정 보 보 호 학 회 지
제28권 제2호, 2018. 4

기업 제품의 보안 취약점 개선을 위한 보상제 동향

유 동 훈*, 노 봉 남**

요 약

최근 새로운 플랫폼과 인터넷 환경의 급속한 변화로 인해 소프트웨어가 다양화되면서 보안 위협도 나날이 증가하고 있어 보안 취약점에 대한 효과적인 대응 방안 마련이 주요 쟁점으로 대두되고 있다. 해외 주요 소프트웨어 벤더사의 경우 보안 취약점에 체계적으로 대응하기 위해 설계 단계에서 보안을 적용하는 사례가 알려져 있지만 해외와 국내 소프트웨어 개발 생태계의 환경적 차이점으로 말미암아 동일한 방식을 일률적으로 적용하는 것은 많은 어려움이 따른다. 이번 논문에서는 기업 제품의 보안 취약점을 개선하기 위한 국내외 사례를 통해 소프트웨어 개발 시 설계에서 배포단계까지 각 단계별 취약점 발굴 활동의 유용성과 배포 후 취약점 발굴 인력의 집단지성을 활용하는 보안 취약점 보상제도의 동향에 대해 설명하고자 한다.

Ⅰ. 서 론

최근 새롭게 등장한 스마트 플랫폼과 사물 인터넷의 발전으로 인터넷 접속 환경이 급속히 변화되었다. 이로 인해 과거에는 PC에 국한되었던 소프트웨어의 이용 범위가 생활 전반으로 확대되었고 소프트웨어 보안 위협도 나날이 증가하고 있는 추세이다. 소프트웨어 이용 범위의 확대는 다양한 소프트웨어 등장으로 이어졌고 그 결과 소프트웨어 취약점으로 인한 침해사고 유발 경로가 증가할 수밖에 없는 상황이다.

올해 CVEdetails에서 발표한 보안 취약점 통계[1]에 따르면 2017년에 한 해 동안 14,515건의 신규 취약점이 발견되었으며 1999년 첫 조사 이후 사상 처음으로 1만 건을 돌파하였다. 지난 6년간 발견된 보안 취약점의 개수를 보면 2011년에 잠시 감소하였다가 다시 해마다 증가하고 있다[2].

이처럼 꾸준히 늘어나는 보안 취약점을 최소화하기 위한 방안으로 소프트웨어 벤더 사에서 적용 중인 설계 단계에서 보안을 적용하는 방법이 널리 알려져 있다. 실제 미국 마이크로소프트사의 사례를 보면 개발 단계에서 SDL(Security Development Lifecycle)[3] 적용 시 보안 취약점이 단기간에 45% 감소하고 3년 동안 91%의 취약점이 감소되었다고 보고된 바 있다. 뿐만 아니라 소프트웨어 개발 단계에서 취약점을 제거할 경우 유지보수 비용을 최대 1/60로 절감시킬 수 있다는 연구 결과가 발표된 바 있다[4].

하지만 일부 소프트웨어 벤더 회사를 제외한 대부분의 소프트웨어 개발 기업이 여전히 소프트웨어 개발 단계에서 부족한 예산, 촉박한 개발 기간, 보안 인식 부재 등의 문제로 인해 소프트웨어 보안 취약점 대응을 등한시 하고 있는 상황이다. 소프트웨어 개발 단계부터 보안을 적용하여 취약점을 제거하는 것이 시간과 비용 감소 측면에서 여러모로 이점을 가지고 있음에도 불구하고 국내 소프트웨어 개발 생태계 특성으로 인해 당장 현실화되기 어려운 한계점이 존재하고 있다.

또한 이렇게 개발 단계에서 보안을 적용한다고 하더라도 모든 소프트웨어 취약점이 제거되는 것은 아니기 때문에 추가적으로 소프트웨어 배포 이후 단계에서 안전성 검증 과정을 필요로 한다. 그러나 현실에서는 누구나 자유롭게 배포 가능한 소프트웨어에 대한 안전성 검증 절차가 존재하지 않고 일반 소프트웨어 제품에 대한 보안성 검증 규제가 전무하여 소프트웨어 배포 단계에서의 안전성 검증이 미흡하다는 문제점을 가지고 있다. 추가적으로 소프트웨어 도입 이후에도 일부 기능 업데이트나 유지보수 개발로 인해 신규 코드가 추가되면서 안전했던 소프트웨어에 다시 취약점이 유발 될 수 있는

* ㈜아이넷캅 스마트 플랫폼 보안 기술 연구소 (x82@inetcop.org)
** 전남대학교 대학원 시스템 보안 연구 센터

근본적인 문제점도 존재하고 있다.

이에 본 논문에서는 기업 제품의 보안 취약점을 개선하기 위한 국내외 사례를 통해 소프트웨어 개발 시 설계에서 배포단계까지 각 단계별 취약점 발굴 활동의 유용성과 배포 후 취약점 발굴 인력의 집단지성을 활용하는 보안 취약점 보상제도의 동향에 대해 설명하고자 한다.

Ⅱ. 보안 취약점 배경 및 동향

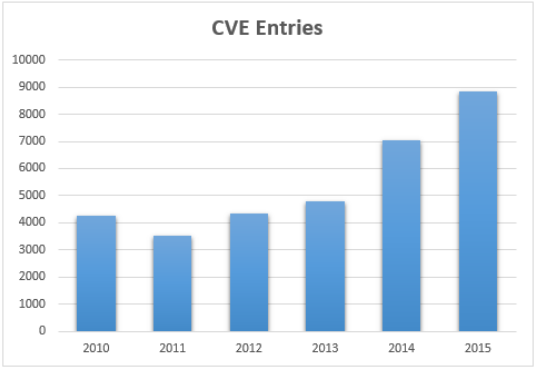
2.1. 보안 취약점 개요

보안 취약점은 소프트웨어에 존재하는 프로그래밍 버그나 설계 결함으로 인해 공격자에게 허용된 권한 이상의 동작이나 허용된 범위 이상의 정보 열람을 가능하게 하는 약점 또는 결함이다. 공격자는 소프트웨어에 존재하는 보안 취약점을 악용하여 시스템 내의 제어 권한을 획득할 수 있고 이를 통해 특정한 정보를 탈취할 수도 있다. 보안 취약점의 종류는 네트워크 여건과 PC 환경의 발전, 시간의 흐름에 따라 변해왔는데 과거 주공격 대상이던 유닉스 운영체제에 존재하는 서버 사이드 취약점부터 웹 애플리케이션을 대상으로 한 웹 서비스 취약점을 거쳐 개인 PC 환경이 확립되면서 서버가 아닌 클라이언트 사이드 취약점까지 다양화 되었다. 클라이언트 사이드 취약점 유형은 사용자의 반응을 필요로 하는 웹 브라우저나 PDF, MS Office 등 문서 뷰어에 존재하는 취약점 유형을 의미한다. 최근 스마트 플랫폼과 사물 인터넷 시대가 도래 하면서 다시 과거와 유사한 고전 유닉스 서버 사이드 취약점 및 웹 어플리케이션 취약점 사례가 발생하고 있으며 서버 사이드, 클라이언트 사이드 취약점의 구별이 모호한 취약점 유형들이 늘고 있는 추세이다.

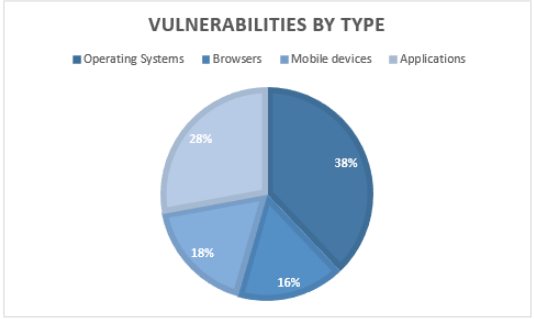
2.2. 보안 취약점 동향

최근 CVEdetail 통계에 따르면 2017년 한 해 동안 발견된 CVE 개수는 총 14,515건으로 1999년 첫 조사 이후 사상 처음으로 1만 건을 돌파했다. 이는 2016년 6,447건 대비 두 배 이상 증가한 수치이다. 우선, 기업별로는 구글(1000건), 오라클(871건), 마이크로소프트(683건), IBM(652건), 애플(557건), 시스코(490)건 순으로 서비스와 제품군이 다양하고 사용자의 이용률이 높은 기업일수록 많은 보안 취약점이 발견되고 있었다. 제품 서비스별로는 안드로이드가 1위(841건)로 가장 높았고 리눅스 커널이 2위(434건), iOS가 3위(365건), 맥 OSX가 5위(278건), 윈도 10이 6위(265건), 윈도 7이 10위(227건) 순으로 집계 되었다. 마지막으로 취약점에 대한 누적 통계는 미국 국가표준기술연구소(NIST), NVD(National Vulnerability Database) 분석 결과를 바탕으로 GFI 발표한 6년간의 통계 자료로 알 수 있다[5].

그림 1의 연도별로 발견된 보안 취약점 개수를 살펴보면 2011년에 잠시 감소하였다가 다시 해마다 증가하고 있는 추세임을 알 수 있으며 그림 2와 같이 주로 발생했던 취약점의 종류는 운영체제가 1위(38%), 일반 애플리케이션이 2위(28%), 모바일 디바이스가 3위(18%), 웹 브라우저가 4위(16%)를 차지하였다.



〔그림 1〕 지난 6년간 발견된 보안 취약점 누적 통계



〔그림 2〕 지난 6년간 발견된 보안 취약점 종류

Ⅲ. 보안 취약점 개선을 위한 대응 사례

3.1. 개발 설계 단계에서의 취약점 대응

마이크로소프트사는 소프트웨어 취약점을 줄이기 위해 2002년 1월부터 SDL(Secure Develop Lifecycle)을 통한 소프트웨어 개발 프로세스를 개선하여 운영 중이다. SDL은 소프트웨어 개발 이전 단계부터 시작해서 설계 및 구현을 거쳐 배포 및 운영 단계, 시험 및 검증 단계까지 소프트웨어에서 발생할 수 있는 취약점을 최소화하기 위한 목적으로 만들어졌다. 마이크로소프트사에서 운영 중인 SDL의 기본 원리는 Secure by Design, Secure by Default, Secure in Deployment, Communication (SD3+C) 이다.

SD3+C를 기반으로 소프트웨어 설계 단계에서 위협 모델 개발, 구현 단계에서 코드 스캐닝을 위한 정적 분석 도구 사용, 검증 단계에서 코드 검증 및 보안성 테스트 작업이 업무에 기본으로 포함되었으며 개발된 소프트웨어는 출시하기 전에 개발팀과는 별도의 팀에서 최초 보안 검증을 거치도록 하였다.

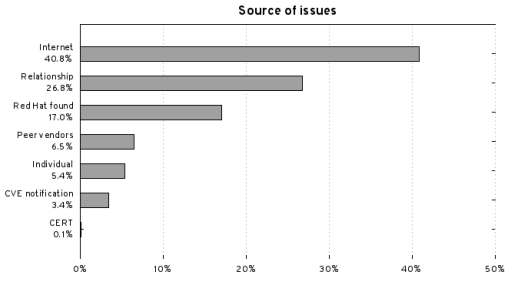
마이크로소프트사의 통계 자료에 따르면 윈도우즈 OS 발표 이후 1년간 공개된 취약점의 수를 비교하였을 때 SDL 적용 이전에 119개(윈도우즈 XP 기준)였던 취약점이 SDL 적용 이후 66개(윈도우즈 vista 기준)로 45% 정도 줄어든 것을 확인할 수 있었다. 장기적인 효과는 더욱 극명하게 나타나는데 제품 발표 이후 3년간 공개된 취약점의 수를 비교한 결과 SDL 적용 이전의 SQL 서버 2000에서 발견된 취약점은 34개였던 것에 비해 SDL 적용 이후 SQL 서버 2005에서 발견된 취약점은 단 3개로 91%의 취약점이 감소한 것으로 확인되었다. 마이크로소프트사의 SDL은 교육 단계, 요구사항 단계, 설계 단계, 구현 단계, 검증 단계, 배포 단계, 대응 단계로 구성되는데 각 단계별 세부 내용은 표 1과 같다.

3.2. 배포 이후 단계에서의 취약점 대응

오픈 소스 소프트웨어에 대한 취약점 발견은 내부의 자체 점검에 의한 것보다는 외부 사용자들에 의한 발견이 더 많은 비중을 차지하고 있는데 대표적으로 과거 5년간 가장 많은 소프트웨어 취약점이 발견되었던 레드햇사의 사례를 들 수 있다[6]. 레드햇의 경우 그림 3과 같이 사용자들이 리포트 하는 정보를 통해 취약점이 발

〔표 1〕 마이크로소프트 SDL 단계별 설명

단계	설명
교육	S/W 개발팀 구성원 대상으로 보안 기초, 최신 보안 동향 정보 교육 매년 1회 교육 실시. 보안 설계, 위협모델링, 보안 코딩, 보안 시험, 프라이버시 등 내용 포함
요구사항	신뢰성 있는 소프트웨어를 구축하기 위한 기본 보안 요구사항, 프라이버시 요구사항 정의 필수 항목: SDL 방법론 적용 여부 결정, 보안 책임자 선정, 보안팀 선정, 버그 리포팅 도구 정의, 보안 버그 경계 정의, 보안 위험 평가 등, 권고 항목: 보안 계획서 작성, 버그 추적 시스템 정의
설계	SDL 구현에서부터 배포에 이르는 동안 수행해야 하는 작업 계획 수립 단계 필수 항목: 보안 설계 검토, 방화벽 정책 준수, 위협 모델링, 위협 모델 품질 보증, 위협 모델 검토 및 승인 등, 권고 항목: 보안 설계서 문서 작성, 보안 디폴트 인스톨 실행, 모든 샘플 소스 코드 보안 검토 수행, 안전하지 않은 함수와 코딩 패턴 알림, 설계 변화 요구에 관한 보안 관련 사항 문서화, 위협 모델을 통해 찾아진 취약점을 위한 작업 목록 작성
구현	구현 단계는 보안 및 프라이버시 문제점을 발견하고 제거하기 위해 개발 모범 사례 수립 후 이행 필수 항목: 최신 버전의 빌드 도구 사용, 금지된 API 사용 회피, Execute 허가를 통한 SQL 안전하게 사용, 저장된 프로시저에서 SQL 사용 등, 권고 항목: 안전하게 소프트웨어 사용을 위한 필요한 사용자 정보 식별, 사용자 중심의 보안 문서 계획, 보안 형상관리에 관한 정보 생성, 자동화 된 금지 API 변환 실행, 프로젝트 팀 전체와 모든 모범사례와 정책에 대해 정의, 문서화, 토론 등
검증	검증 단계는 코드가 이전 단계에서 설정한 보안과 프라이버시를 지키는지 보안 및 프라이버시 시험과 보안 푸쉬(security push), 문서 리뷰를 통해 확인 (보안 푸쉬는 팀 전체에 걸쳐 위협 모델 갱신, 코드 리뷰, 테스팅에 초점을 맞춘 작업) 필수 항목: 커널-모드 드라이버를 위한 시험 완료, COM 객체 시험 수행, 인증된 사이트 크로스 도메인 스크립팅을 위한 시험, 애플리케이션 검증 시험 수행, 파일 퍼징 수행, 위협 모델 검토 및 수정 등, 권고 항목: 보안 시험 계획 완료, 침투 시험 수행, 보안 코드 검토, 보안 푸쉬를 시작하기 전 모든 코드에 대한 우선 순위 결정, 보안 문서 계획서 검토 등



〔그림 3〕 레드햇의 보안 취약점 발굴 출처

견되는 경우가 전체의 무려 73%를 차지하였으며 직원들에 의해 발견되는 경우가 17% 정도의 비율을 차지했고 그 다음으로 CERT(Computer Emergency Response Team)나 CVE(Common Vulnerabilities and Exposures)와 같이 공개 기관을 통해 알려지는 경우가 10% 정도의 비율을 차지하였다[7].

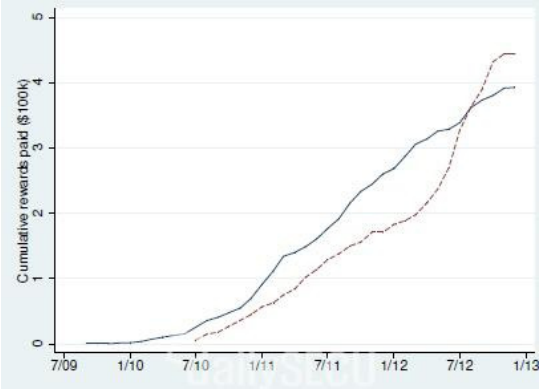
실제로 2014년에 발견되었던 중대 결함 2개의 경우 레드햇사의 직원이 아닌 외부에 의해 발견되었는데 가장 악명 높았던 OpenSSL의 하트블리드(HeartBleed) 메모리 노출 결함[8]의 경우 구글 보안팀에 의해 발견되어 알려졌으며 OpenSSL CCS 인젝션 결함[9]의 경우 사용자의 신고를 통해 알려진 사례이다. 이렇게 오픈 소스 소프트웨어의 경우 집단지성을 통해 취약점을 보다 많이 발견하여 소프트웨어의 안전성을 높일 수 있는 장점을 가진 반면 발견된 취약점을 개선하기 쉽지 않다는 단점도 가지고 있다.

미국 보안 업체인 Trustwave사는 2012년에 발견된 제로데이 취약점이 패치 되는데 소요된 시간을 분석하여 발표[10]한 바 있는데 그 결과에 따르면 리눅스 커널 취약점 해결에 평균 2년 이상 소요되며 이는 윈도우즈의 2배 이상이라고 밝혔다. 정확히 말하면 윈도우즈 운영체제의 제로데이 취약점의 경우 패치에 375일이 소요된 반면, 리눅스 커널 제로데이 취약점 해결은 평균 857일이나 소요된 것으로 조사되었다. 이와 같은 결과가 나타난 것은 여러 원인이 있지만 소프트웨어를 책임지고 배포하는 벤더의 입장에서는 소프트웨어 문제점에 대한 전적인 책임과 패치 제작이 표준화 되어 소요 기간이 적은 반면, 오픈 소스의 경우에는 다양한 사용자들이 해당 취약점을 해결하기 위해 참여해야 하는 구조적 차이점이 가장 큰 것으로 판단되고 있다.

Ⅳ. 취약점 발견자 보상제도 운영 사례

버그 바운티(취약점 발견자 보상제도)는 기업이 제공중인 서비스나 소프트웨어에서 취약점을 발견한 사람에게 신고를 받고 그에 합당한 보상금을 지급하는 제도이다. 버그 바운티는 과거 1995년 넷스케이프의 웹 브라우저 취약점 신고 제도부터 시작되었는데 2010년 이후부터 다양한 기업들이 자사 소프트웨어에서 발굴되는 결함에 대해 보상금을 지급하고 있으며 해외 벤더(페이스북, 구글, 마이크로소프트, 링크드인, 어도비, 트위터, 핀터레스트, 라이엇게임즈 등), 글로벌 IT 업체들이 잠재적으로 고객에게 피해가 발생하는 것을 방지하기 위한 목적으로 자신들이 운영하고 있는 서비스를 공격하여 취약점을 발견한 사람들에게 상금으로 포상해주고 있다. 반면 국내의 경우 대부분이 버그 바운티 프로그램을 운영하지 않고 있는데 현재까지 제도에 대해 알지 못하는 기업이 대다수일 것으로 예상되고 있으며 사용자가 취약점을 발굴하여 신고하더라도 기업의 허가 없이 소프트웨어를 개작하거나 분석하는 행위 자체가 소프트웨어 이용 약관에 위배되기 때문에 제도의 정착을 어렵게 만들고 있다.

실제 캘리포니아 대학에서 진행한 '취약점 보상금 제도에 대한 실증적 연구' 보고서[11]에 따르면 버그 바운티를 도입하는 것이 정규 보안 전문직을 채용하는 것보다 최대 100배의 비용 절감 효과가 있음을 발표한 바 있다. 해당 연구는 지난 3년간 구글과 모질라 재단에서 운영 중인 버그 바운티를 분석하여 약 40만 달러를 버그 바운티로 지출한 것이 동일 수의 취약점을 발견하기 위해 직원을 고용한 것보다 훨씬 적은 비용이 소요되는



〔그림 4〕 구글(실선)과 모질라(점선)의 보상금 추이

것으로 조사되었다. 이러한 통계로 비추어 보았을 때 소프트웨어를 제조하는 국내 기업들의 자발적인 버그 바운티 제도를 운영하는 것이 보안을 위해 지출되는 비용을 절감하고 취약점을 대량으로 발굴하여 제품의 보안성을 강화시키는데 더 효율적이라고 판단할 수 있다.

4.1. 해외 취약점 발견자 보상제도 기업 운영 사례

4.1.1. 구글[12]

2010년 11월부터 구글에서 운영하는 웹 서비스의 취약점을 발견한 사람에게 보상금을 지급하고 있으며 2011년 한 해 동안 총 7억 원을 지급하면서 활성화되었다. 취약점을 신고한 사람에게는 보상금 지급 외로 그 공로를 인정하여 명예의 전당(Hall of Frame)에 등록된다. 웹 서비스 취약점이다 보니 XSS, CSRF, SQL 인젝션 등의 취약점이 가장 많이 신고 되고 있으며 그 밖에 인증 우회 및 정보 유출, 원격 코드 실행 등의 결함을 발견하였을 때도 포상해준다. 최근 버그 바운티 범위를 모바일 앱까지 확장시켰으며 양질의 취약점을 발견할 수 있도록 벤더 최초로 연구 시작 이전에 조건 없이 사전 투자를 지급하는 선행 보상 제도를 운영하기로 하였다. 2016년 보상금 규모가 35억여 원까지 5배 증가하여 2017년 말까지 총 100억 원 이상을 버그 바운티 보상금으로 지급한 것으로 알려졌다.

4.1.2. 페이스북[13]

2011년 8월부터 페이스북 웹 서비스 취약점을 발견한 사람에게 신고 포상제를 운영하고 있다. 서비스 특성상 계정 인증을 필수적으로 요구하는 탓에 실제 계정이 아닌 가급적 취약점 발견을 위한 시험 계정을 사용하도록 권고하고 있다. 보상해주는 취약점으로는 인증 우회, XSS, CSRF, 프라이버시 권한 모델 회피, 권한 상승 등이 있으며 사용자 데이터를 훼손할 수 있거나 내부 인프라에 접근 가능한 취약점을 발견하였을 경우 보상금을 지급하고 감사 페이지 공로 목록에 이름을 등록해준다. 2017년 말까지 총 57억 원을 제보자들에게 지급하였다고 한다.

4.1.3. 마이크로소프트[14]

2013년 6월부터 버그 바운티 제도를 운영하기 시작하였으며 IE11에 대한 취약점을 신고하면 보상금을 지급하고 있다. 타사의 버그 바운티 제도와 비교하여 특이한 점은 운영체제 최신 버전에 적용된 공격 대응 기술을 무력화할 수 있거나 취약점 대응을 위한 효과적인 아이디어를 제시하는 사람에게 대규모의 보상금을 지불하고 있다는 점이다. 두 아이디어를 제공할 경우 IE11 취약점 보상금의 최소 5배에서 10배 정도 규모의 보상이 이루어지고 있다.

4.1.4. 링크드인[15]

링크드인의 경우 타사와 달리 비공개 버그 바운티 체제로 운영되고 있다. 링크드인의 정보보안책임자 설명에 따르면 업계에서 공개적으로 진행되고 있는 버그 바운티 중에 70% 이상이 잡음이 나오는 것으로 조사되었으며 전세계 보안 능력자들이 버그 바운티를 위해 취약점을 찾기 시작하면 파생되는 리스크가 더 클 것으로 보인다고 언급하였다. 2015년 6월 17일 링크드인이 버그 바운티를 운영하는 목적은 리스크 최소화이며 이를 위해 기존에 진행해오던 비공개 버그 바운티 체제를 유지하겠다는 계획을 공식 발표하였다.

4.2. 국내 취약점 발견자 보상제도 기업 운영 사례

4.2.1. 삼성전자[16]

삼성전자의 경우 2012년부터 삼성 스마트TV 및 블루레이 제품에서 취약점을 발견한 경우에 한하여 보상금을 지급하고 있으며 2017년부터 스마트폰 제품을 대상으로 취약점을 발견한 경우에도 상금을 지급하고 있다. 보상금 지급 대상은 출시 후 2년이 지나지 않은 제품으로 타 3rd 파티 모듈이 아닌 제품 내에서 발생하는 취약점에 한해서만 보상하고 있으며 취약점 신고 페이지에 취약점 검증을 위한 정보를 제출한 후 검증이 완료되고 나면 보상금 지급과 함께 명예의 전당에 공로 내역을 등록해준다.

4.2.2. 한국인터넷진흥원[17]

한국인터넷진흥원은 국내 소프트웨어 취약점이 악용되어 발생하는 피해를 예방하고자 2013년부터 신규 취약점 신고 포상제를 운영하고 있으며 신고 되는 취약점을 분기별로 평가하여 합당한 보상금을 지급하고 있다. 시행 목적 때문에 타 벤더에서 시행하는 버그 바운티와 달리 국내외에서 사용 중인 소프트웨어에 대한 취약점을 신고 받고 있으며 웹 게시판부터 공유기, 오피스 소프트웨어, 동영상 플레이어 등 다양한 범위의 소프트웨어 취약점에 대해 평가하고 있다.

4.2.3 라인[18]

NHN 라인은 2015년부터 시범 운영, 2016년 이후 상시 운영 개시로 버그 바운티 프로그램을 운영하고 있으며 라인 앱과 웹 사이트 등에서 발견되는 취약점에 대해 보상하고 있다. 보고된 취약점은 두 번의 내부 심사를 거쳐 결과에 따른 포상금이 지급되며 구글과 유사한 Hall of fame 페이지를 운영하고 있다.

4.3. 취약점 발견자 보상제도의 맹점

과거 사례로 비추어보면 버그 바운티 제도는 긍정적인 면과 동시에 부정적인 면 또한 갖추고 있다고 볼 수 있다. 그 이유는 해외 링크드인의 정보보안 책임자가 언급한 바와 같이 취약점을 분석하는 사람이 증가하고 전 세계 보안 능력자들이 취약점을 찾기 시작하면 그에 따른 부작용도 함께 나타날 수 있기 때문이다.

첫 번째 사례는 2011년 국내에서 최초로 제로데이 취약점을 웹 페이지 상에 개시해놓고 판매했던 'We Do Hack'의 사례이다[19]. 국내의 경우 실정법상 기업의 취약점 정보를 팔고 사는 행위가 법률을 위반하는 불법으로 해석될 수 있기 때문에 취약점 발굴에 따른 부작용으로 볼 수 있다. 이는 해외의 경우에도 마찬가지인데 보안 업체가 취약점을 구매하여 제조사에게 제보하는 중간자 역할이 수행하고 있음에도 불구하고 더욱 높은 가격을 받을 수 있는 블랙마켓을 통한 취약점 판매가 활발하게 이루어지고 있는 상황이다.

두 번째는 HP 사의 마이크로소프트 인터넷 익스플로러 취약점 발표 사례이다. 해당 사례는 HP가 취약점 발표 4개월 전에 마이크로소프트에 취약점을 신고하여 버그 바운티를 받았으나 120일이 지난 시점에도 마이크로소프트 측이 취약점을 패치하지 않았고 향후 취약점을 패치할 계획이 없다는 의사를 밝히면서 HP가 받았던 상금을 전액 기부하고 직접 취약점을 발표하게 된 것이다. 이렇게 버그 바운티 제도를 운영하고 있더라도 소프트웨어 제조사가 취약점에 적극적으로 대응하지 않는다면 제도 운영 자체가 무의미해지는 것을 단편적으로 보여준 사례라고 할 수 있다[20].

4.4. 해외 버그 바운티 플랫폼 전문 기업의 등장

2012년 이후부터 버그 바운티 플랫폼을 전문적으로 다루는 스타트업 기업들이 생겨나면서 전 세계적으로 버그 바운티 제도의 확대를 추진 중이다. 기존 버그 바

〔표 2〕 해외 버그 바운티 플랫폼 전문 기업

기업명	설명
Bugcrowd	최초의 공개 버그 바운티 플랫폼으로 37000명 이상의 연구원과 해커가 함께 일하고 있으며 최대 규모의 보안팀을 이루고 있는 것으로 알려지고 있음
Hackerone	기업을 위한 보안 편지함을 별도로 제공하는 버그 바운티 플랫폼으로 제출된 결과물에 대한 유효성 검사를 클라이언트가 담당하여 처리. 약 3700명의 연구원이 이용한 것으로 집계됨
Synack	전문 인력 고용을 위해 필기시험, 실기시험, 연구자의 배경을 조사하는 버그 바운티 플랫폼, 타 경쟁사보다 큰 보상금을 지불하며 연구원과 고객 모두 공개하지 않은 특징 존재
Cobalt	버그 바운티 플랫폼과 집단 지성을 이용한 침투 시험 서비스를 제공하는데 5000명의 연구원과 200명의 조사 관련 연구원, 329명이 유효한 보고서를 제출한 실적이 있음
ZeroCopter	연구원을 위한 초청 전용 플랫폼으로 운영되는 버그 바운티 플랫폼. 약 100명의 선택된 연구원이 일하고 있으며 모든 업무가 보고서로 처리되는 철저하게 관리되는 프로그램

운티 제도의 문제점은 기업이 업무 프로세스를 신뢰하지 못한다는 점과 취약점 발굴 인력에 대한 신뢰도가 떨어지는 문제 등을 가지고 있는데 버그 바운티 플랫폼은 이를 해결하기 위해 신뢰도가 떨어지는 것으로 판단되는 인력들을 걸러내는 기능을 제공함으로써 종래 버그 바운티 제도와 차별점을 제시하고 있는 것으로 알려져 있다. 버그 바운티 프로그램을 벤더에게 전문적으로 제공하는 서비스 플랫폼은 표 2와 같은데 대표적으로 Bugcrowd[23], Hackerone[24], Synack[25], Cobalt[26], ZeroCopter[27] 등이 있으며 매년 가파른 속도로 성장하고 있는데 2011년 실리콘밸리에서 창업한 Hackerone의 경우 2015년에 약 17,000여개의 취약점을 찾아 약 584만 달러에 달하는 매출을 달성하였다. 국내에는 수요 시장의 한계로 인해 이러한 버그 바운티 플랫폼을 전문으로 하는 기업 사례가 존재하지 않지만 버그 바운티를 운영하는 기업이 늘어날수록 향후에는 서비스를 제공하는 스타트업 기업들이 나타날 것으로 예상된다.

Ⅴ. 시사점 및 결론

본 논문에서는 보안 취약점 통계 동향을 통해 해마다 보안 취약점이 증가하고 있는 추세임을 소개하였고 이에 대응하기 위해 소프트웨어 설계 단계에서 적용할 수 있는 보안 대응 방안인 마이크로소프트의 SDL 사례에 대해 설명하였다. 그러나 국내 소프트웨어 생태계 환경의 특성상 소프트웨어 설계 단계에서 범용성 있게 기술을 도입하기 어려운 한계점을 가지고 있어 소프트웨어 배포 이후 단계에서 다수 보안 취약점 발굴 인력의 집단지성을 활용하는 보안 취약점 보상 제도에 대해 설명하였다. 해외 기업의 경우 그룹 내부에서 설계 단계의 보안 프로세스를 거치고 있음에도 추가적으로 취약점 발견자 보상 제도를 운영하고 있는데 이는 개발 설계 단계에서 발견되지 않은 심각한 보안 취약점이 소프트웨어 배포 이후 단계에서 발견될 수 있기 때문에 효과적임을 확인할 수 있었다.

지금까지 소프트웨어에 보안 취약점은 개발자 그룹이 해결해야 하는 과제로 비춰지고 있었으나 설계 단계에서 안전한 소프트웨어를 개발하도록 노력하는 것과 더불어 보안 취약점 발굴 인력의 집단지성을 통해 배포 이후 단계에서도 다양한 취약점을 제거할 수 있는 대응 방안 마련이 절실히 필요하다.

향후 취약점 보상 제도가 체계적이고 활발하게 운영되기 위해서는 제도가 가지고 있는 맹점에 대해 소프트웨어 개발 기업과 취약점 발굴 인력 모두가 우선적으로 인지하고 부작용 방지를 위한 노력이 뒷받침되어야 비로소 쌍방의 이익도 극대화 될 수 있을 것이다.

참 고 문 헌

[1] CVEdetails, https://www.cvedetails.com/

[2] GFI Blog, "2015's MVPs - The most vulnerable player"

[3] Microsoft, "Security Development Lifecycle", http://www.microsoft.com/en-us/sdl/

[4] Steve Lipner, Michael Howard, "The Trustworthy Computing Security Development Lifecycle", Microsoft Corporation, Mar 2005.

[5] NVD DB, https://nvd.nist.gov/

[6] Secunia, "Secunia Yearly Report 2011, Vulnerabilities Are Resilient", P.4~P.11, 2012.

[7] RedHat Security Blog, "The Source of Vulnerabilities, How Red Hat finds out about vulnerabilities", Oct 2014.

[8] Heartbleed bug, http://heartbleed.com/

[9] OpenSSL CCS Injection bug, http://ccsinjection.lepidum.co.jp/

[10] Trustwave, "Linux trailed Windows in patching zero-days in 2012, report says", 2012.

[11] Matthew Finifter, Devdatta Akhawe, and David Wagner, "An Empirical Study of Vulnerability Rewards Programs", 2013.

[12] Google, "Google Vulnerability Reward Program (VRP) Rules"

[13] Facebook, “Bug Bounty Program”

[14] Microsoft, "Microsoft Bounty Programs"

[15] LinkedIn's Security Blog, "LinkedIn’s Private Bug Bounty Program: Reducing Vulnerabilities by Leveraging Expert Crowds“

[16] Samsung "SMART TV BUGBOUNTY PROGRAM"

[17] KISA, "S/W 신규 보안 취약점 신고 포상제"

[18] Line, "LINE Security Bug Bounty Program"

[19] We Do Hack, http://wedohack.appspot.com/
[20] HP Security Research Blog, "There and back again: a journey through bounty award and disclosure"
[21] 김형열·김태성, "취약점 마켓 도입 영향요인에 대한 탐색적 연구: 화이트해커 중심으로", 「2016 한국경영정보학회 춘계학술대회」, 한국경영정보학회, 2016.
[22] 홍준호, 유현우, "화이트 해커 양성 및 활성화 방안에 대한 연구", 한국법학회, 법학연구 제17권 제4호(통권 68호), 2017.
[23] Bugcrowd, "Vulnerability Disclosure & Bug Bounty Programs"
[24] HackerOne, "Bug Bounty, Vulnerability Coordination"
[25] Synack, "Penetration Testing & Private Bug Bounty"
[26] Cobalt Labs, "Cobalt Bug Bounty Program"
[27] Zerocopter, "Vulnerability Disclosure Policy"

〈저자소개〉

유 동 훈 (Dong-hoon Yoo)
2010년 8월 : 전남대학교 정보보호학과 석사
2015년 2월 : 전남대학교 정보보호학과 박사
2001년 3월~현재 : ㈜아이넷캅 스마트 플랫폼 연구소
관심분야 : 정보보호, 모바일 및 사물인터넷 보안 등

노 봉 남 (Bong-nam Noh)
정회원
1978년 : 전남대학교 수학교육과 졸업 학사
1982년 : KAIST 대학원 전산학과 석사
1994년 : 전북대학교 대학원 전산과 박사
1983년~현재 : 전남대학교 전자컴퓨터정보통신공학부 교수
2000년~현재 : 시스템보안연구센터 소장
관심분야 : 정보보호, 시스템 및 네트워크 보안 등