다양한 취약점 점검 도구를 이용한 자동화된 네트워크 취약점 통합 분석 시스템 설계

(An Automatic Network Vulnerability Analysis System using Multiple Vulnerability Scanners)

윤 준 † 심 원 태 ††
(Jun Yoon) (Wontae Sim)

요 약 본 논문에서는 네트워크 취약점 분석 결과의 정확성을 향상시키기 위한 방법으로 다양한 취약점 점검 도구를 통합할 수 있는 네트워크 취약점 자동 분석 시스템을 제안한다. 일반적으로 전문가에 의한 수동 점검이 가장 정확한 취약점 점검 방법으로 평가되지만, 복잡하고 규모가 큰 네트워크의 경우 효율적인 취약점 분석을 위해 자동화된 네트워크 취약점 점검 도구를 활용한다. 그런데 취약점 점검 도구의 종류에 따라 점검 대상이 다르거나 동일한 점검 대상에 대해서도 점검 항목과 점검 결과가 다를 수가 있어, 상호보완적인 목적으로 몇 개의 취약점 점검 도구를 동시에 사용하는 것이 효과적이다. 그러나 취약점 점검 도구들의 점검 결과에 대한 연관성 분석과 통합 분석에는 사람에 의한 수동적인 분석 작업이 필요하기 때문에, 이것은 상당히 시간 소모적인 작업이 되고 네트워크의 규모에 따라 통합 분석이 불가능하기도 하다. 본 논문에서는 다양한 취약점 점검 도구를 통합할 수 있는 인터페이스를 제공하고, 공통의 점검 정책 수립과 통합 분석의 자동화를 특징으로 하는 네트워크 취약점 통합 분석 시스템을 제안한다.

· 본 연구는 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술 개발사업의 일환으로 수행하였음. [2005-S-606-02, 차세대 침해사고 예측 및 대응 기술]
· 이 논문은 2007 한국컴퓨터종합학술대회에서 '다양한 취약점 점검 도구를 이용한 자동화된 네트워크 취약점 통합 분석 시스템 설계'의 제목으로 발표된 논문을 확장한 것임

† 정 회 원 : 한국정보보호진흥원 인터넷침해사고대응지원센터 주임연구원
jun.yoon@gmail.com
†† 정 회 원 : 한국정보보호진흥원 인터넷침해사고대응지원센터 팀장
wtsim@kisa.or.kr

논문접수 : 2007년 9월 27일
심사완료 : 2007년 12월 8일

키워드 : 취약점 점검 도구, 취약점 통합 분석, 다중 취약점 점검 도구, 취약점 연관성 분석

Abstract This paper presents the design of network vulnerability analysis system which can integrate various vulnerability assessment tools to improve the preciseness of the vulnerability scan result. Manual checking method performed by a security expert is the most precise and safe way. But this is not appropriate for the large-scale network which has a lot of systems and network devices. Therefore automatic scanning tool is recommended for fast and convenient use. The scanning targets may be different according to the kind of vulnerability scanners, or otherwise even for the same scanning target, the scanning items and the scanning results may be different by each vulnerability scanner, Accordingly, there are the cases in which various scanners, instead of a single scanner, are simultaneously utilized with the purpose of complementing each other. However, in the case of simultaneously utilizing various scanners on the large-scale network, the integrative analysis and relevance analysis on vulnerability information by a security manager becomes time-consumable or impossible. The network vulnerability analysis system suggested in this paper provides interface which allows various vulnerability assessment tools to easily be integrated, common policy which can be applied for various tools at the same time, and automated integrative process.

Key words : integrative vulnerability analysis, multiple vulnerability scanners, vulnerability relevance analysis

1. 서 론

최근 온라인 쇼핑, 인터넷 뱅킹 등 개인의 경제적 활동 뿐만 아니라 국가 주요 기반시설의 정보통신에 대한 의존도가 높아짐에 따라, 해킹·바이러스로부터 정보통신망을 보호하고 안전하게 운영해야 될 필요성이 강조되고 있다.

정보통신망을 해킹·바이러스로부터 보호하기 위해서는 네트워크에 대한 취약점 분석을 통해 문제점을 파악하고 대비책을 세워 피해를 사전에 예방하는 것이 가장 좋은 방법이다.

네트워크 취약점 분석은 사람에 의한 수동적인 점검 방법과 자동화된 도구를 이용한 점검 방법으로 구분해 볼 수 있다. 이 중 잘 만들어진 체크리스트를 이용하여 관리자가 직접 분석을 하는 수동적인 방법이 가장 정확하고 안정적인 취약점 분석 방법이다. 하지만 이 방법은 상당히 시간 소모적이기 때문에, 서버가 몇 대만 있는 소규모 기업이나 안정성이 요구되는 주요 서버에만 적용할 수 있다는 한계가 있다. 그래서 복잡하고 규모가 큰 네트워크의

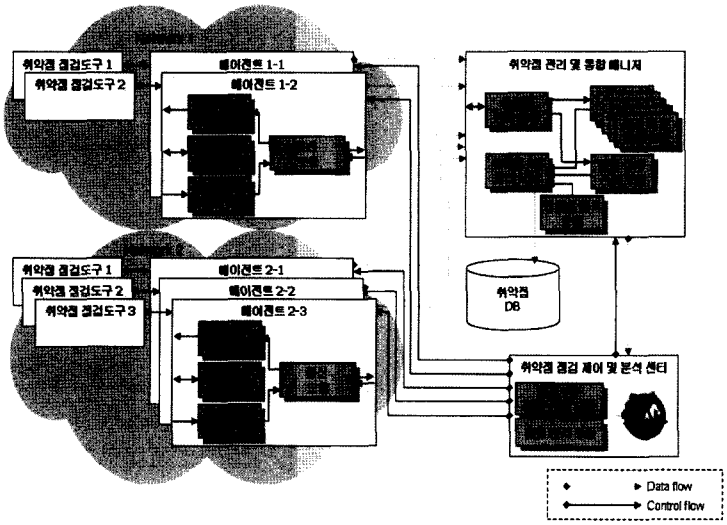

그림 1 MSCA 시스템 구성도

경우 자동화된 취약점 점검 도구들을 활용해서 취약점 분석을 수행하는 것이 효과적이다.

취약점 점검 도구의 종류에 따라 점검 대상이 다르거나, 동일한 점검 대상에 대해서도 취약점 점검 도구별로 점검 항목과 점검 결과가 다를 수 있다. 경험에 의해서 특정 취약점 점검 도구가 다소 우수하다고 평가할 수는 있으나, 그 점검 도구의 결과가 100% 정확하다거나 점검 대상에 존재하는 모든 취약점을 찾아냈다고 판단하기 어렵다. 따라서 네트워크 취약점 분석 평가 수행시 한 가지 취약점 점검 도구보다 다양한 점검 도구를 상호보완적인 목적으로 동시에 활용하는 것이 효과적이다.

그러나 점검 대상이 증가할수록 점검 결과의 양도 증가해서, 사람에 의한 수동적인 분석에 상당한 시간이 소요된다. 더구나 두 개 이상의 취약점 점검 도구를 동시에 사용할 경우, 점검 결과에 대한 상호 연관성 분석 과정이 필요하기 때문에 사람에 의한 수동적인 분석이 거의 불가능해진다.

본 논문에서 제안하는 자동화된 보안취약점 통합 점검 도구(Multiple Scanner Integrated Controller& Analyzer, 이하 MSCA)는 다양한 점검 도구들을 중앙 집중식으로 관리하고, 사용자의 점검 정책을 일관성 있게 적용할 수 있으며, 점검 결과에 대한 통합 분석이 가능한 시스템이다.

기존에도 전사적인 취약점 분석 시스템에 대한 연구가 있어왔고[1], 최근 개발되는 취약점 점검 도구도 취약점 점검 기능뿐만 아니라 분산된 점검 도구의 중앙 집중식 관리와 취약점 정보의 히스토리 관리를 통해 다양한 관점의 분석 결과를 보여주는 기능까지 추가하고 있다. 그러나 이들은 한 가지 점검 도구를 이용한다는 점에서 MSCA와 차이점이 있고, 이로 인해 해결해야 될 이슈가 다르다.

MSCA는 다양한 취약점 점검 도구를 활용하여 상호 보완적인 취약점 점검이 가능하고, 취약점 점검 결과의 정확성과 포괄성을 높일 수 있으며, 네트워크에 대한 종합적인 취약점 분석이 가능하다.

본 논문의 구성은 다음과 같다. 2장에서는 MSCA의 취약점 통합 분석 절차를 설명하고, 3장에서는 전체적인 시스템 구조와 모듈별 기능을 설계한다. 4장에서는 결론으로 논문을 끝맺는다.

2. 통합 분석 절차

MSCA는 점검정책 수립 단계와 취약점 점검 및 결과 수집 단계, 점검결과 통합분석 단계의 3단계로 구성된다.

점검정책 수립 단계는 공통의 취약점 점검 정책을 생성하는 단계, 점검 정책을 다중 취약점 점검 도구에 적용하고 제어하는 단계, 점검 정책을 점검 도구에 맞게 구체화 하는 단계로 세분화되는데, 이 단계에서 관리자는 모든 취약점 점검 도구에 적용이 가능한 점검 정책을 수립할 수 있고, 모든 취약점 점검 도구를 동시에 제어할 수 있다. 일관성 있는 점검 정책을 유지하기 위해, 적용된 모든 점검 정책은 데이타베이스에 저장이 되어 조회가 가능하다.

취약점 점검 및 결과 수집 단계는 취약점 점검을 동시에 수행하는 단계, 점검 결과를 일반화하는 단계, 다양한 취약점 점검 도구로부터 자동으로 점검 결과를 수집하여 저장하는 단계로 세분화되는데, 이 단계에서 관리자는 점검 정책에 따라 취약점 점검을 수행하고, 점검 결과를 공통의 포맷으로 일반화할 수 있다. 일반화된 점검 결과들은

한 곳에 수집이 되어 취약점 데이타베이스에 저장된다.

점검결과 통합분석 단계는 서로 다른 취약점 점검도구에 의해 발견된 취약점간에 상호연관성을 분석하는 단계, 점검 결과에 대한 통합 분석 및 결과 저장을 수행하는 단계, 분석 결과에 대한 관리자의 피드백 단계로 세분화 되는데, 이 단계에서 관리자는 통합 분석 결과를 조회해 볼 수 있고, 피드백을 통해 통합 분석 결과의 정확성을 높이거나 조직 환경에 적합한 대책 내용을 추가 할 수 있다.

3. MSCA 설계

3.1 시스템 구성

MSCA는 다양한 취약점 점검 도구를 원격에서 제어하고, 점검 결과를 수집하여 통합 분석할 수 있는 구조이다. 이러한 기능들은 그림 1에서 보는 바와 같이 MSCA를 구성하는 취약점 점검 도구 에이전트, 취약점 관리 및 통합 매니저, 취약점 점검 제어 및 분석 센터에 의해 구현된다.

3.1.1 취약점 점검 도구 관리 에이전트

취약점 점검 도구 관리 에이전트는 취약점 점검 도구와 함께 동일한 시스템에 설치되어서 해당 취약점 점검 도구에 대한 실행 및 제어, 점검 정책 수신 및 점검 결과 전송을 수행한다. 따라서 취약점 점검 도구 마다 적합한 에이전트가 개발되어야 한다.

MSCA 시제품에서는 개발 후 무료로 배포될 수 있는 Nessus[1], SARA[2], Nikto[3]를 위한 에이전트를 개발하고, 상용 제품 중 명령어 인터페이스를 제공하여 구현이 용이한 Languard[4]를 위한 에이전트를 개발한다.

3.1.2 취약점 관리 및 통합 매니저

취약점 관리 및 통합 매니저는 모든 에이전트와 통신하면서 각 취약점 점검도구의 점검 결과를 수집하고, 점검 결과에 대한 상호 연관성 분석을 수행하며, 분석 결과를 취약점 데이타베이스에 저장하는 기능을 수행한다.

3.1.3 취약점 점검 제어 및 분석 센터

취약점 점검 제어 및 분석 센터는 다중 취약점 점검 도구를 제어 및 실행시키고, 다중 취약점 점검 도구의 점검 결과와 상호 연관성 분석 결과를 바탕으로 비즈니스 기반의 통합 분석을 수행하여 관리자에게 웹 기반 사용자 인터페이스를 통해 보여주고, 관리자에게 통합 분석 결과에 대한 쿼리(query) 및 피드백 기능을 제공하며, 취약점 점검 정책의 일관성을 유지하기 위한 점검 정책 히스토리 관리 기능을 제공한다.

3.2 점검 정책 수립 및 관리

취약점 점검 도구 별로 점검 정책 표현 범위와 상세 수준이 다르다. 특정 점검 도구에 존재하는 점검 정책이 다른 점검 도구에는 없을 수 있고, 특정 점검 도구에서는 한 가지로 표현되는 점검 정책이 다른 점검 도구에서는 좀 더 상세한 여러 가지 점검 정책으로 표현될 수 있다.

다양한 취약점 점검 도구를 통합 관리하기 위해서 Nessus, Retina[5], SARA, Nikto, SSS[6], ISS Internet scanner[7], Web-Inspect[8], N-stealth[9], SWA3[10], Languard 의 취약점 점검 정책에 대한 분석을 바탕으로 모든 취약점 점검 도구에 공통적으로 적용이 가능한 다음 점검 정책을 도출하였다.

- 대상 IP 범위(target IP range) : 취약점 점검 대상 시스템 및 네트워크 장비의 IP 주소
- 안전 점검(safe check) : 취약점 점검으로 인해 시스템을 불안정한 상태로 만들거나 다운되지 않도록 하고, 과도한 트래픽으로 네트워크 성능에 영향을 주지 않도록 하는 옵션
- 웹 서버 IP와 포트 : 웹 취약점 점검 도구에 입력될 웹 서버의 IP 혹은 도메인 네임과 웹 서비스 포트 번호
- 웹 서버 스캐닝 : 웹서비스가 구동중인 서버만 점검
- 특정 포트 오픈 호스트 스캐닝 : 특정 포트가 오픈되어 있는 호스트만 점검
- 특정 OS 스캐닝 : 특정 OS를 구동중인 호스트만 점검
- 네트워크 장비 스캐닝 : 네트워크 장비만 점검
- 점검소요 시간 : 취약점 점검에 소요되는 시간을 상세 점검, 일반점검, 핵심점검으로 구분하여 지정
- 점검 도구 종류 및 점검 위치 선택 : 다양한 취약점 점검 도구가 설치되고 점검 도구별로 점검 영역이 다를 경우, 점검 세션마다 점검 도구의 종류와 점검 위치를 선택
- 점검 일정 : 점검 일정을 즉시 점검과 정기 점검, 특정일 점검으로 구분하여 지정
- 플러그인 업데이트 일정 : 플러그인 정보를 즉시 업데이트, 지정일 업데이트, 자동 업데이트로 구분하여 업데이트
- 점검 정책 저장 : 관리자가 적용한 점검 정책을 저장
- 과거 점검 정책 조회 : 과거에 적용되었던 점검 정책들을 시점별로 조회

보안 관리자는 다중 취약점 점검 도구를 중앙집중식으로 제어할 수 있다. 동시에 전체 점검 도구를 제어하거나 특정 점검 도구만 선택적으로 제어할 수 있다. 점검 도구를 제어하기 위한 명령어는 다음과 같다.

- 점검 시작/종료/일시정지/재시작/상태조회/정책전달

이중 일부 기능은 취약점 점검도구에서 제공되는 기능을

1) Nessus, http://www.nessus.org/
2) SARA, http://www-arc.com/sara/
3) Nikto, http://www.cirt.net/code/nikto.shtml
4) Languard, www.gfi.com/languard/
5) Retina, http://www.eeye.com/Retina/
6) Shadow Security Scanner, http://www.safety-lab.com/
7) ISS Internet scanner, http://www.iss.net/
8) Web-Inspect, http://www.spidynamics.com/
9) N-stealth, http://www.nstalker.com/
10) Shadow Web Analyzer, http://www.safety-lab.com/

그대로 사용할 수 있고, 일부 기능은 점검도구 에이전트에서 에뮬레이션 될 수 있다.

3.3 점검 정책 구체화

점검 도구별로 점검 정책의 범위와 상세 수준이 다르기 때문에, 공통의 점검 정책을 각 점검 도구의 정책에 맞게 구체화할 필요가 있다.

일부 정책은 각 점검 도구의 정책과 직접 매핑이 될 것이고, 일부 정책은 에이전트나 취약점 점검 제어 및 분석 센터에서 에뮬레이션을 통해 구현이 가능하다.

'웹 서버 스캐닝', '특정 포트 오픈 호스트 스캐닝', '특정 OS 스캐닝', '네트워크 장비 스캐닝'은 nmap등 외부 도구를 활용하여 명시된 특징을 가진 점검 대상을 식별한 뒤 점검하도록 한다.

공통 취약점 점검 정책에는 포함되지 않고 특정 취약점 점검 도구에만 존재하는 점검 정책들은 상세하고 정확한 결과가 나올 수 있도록 가능한 모든 정책을 선택하도록 한다. 다만, 많은 양의 트래픽을 생성해서 네트워크 가용 대역폭에 크게 영향을 줄 수 있는 정책, 시스템이 다운되도록 만들 수 있는 정책, 서비스 거부나 서비스 지연을 발생시킬 수 있는 정책은 '안전 점검' 선택 시 제외되도록 한다.

표 1 취약점 점검 결과 DB 스키마

필드명	설명
scan_session_id*	스캔 세션 식별자
host_ip*	취약점이 존재하는 호스트의 IP
hostname*	취약점이 존재하는 호스트명
vulnerability_name	점검결과에 명시된 취약점명
scanner_name*	취약점 발견에 사용된 점검도구 명
public_id	점검결과에 명시된 공인 취약점 ID
plugin_id	점검결과에 명시된 플러그인 ID
severity*	점검결과의 위험도(High, Medium, Low 3단계)
description*	점검결과에 명시된 설명 필드 내용
is_false_positive	기본값 FALSE. 나중에 취약점이 존재하지 않은 것으로 밝혀진 경우 TRUE로 변경
found_dt*	취약점이 발견된 일자
create_dt*	DB에 레코드가 생성된 일자

* 표시는 필수입력 필드

3.4 점검 결과 일반화

취약점 점검 도구에 따라 점검 결과의 포맷과 기술 내용이 다르기 때문에, 상호 연관성 분석을 위해서는 우선 공통 포맷으로 일반화하는 작업이 필요하다. 취약점 점검이 종료되면, 에이전트는 점검 결과를 표 1의 공통 포맷으로 변환하는 작업을 수행한다.

취약점 점검 도구별로 표현되는 위험 수준이 다른데, 위험도는 표 2의 정의에 따라 3단계로 변환하도록 한다.

표 2 위험도

위험도	설명
상	취약점 악용이 발생할 경우 심각한 문제가 발생할 수 있고, 반드시 검토하고 제거해야 하는 취약점
중	취약점 악용이 발생할 경우 심각한 문제가 발생하지는 않지만, 취약점 악용이 조직에 미치는 영향에 대해 좀 더 검토할 필요가 있는 취약점
하	취약점으로 보기 어려우며, 네트워크나 시스템 보안 관리에 참고해야 하는 수준의 정보

3.5 상호 연관성 분석

서로 다른 취약점 점검 도구들의 점검 결과를 통합 분석하기 위해서 상호 연관성 분석을 통해 동일한 취약점을 식별하는 과정이 필요하다

본 논문에서 설명하는 연관성 분석은 이종의 보안도구에 의해 발생된 이벤트간의 연관성 분석을 통해 침해사고 판단 정확도를 향상시키고 심각도를 측정하는 다른 연구들과는 차이가 있다[2]. 본 논문은 동종의 보안제품인 취약점 점검 도구의 점검 결과에 대한 연관성 분석 방법을 다루고 있는데, 사전에 플러그인 분석을 통한 취약점 정보 매핑 방법과 공인 취약점 ID에 기반한 분석 방법을 고려할 수 있다. 전자의 경우 플러그인 분석에 상당한 시간과 노력이 필요하고 신규 취약점이 발생할 때마다 추가적인 분석이 필요하며, 각 점검도구 개발 업체의 지원이 필요하다. 따라서 신속한 보안 점검 및 대응을 위해서는 플러그인 분석에 의한 매핑 방법보다 공인 취약점 ID에 기반을 둔 연관성 분석 방법이 효과적이다.

취약점 점검 결과에 여러 개의 공인된 ID가 연관되어 있을 수 있는데, 본 시스템에서는 CVE ID[3], Bugtraq ID[4], Cert advisory[5], Microsoft security bulletin ID[6]를 활용하고, 다음과 같은 우선순위를 갖도록 한다.

CVE ID > Bugtraq ID > Cert advisory > Microsoft security bulletin ID

* CVE ID를 높은 우선순위로 정한 이유는 이것이 가장 보편적으로 사용되고 있고, 벤더 중립적인 포럼과 편집위원회를 통해 검증이 수행되기 때문이다[7].

공인 취약점 ID를 기본적인 식별 방법으로 사용하고, 공인 ID가 없는 취약점에게는 No-match ID를 부여하고 그 기록을 관리하는 방법을 사용한다.

취약점의 위험도가 높으면서 공식적인 취약점 ID를 할당받지 못한 것은 최근에 발견되어 급속히 확산되고 있는 취약점이라고 볼 수 있다. 이런 경우에는 No-match ID를 발급하고, 취약점 정보를 기록하였다가 앞으로 발견되는 같은 취약점에 대해서 동일한 No-match ID를 사용하도록 한다. 추후 이 취약점에 대해 공식적인 취약점 ID가 발급된다면, No-match ID와의 매핑 정보를 저장한다.

취약점의 위험도가 낮으면서도(위험도 : 중, 하) 공인 ID가 없는 취약점은 취약점 정보로서 중요도가 낮고 취약점 통합 분석에 큰 영향을 주지 않기 때문에 포트별로 통합하여 분석하거나 각기 별도의 취약점으로 구분한다. 취약점 결과 통합 분석에서는 위험도가 상인 취약점을 중심으로 분석을 수행한다.

3.6 통합 분석

다중 취약점 점검 도구를 사용할 경우, 동일 취약점에 대한 취약점 명, 위험도, 취약점 설명이 취약점 점검도구마다 다를 수 있다.

위험도의 경우 취약점 점검 결과의 불확실성과 위험도 높은 모든 취약점을 찾아내야 한다는 점을 고려할 때, 가장 높은 위험도를 그 취약점의 통합 위험도로 사용하는 것이 바람직하다. 취약점 명은 관리자가 사전에 설정한 신뢰도가 높은 점검도구의 것을 사용하고, 취약점 설명은 관리자의 폭넓은 이해를 위해 모든 점검도구의 것을 통합한다.

4. 결 론

본 논문에서는 다양한 취약점 점검 도구를 활용할 수 있는 자동화된 네트워크 취약점 통합 분석 시스템을 설계하고, 점검 도구 통합시 발생할 수 있는 주요 문제점과 해결방안을 설명하였다.

MSCA의 자동화된 취약점 점검 및 통합 분석 절차는 대규모의 복잡한 네트워크 보안 관리에 효율적이고, 새로운 취약점 발견 후 이를 악용한 해킹 기법의 배포 및 웜바이러스의 확산이 매우 빠른 최근 경향에 대해 신속한 파악 및 대응을 가능하게 해준다.

MSCA와 단일 취약점 점검 도구의 특징을 비교하면 표 3과 같다. 이러한 특징들은 취약점 점검에 이용되는 도구의 다양성과 제공되는 취약점 정보의 상세 수준에서 비롯된다.

표 3 MSCA와 단일 취약점 점검 도구의 특징 비교

	단일 취약점 점검 도구	MSCA
장점	· 세밀한 취약점 점검 정책 수립 가능 · 취약점간의 연관성 분석이 용이함 · 신속한 업데이트가 가능함	· 취약점 점검 결과의 정확성 및 포괄성 향상 · 다양한 점검 도구 통합이 용이함 · 조직의 네트워크 환경과 재정 상황에 적합한 도구 선택 가능
단점	· 특정 점검 도구에 의존적임 · 점검 결과의 포괄성이 상대적으로 낮음	· 연관성 분석이 불가능한 취약점이 존재 · 업데이트가 여러 개발업체에 의존적임

이 시스템은 다양한 점검 도구를 상호 보완적인 목적으로 사용하여 취약점 점검 결과의 정확성과 포괄성을 향상시킬 수 있다는 것이 가장 큰 특징이다. 또한 특정 취약점 점검 도구에 의존적이지 않고 다양한 취약점 점검 도구를 쉽게 통합하여 사용할 수 있기 때문에 조직의 네트워크 환경과 재정적인 상황에 맞는 점검 도구를 유연하게 선택할 수 있다는 장점이 있다.

MSCA는 네트워크 맵 생성 기술과 통합하여 시큐리티 맵 기반의 네트워크 보안관리 시스템으로 확장될 예정이다.

참 고 문 헌

[1] C. Ying, A. Tsai, and H. Yu, "Vulnerability assessment system (VAS)," In Proceedings IEEE 37th Annual 2003 International Carnahan Conference, October 2003.

[2] 이수형, 방효찬, 장범환, 나중찬, 효과적인 보안상황 분석을 위한 보안이벤트 처리, 전자통신동향분석 제22권 제1호, 2007년 2월.

[3] Common Vulnerabilities and Exposures, http://cve.mitre.org/, Page last updated: 29-Mar-07.

[4] Bugtraq, http://www.securityfocus.com/vulnerabilities, Page last updated: 6-Apr-07.

[5] Cert advisory, http://www.cert.org/advisories/, Page last updated: 30-Jan-07.

[6] Microsoft security bulletin, http://www.microsoft.com/technet/security/current.aspx, Page last updated: 3-Apr-07.

[7] M. Rohse, Vulnerability naming schemes and description languages: CVE, Bugtraq, AVDL and VulnXML, GSEC Practical Version 1.4b(1) Apr 22, 2003.