocus.com/bid/23275

[6] Microsoft Windows OLE32.DLL Word Document Handling Denial of Service Vulnerability, http://www.securityfocus.com/ bid/22847

[7] Microsoft Excel NULL Pointer Dereference Denial of Service Vulnerability, http://www.securityfocus.com/bid/22717

[8] Jeffrey M. Voas, Gary McGraw, Software Fault Injection Inoculating Programs Against Errors, Wiley

[9] Ballistar Project, http://www.ece.cmu.edu/~koopman/Ballistar/

[10] Nathan P. Kropp, Philp J. Koopman, Daniel P. Siewiorek, "Automated Robustness Testing of Off-the-Shelf Software Components", Proceeding of the 28th Fault Tolerant Computing Symposium, pp230-239, 1998

[11] Michael Sutton, Adam Greene, "The Art of File Format Fuzzing", Blackhat, 2005

[12] SPIKEfile, http://labs.idefense.com/software/fuzzing/php#more_spikefile

[13] Ilja van Sprundel, "Unix Kernel Auditing"

[14] Justin E. Forrester, Barton P. Miller, "An Empirical Study of the Robustness of Windows NT Applications using Random Testing", 4th USENIX Windows System Symposium, 2000

[15] James C. Foster, Vitaly Osipov, Nish Bhalla, Buffer Overflow Attacks, Syngress

[16] Jack Koziol, David Litchfield, Dave Aitel, Chris Anley, The Shellcoder's Handbook: Discovering and Exploiting Security Holes, Wiley

[17] Debugging Tools for Windows, http://www.microsoft.com/whdc/devtools/debugging/ de-

fault.mspx
[18] Wotsit's Format, http://www.wotsit.org
[19] Matt Pietrek, "An In-Depth Look into the Win32 Portable Executable File Format", http://msdn.microsoft.com/msdnmag/issues/02/02/PE/default.aspx
[20] Common Vulnerability and Exposures, http://www.cve.mitre.org
[21] Charles Petzold, Programming Windows 5th Edition, Microsoft Press
[22] Windows MetaFile, http://en.wikipedia.org/wiki/Windows_Metafile

〈著者紹介〉

최 영 한 (Young-Han Choi)
2002년 2월: 한양대학교 전자전기학과 졸업
2004년 2월: 한국과학기술원 전자전산학과 석사
2004년 2월~현재: 국가보안기술연구소
<관심분야> 운영체제, 정보보호, 소프트웨어 테스팅

김 형 천 (Hyoung-Chun Kim)
1999년 8월: 고려대학교 전산학과 졸업
2001년 8월: 고려대학교 전산학과 석사
2001년 3월~현재: 국가보안기술연구소 선임연구원
<관심분야> 시스템 보안, 소프트웨어 테스팅, 데이터마이닝

홍 순 좌 (Soonjwa Hong)
1989년 2월: 숭실대학교 전자계산학과 졸업
1991년 2월: 숭실대학교 전자계산학과 석사
1991년 3월~2000년 국방과학연구소 선임연구원
2000년~현재: 국가보안기술연구소 선임연구원
<관심분야> 컴퓨터 보안, 유/무선 통신 보안