



T. C.

**GEBZE YÜKSEK TEKNOLOJİ ENSTİTÜSÜ
MÜHENDİSLİK VE FEN BİLİMLERİ ENSTİTÜSÜ**

AKILLI AÇIKLIK TARAYICI TASARIMI

**Türker AKYÜZ
YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI**

TEZ DANIŞMANI

Yrd. Doç. Dr. İbrahim SOĞUKPINAR

GEBZE

2005



Türker AKYÜZ' ün tez çalışması, G.Y.T.E. Mühendislik ve Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 25.07.2005 tarih ve 2005 /26 sayılı kararıyla oluşturulan jüri tarafından Bilgisayar Mühendisliği Anabilim Dalında YÜKSEK LİSANS Tezi olarak kabul edilmiştir.

JÜRİ

ÜYE : Yrd. Doç. Dr. İbrahim SOĞUKPINAR
(Tez Danışmanı)

ÜYE : Yrd. Doç. Dr. Mehmet GÖKTÜRK

ÜYE : Yrd. Doç. Dr. Serdar Süer Erdem

ONAY

G.Y.T.E. Mühendislik ve Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 08.09.2005 tarih ve 2005/32 sayılı kararı.



ÖZET

Bilgisayar sistemlerine yönelik saldırılar gittikçe artmakta ve bunun sonucunda oluşan ekonomik zararlar hızla büyümektedir. Bu artışın en önemli sebeplerinden biri saldırı araçlarının artık herkes tarafından kolayca bulunur ve kullanılabilir hale gelmesidir. Çok fazla uzman bilgisi gerektirmeyen, kullanımı kolaylaştıran araç yüzlerine sahip ve birçok otomatikleştirilmiş özelliği olan saldırı araçları sayesinde saldırganlar bilgisayar sistemlerindeki zayıf noktaları rahatlıkla bulabilirler. Kötü niyetli kişiler tarafından istismar edilerek sisteme yetkisiz erişimlere olanak sağlayabilecek bu zayıf noktalara açıklık denir.

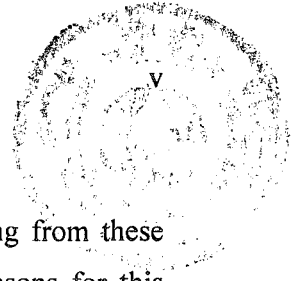
Bilgisayar sistemlerindeki açıklıkların herhangi biri tarafından istismar edilmeden önce taranıp bulunarak gerekli önlemlerin alınması, bilgi güvenliği için önemli bir adımdır. Bilgisayar sistemlerini, bilinen açıklıkları bulmak için tarayan ve bulunduğu sonuçları ve alınması gereken önlemleri rapor halinde sunan yazılımlara açıklık tarayıcı denir.

Mevcut açıklık tarayıcı yazılımları genel olarak ağ ve konak tabanlı olarak ikiye ayırmak mümkündür. Ağ tabanlı açıklık tarayıcıları, sistemin açık portları üzerindeki servislerin açıklıklarını tararken, konak tabanlı açıklık tarayıcıları sistem seviyesindeki açıklıklarla ilgilenirler.

Bu çalışmada açıklık tarayıcılar için yeni bir model önerilmiştir. Önerilen model hem ağ tabanlı hem de konak tabanlı açıklık tarayıcılarının avantajlarını kullanarak, bilgisayar sistemlerini her iki tip açıklıkları bulmak amacıyla tarar ve rapor üretir. Ayrıca önerilen model sadece bulunan açıklıkları rapor etmekle kalmayıp, bazı açıklıkları güvenlik duvarı ile beraber çalışmak suretiyle kapatarak aktif güvenlik özelliği taşır.

Önerilen model, Linux işletim sistemi platformunda, C++ programlama dili kullanılarak ve birçok açık kaynak kodlu program ve kütüphanelerden yararlanılarak gerçekleştirilmiş olup elde edilen sonuçlar ortaya konmuştur.

SUMMARY



Attacks towards computer systems and economical loss resulting from these attacks have been increasing rapidly. One of the most important reasons for this increase is attacking tools becoming easily found and used by everyone. By the help of attacking tools which do not require any expert information, have easy-to-use interfaces and a lot of automatic utilities, attackers can find weak points in computer systems easily. These weak points which may be exploited by malicious people are called vulnerabilities.

Knowing the weak points of your system and performing necessary precautions before they are exploited is a very important step for information security. Tools that scan computer systems in order to find known vulnerabilities and present the results and precautions as a report are called vulnerability scanning tools.

It is possible to divide current vulnerability scanning tools into two categories: Network-based and host-based vulnerability scanners. While network-based vulnerability scanners scan vulnerabilities of the services working on open ports of the system, host based vulnerability scanners are interested in system level vulnerabilities.

In this work, a new model for vulnerability scanners is proposed. Proposed model uses advantages of both network and host based vulnerability scanners so, it can scan the system for both kind of vulnerabilities and generates a report. Also, proposed model can fix some vulnerability by working with the firewall.

Proposed model is realized on Linux operating system platform by using C++ programming language and utilizing a number of open source programs and libraries. Also, results obtained from the model are presented.

TEŞEKKÜR



Tezime başlamamdan bitirmeme kadar bana devamlı yol gösteren ve destek veren danışmanım Sayın Yrd. Doç. Dr. İbrahim Soğukpınar' a; yardımlarından dolayı arkadaşlarım Araş. Gör. Rahim Karabağ, Dr. Hidayet Takçı ve Araş. Gör. Mustafa Çoşkun' a sonsuz teşekkürlerimi sunarım.



İÇİNDEKİLER DİZİNİ



ÖZET	iv
SUMMARY	v
TEŞEKKÜR	vi
İÇİNDEKİLER DİZİNİ	vii
SİMGELER VE KISALTMALAR DİZİNİ	ix
ŞEKİLLER DİZİNİ	x
TABLolar DİZİNİ	xi
1 GİRİŞ	1
1.1 Saldırılar ve Saldırganlar	1
1.2 Açıklık Nedir?	2
1.3 Açıklık Tarayıcı Nedir?	4
2 GENEL KAVRAMLAR VE KONUYLA İLGİLİ DİĞER ÇALIŞMALAR	8
2.1 Açıklıklar İle İlgili Kaynaklar	8
2.2 CVE (Common Vulnerabilities and Exposures)	9
2.3 Harmanlanmış (Harmonized) Açıklık Sınıfları	10
2.4 Açıklık Veritabanları	11
2.5 Açıklık Tarayıcıların Genel Özellikleri	12
2.6 Konuyla İlgili Diğer Çalışmalar	14
2.6.1 Nessus	14
2.6.2 Ferret	17
2.6.3 Cisco Secure Scanner	18
2.6.4 Açıklık Tahmini (Vulnerability Forecasting)	19
3 AKILLI AÇIKLIK TARAYICI MODELİ VE UYGULAMASI	21
3.1 Modelin Genel Özellikleri	21
3.2 İstemci Kısmı	22
3.2.1 Yönetim merkezi	22
3.2.2 İstemci Açıklık Tarama Motoru	24
3.2.3 Raporlama	27
3.2.4 Açıklık Veritabanı	28
3.2.5 Açıklıkların Düzeltilmesi	28
3.3 Sunumcu Kısmı	29
3.3.1 Sunumcu Açıklık Tarama Motoru	29

4 DENEYSEL SONUÇLAR VE ANALİZ	35
4.1 Deney Ortamı	35
4.2 Deneysel Sonuçlar	35
4.3 Diğer Açıklık Tarayıcılar ile Karşılaştırma	36
5 SONUÇ VE ÖNERİLER	38
KAYNAKLAR	40
ÖZGEÇMİŞ	42



1 GİRİŞ

Bilgisayar sistemlerine yönelik saldırıların artması ile birlikte bilgi güvenliği kavramı, gerek akademik çevrelerde gerekse devlet ve özel sektör kurumlarında bir hayli önem kazanmış ve bunun sonucunda birçok kurum çeşitli koruma yöntemleri kullanmaya başlamıştır. Virüs tarama programları, güvenlik duvarları, şifreleme yöntemleri, erişim kontrolleri, nüfuz tespit sistemleri vb. gibi araçların tamamı bilgi güvenliği amacıyla kullanılmaktadır. Ancak, bu araçların birinin veya birkaçının, hatta tamamının aynı anda kullanılması bile bir sistemin tamamen güvenli olduğu garantisizdir. Sistemde yer alan yazılım paketlerinden birinde olabilecek bir güvenlik problemi kötü niyetli kişilerin bu problemi istismar etmek suretiyle sisteme yetkisiz erişimine olanak verebilir. Bu durumda alınan bunca güvenlik önlemi bir işe yaramayacaktır. O halde; bir sistemin ne kadar güvenli olduğu en zayıf noktasının ne kadar güvenli olduğu ile ilgilidir.

1.1 Saldırılar ve Saldırganlar

Bilgi güvenliği, bilgisayar sistemlerinin ve bu sistemleri oluşturan verilerin, ağı ve makinelerin dışarıdan veya içeriden gelebilecek her türlü saldırılara karşı korunmaya çalışılması anlamına gelir. Saldırı, sisteme girmeye yetkisi olmayan kişilerin sistemi kırmaya veya kaynaklarını yanlış kullanmaya çalışmasıdır. Saldırıların genellikle; sistemin yetkilendirme için şifre koruması kullanmadığı veya zayıf şifreler kullandığı zamanlarda, sistemin doğru yapılandırılmadığı zamanlarda veya sistemin bazı yazılım komutlarının uygulanması ile ortaya çıkan açıklıklar içeren yazılım paketlerini çalıştırdığı zamanlarda meydana gelir. Kısacası bir saldırı başarılı bir saldırı gerçekleştirmek için öncelikle sistemin bir açığını bulmalıdır. Çoğu durumda saldırıların yazılım paketlerindeki çok iyi bilinen açıklıkları istismar etmeye çalıştıkları bilinmektedir. Son Symantec Internet Tehdit Raporu [Symantec Corporation, 2005] tespit edilen atakların %47' sinin keşif (reconnaissance) sınıfına girdiğini ortaya koymuştur. Yani yapılan atakların neredeyse yarısı hedef sistem hakkında bilgi elde etmeye yöneliktir. Ayrıca, en çok istismar edilen ilk 5 atak tipine ait ataklar toplam atakların %50' sini oluşturmaktadır. Yüzlerce atak tipinin mevcut olduğunu düşünürsek sadece 5 tanesinin tüm atakların yarısını oluşturması anlamlıdır. Bu istatistikler çoğu

saldırmanın bir sistemde çok iyi bilinen bir açıklık bulamazsa daha kolay başka bir hedefe yöneldiğini göstermektedir.

1.2 Açıklık Nedir?

Açıklık (vulnerability) kelimesinin sözlük anlamı zayıflık veya korumanın yetersiz olmasıdır. Bilgi güvenliği alanında ise açıklığın tanımı şu şekilde yapılabilir: Bir yazılım içerisinde bilerek veya bilmeyerek yapılmış bir hata neticesiyle oluşan ve istismar edilmesi durumunda sistem için tehdit oluşturabilecek zayıflıklara açıklık denir. Programcılar yazılımlarını geliştirirken birçok şekilde hata yapabilir ve bu hataların bir kısmı güvenlik ile ilgili bir tehdit oluşturabilirler. Programcılar tarafından yapılabilecek ve açıklığa sebep olabilecek hatalar şu şekilde sınıflandırabilir:

- Sınır koşulu hataları: Genellikle tampon bellek taşması (buffer overflow) olarak da bilinir. Tampon belleklerin, dizilerin vs. sınır boyutlarının yeterince iyi kontrol edilmemesi sonucunda oluşurlar.
- İstisnai durumların halledilmemesi: İstisnai durumlar (exceptional condition) oluştuğunda sistemin hatalı durumu güvenli bir şekilde işleyebilmesi için istisnai durumların halledilmesi gerekir.
- Erişim doğrulama hatası: Yetki ve erişim izinlerinin yeterince kontrol edilmemesi sonucunda oluşurlar.
- Kaynak doğrulama hatası: Örneğin güvenli olmayan dosya yaratılması veya sembolik bağlantıların (symbolic links) yanlış kullanımı gibi hatalardır.
- Girdi doğrulama hatası: parametrelerin ve argümanların program veya fonksiyon tanımlamalarına uymaması durumunda oluşan hatalardır.
- Konfigürasyon hataları: İstenilen güvenlik seviyesine ters düşecek ayarlamaların yapılması durumunda oluşurlar. Örneğin zayıf şifreler gibi.
- Program bütünlüğünün sağlanamaması: Programın kötü niyetli davranması ile oluşurlar (örneğin e-mail virüsleri ve solucanlar gibi).

Yazılım kodunda yapılabilecek küçük bir hata sonucu tahmin edilemeyecek boyutta maddi ve manevi zararlara sebep olabilir. Zaman ve işgücü kaybı gibi maddi

kayıpların yanında itibar kaybı da söz konusu olabilir. Örneğin, 2003 yılının Ağustos ayında Windows 2000 ve XP sistemlerindeki RPC (Remote Procedure Call) protokolündeki bir açıklığı istismar eden MSBlaster isimli bir solucan Internet üzerinde hızla yayılmıştır. Tahminlere göre bu solucan ve türevleri en az bir buçuk milyon sisteme bulaşmış ve yarattığı trafik ile birçok kurumun ağını tıkamıştır. MSBlaster solucanının istismar ettiği açıklık, TCP/IP tabanlı mesaj alış verişi gerçekleştirilen bir RPC fonksiyonu içindeydi [Schultz, 2003]. Kötü niyetli bir istemci, kurban olarak seçilen sunucuya göndermek üzere özel parametreler kümesi yaratarak RPC DCOM ara yüz açıklığını istismar edebilir. Parametreler bir RPC paketinin içine konur ve paket sunucuya gönderilir. Bu açıklığın olduğu bir sistemde sunucu parametreleri alır ve açar. Bu da parametrelerin sistem seviyesinde yetkilere sahip biçimde çalıştırılmasına sebep olur. Böylece saldırgan sistem seviyesinde komutlar çalıştırma yetkisini kazanır.

90' lı yıllarla birlikte bilgisayar sistemlerinin gelişmesi ve İnternet' in kullanımının artması ile birlikte bilgisayar sistemlerine yönelik saldırılar hızla artmıştır. 1990 yılında 252 saldırı vakası rapor edilirken 2003 yılında bu rakam 137,529' a ulaşmıştır [CERT/CC, 2005]. Aynı şekilde keşfedilen açıklıkların sayısı da bu zaman içerisinde bir hayli artmıştır. Tablo 1.1' den de görüleceği gibi son on yılda keşfedilen açıklıkların sayısı 20 kata çıkmıştır. Bunun anlamı gelecekte de bilgisayar sistemlerinde birçok açıklık var olacaktır.

Tablo 1.1. Keşfedilen açıklıkların yıllara göre değişimi [CERT/CC, 2005].

Yıl	1995	1996	1997	1998	1999
Açıklık Sayısı	171	345	311	262	417
Yıl	2000	2001	2002	2003	2004
Açıklık Sayısı	1090	2437	4129	3784	3780

Şüphesiz ki, bir bilgisayar sistemini oluşturan en önemli parçalardan biri işletim sistemidir. İşletim sistemleri de büyük birer yazılım olduklarına göre onlar da bünyelerinde birçok açıklık barındırırlar. Günümüzde en popüler işletim sistemlerinden biri olan Windows işletim sistemlerinin yayımlayıcısı Microsoft' un Internet sitesine bakacak olursak, Windows' un değişik sürümleri için çeşitli açıklıkları düzeltmek amacıyla yamalar (patch) ve bunlarla ilgili bültenler yayımlandığını görebiliriz. Bu durum sadece Windows için değil hemen hemen tüm

işletim sistemleri için geçerlidir. Tablo 1.2' de bazı işletim sistemlerinin açıklıklarının bulunma eğilimleri gösterilmiştir.

Tablo 1.2. Bazı işletim sistemlerinde açıklıkların bulunma eğilimleri [Lee and Davis, 2003].

İşletim Sistemi	Top. açıklık sayısı	Ort. Bul. oranı (açıklık/ay)
Solaris	88	2.9
HP-UX	52	1.8
Red Hat Linux	127	3.8
Caldera OpenLinux	94	3.1
Windows NT	143	4.8

FBI ve The SANS Institute' ün 2004 yılında yayınladığı ortak liste (Top 20 List) UNIX ve Windows ortamlarında en fazla istismar edilen 20 açıklığı tanımlamıştır. Bu rapor, işletim sistemlerine yapılan başarılı saldırıların büyük bir kısmının sadece birkaç yazılım açığından kaynaklandığını ortaya koymuştur. Windows ve UNIX işletim sistemlerinde en çok istismar edilen açıklıklar Tablo 1.3' te gösterilmiştir.

Tablo 1.3. Windows ve Unix ortamlarında en çok istismar edilen açıklıklar.

#	Windows	Unix
1	Web Sunucuları ve Servisleri	BIND Alan Adı Sistemi
2	Workstation Servisi	Web Sunucusu
3	Windows Uzaktan Erişim Servisleri	Kimlik Doğrulama
4	Microsoft SQL Server	Versiyon Kontrol Sistemi
5	Kimlik Doğrulama	SMTP
6	Web tarayıcıları	SNMP
7	Dosya Paylaşım Uygulamaları	Open SSL
8	LSASS	NIS/NFS
9	Mail İstemcileri	Veritabanları
10	Instant Messaging	Kernel

1.3 Açıklık Tarayıcı Nedir?

Saldırıya maruz kalma riski, güvenlik duvarları, virüs yazılımları, nüfuz tespit sistemleri ve açıklık tarayıcıları gibi çeşitli güvenlik teknolojileri kullanılarak minimuma indirilebilir. Bu güvenlik teknolojilerinin son zamanlarda bilgisayar ve ağlara yapılan saldırılara karşı büyük başarılar kazanmalarına rağmen hala birçok

yönden yetersiz kalabilmektedirler. Nüfuz tespit sistemlerinin birçok yanlış alarm vermesi, güvenlik duvarlarının yeterince akıllı olmaması ve çok fazla insana bağımlı yapılandırmaya ihtiyaç duymaları veya anti-virüs yazılımlarının çok sık güncellenme gerekliliği buna örnek olarak verilebilir.

Güvenlik problemlerini halletme konusunda iki temel yaklaşım vardır: Birincisi, saldırı gerçekleştikten sonra saldırının nasıl gerçekleştirildiğinin araştırılıp bulunması veya saldırı devam ederken saldırının farkına varılması böylece bir daha aynı yöntemin kullanılmasının engellemesidir. Bu tip güvenlik önlemlerine proactive güvenlik denir. Nüfuz tespit sistemleri bu tür güvenlik araçlarıdır. İkinci yaklaşım ise saldırganlar sistemdeki zayıf noktaları keşfetmeden önce bu noktaların bulunarak güvenli hale getirilmesidir. Bu tür güvenliğe ise reactive güvenlik denir.

Bilgi ve ağ güvenliği teknolojilerinin bir diğer örneği de açıklık tarayıcılarıdır. Açıklık tarayıcılar reactive güvenlik araçlarıdır. Bu teknoloji yeni olmamasına rağmen gelişimini sürdürmektedir. Açıklık tarayıcıları kısaca, bilgisayar sistemlerini, bilinen açıklıkları bulmak için tarayan güvenlik araçlarıdır. Açıklık tarayıcı araçlar, yetkisiz kullanıcıların sistemdeki açıklıkları kurumun bilgi hazinesine erişim hakkı elde etmek için kullanmalarına karşı sistem ve ağ yapılandırmalarını inceler. Tarayıcı araçların çoğu sistem yöneticileri tarafından periyodik olarak çalıştırılan programlar ve scriptlerdir.

Açıklık tarayıcıları genellikle diğer güvenlik teknolojileri ile birlikte kullanılırlar ve bu araçlarla sıkı bir ilişki içindedirler. Örneğin açıklık tarayıcıları ve nüfuz tespit sistemleri arasındaki ilişki geliştirilmesi gereken ilginç bir konudur. Kısaca nüfuz tespit sistemleri, bilgisayar sistemindeki olayları izler ve belli nüfuz imzalarını tanımak amacıyla bu olayları analiz eder. Bunun anlamı nüfuz tespit sistemlerinin olası saldırıları meydana geldikten sonra tespit etmeleridir. Oysa, saldırı gerçekleştikten sonra geç kalınmış olabilir. Açıklık tarayıcıları ise istenmeyen olaylar olmadan önce açıklıkları tarar. Uluslar arası Standartlar Örgütü (ISO) açıklık tarama teknolojilerini kurumun açıklıklarının, saldırganlara istismar etme fırsatı vermeden önce bulmasını sağlayan araçlar olarak tanımlar. Açıklık tarayıcılar ile nüfuz tespit sistemleri arasındaki ilişki şöyle tanımlanabilir: Açıklık tarayıcıları nüfuz tespit sistemlerinin aradıkları saldırıları büyük ölçüde azaltırlar.

Açıklık tarayıcılar önceleri platforma özel kaynak kodu modülleri olarak inşa edilmişlerdi. Donanım platformlarında, işletim sistemlerinde ve uygulama yazılımlarındaki büyük gelişmelerle birlikte açıklık tarayıcıları bu yeni platformlarda çalışmak üzere yeniden inşa edilmek zorunda kalmışlardır. Aslında ilk nesil tarayıcılar saldırı aracı olarak tasarlanmıştı fakat zaman geçtikçe daha karmaşık hale geldiler ve savunma amacıyla da kullanılmaya başladılar. Bir ağdaki her bir konak üzerinde çalışmak yerine bir host üzerinden birden çok platform ve işletim sistemini tarayabilmektedirler. Birden çok hosttan gelen verilerin analiz edilebilmesi ile sistem yöneticileri sistemin bütünündeki potansiyel açıklıkları görebilir.

Genellikle konak ve ağ tabanlı olmak üzere iki tip açıklık tarayıcı vardır. Ağ tabanlı tarama araçları ağ trafiğini, çeşitli ağ konaklarına ve makinelerine bu sistemlerdeki istismar edilebilecek açıklar hakkında bilgi toplama amacıyla gönderir [Humphries, 2000]. Bu tip araçlara örnek olarak port tarayıcıları verilebilir. Port tarayıcıları herkesçe bilinen portlar üzerinde düzgün yapılandırılmamış servisleri arar. Konak tabanlı araçlar ise her bir konak üzerinde çalışır ve yetkilendirilmemiş yazılım, yetkilendirilmemiş kullanıcı hesabı, zayıf şifreler, virüsler ve uygun olmayan erişim yetkileri gibi sistem problemlerini tararlar [Humphries, 2000].

Her iki modelin de kendine göre avantajları vardır. Ağ tabanlı tarayıcılar sistemi dışarıdan görürler. Hedef konak üzerinde başka bir program kurmadıklarından sistem seviyesinde erişim hakları yoktur ve bu nedenle işletim sistemi ile ilgili veya konak üzerinde çalışan yazılımlarla ilgili bilgi elde edemezler. Bu araçlar sisteme sadece sistemin dışarı açılan kapısı olan portlarla ulaşabilirler. Ayrıca yine konak üzerinde başka bir program kurmadıklarından bulunan açıklıkların düzeltilebilmesi ile ilgili hiçbir şey yapamazlar. Öte yandan konak tabanlı açıklık tarayıcıları ise sadece bulundukları makineye kurulan programlar oldukları için yalnızca üzerlerinde çalışan işletim sistemine ait açıklıkları, kullanıcıların sebep olduğu açıklıkları, registry ile ilgili bozuklukları veya konak üzerinde kurulmuş yazılımlara ait açıklıkları bulabilirler. Ancak sistemi dışarıdaki bir saldırgan gibi göremediklerinden dışarıdan gelebilecek saldırıların sistemin hangi noktasından gelebileceğini anlamakta yeterli değildirler. Bu sayılan sebeplerden dolayı her iki açıklık tarayıcı modelinin de kendine göre avantajları ve dezavantajları vardır.

İster ağ tabanlı ister konak tabanlı olsun mevcut açıklık tarayıcı programların çoğu, sistemi bilinen açıklıklara göre tarayarak, veritabanlarında bulunan açıklık imzalarıyla karşılaştırırlar ve buldukları açıklıkları ve bu açıklıklar ile ilgili ne önlem alabileceklerini rapor halinde sunarlar. Ancak genellikle bunun pek ötesine gitmezler ve açıkların kapatılmasına yönelik aktif bir hareket yapmazlar.

Bu tezde önerilen model, hem ağ hem de konak tabanlı açıklık tarayıcıların avantajlarından yararlanmakta ve her iki tip tarayıcıların bazı dezavantajlarını ortadan kaldırmaktır. Ayrıca modelin diğer bir avantajı da bulunan açıklıkların aktif olarak düzeltilmesine olanak sağlamaktır. Önerilen modeldeki “akıllı” sıfatının anlamı, bulunan açıklıkların bir adım ileri giderek bir saldırgan gibi istismar edilmeye çalışılıp sistemde açık olduğunun kanıtlanması ve güvenlik duvarı ile beraber çalışarak açıklıkların kolayca kapatılabilmesidir.



2 GENEL KAVRAMLAR VE KONUYLA İLGİLİ DİĞER ÇALIŞMALAR



2.1 Açıklıklar İle İlgili Kaynaklar

Ticari veya açık kaynak kodlu yazılım sistemlerindeki açıklıkların bulunması kritik bir aşamadır. Bu açıklıkların düzeltilmesi için neye ihtiyaç olduğunu ve bunların nasıl temin edilebileceğinin bilinmesi halinde, basit bir yama, güncelleme ya da konfigürasyonun değiştirilmesi çoğu açıklıkların yok edilmesi için yeterli olabilir. Kullanılan yazılımlardaki açıklıkları tanımlayabilmek için araştırmak, biraz da para harcamak gerekebilir. Çoğu ticari yazılım kullanıcıları, satın aldıkları yazılımların içyapılarını ve mimari detaylarını bilmezler. En fazla yazılım paketlerinin genel mimarisi ve tasarım mantığını anlayabilirler. Ticari yazılım üreticileri genellikle yazılımın kodunu ve tasarım detaylarını şirket sırrı olarak görürler ve oluşabilecek problemleri müşterileri ile paylaşmakta istekli değildirler. Bu üreticiler kullanıcılara yazılım açıklıkları hakkında bilgi vermezse bu açıklıklar nasıl bulunabilir? Bugüne kadar bu soruya cevap arayan üç grup çıkmıştır: Hackerlar, ticari amaç güdenler ve hayırseverler [Martin, 2001]. Hackerlar iyi niyetli veya kötü niyetli olabilir. Kötü niyetli hackerlar, yazılım açıklıklarını bularak bu açıklıkları kötü amaçlar için kullanabilir ve elde ettikleri bilgileri kendileri gibi düşünen kişilerle paylaşırlar. Diğer grup ise sisteme zarar vermekten çok gerekli kişilere sistemde açık bulunduğunu göstermeye çalışırlar. Ticari amaç güdenler ise genellikle yazılım ve ağ güvenliği şirketleri olup yazılım hataları ile ilgili danışmanlık hizmeti verirler veya bu hataları bulabilecek araçları sağlarlar. Hayırseverler olarak adlandırılan grup üyeleri ise akademik kuruluşlara, devlet kuruluşlarına veya kar amacı gütmeyen kurumlara mensup güvenlik araştırmacılarıdır. Bu tür kişiler elde ettikleri bilgileri ve geliştirdikleri araçları ücret talep etmeden paylaşırlar.

Yazılım ürünlerindeki açıklıklar hakkındaki bilgiler kolayca bulunabilir. Yukarıda belirttiğimiz üç grubun elemanları bulunan yeni açıklıklar ile ilgili bilgileri web sitelerinde, haber gruplarında, e-mail listelerinde veya çeşitli bültenlerde yayınlarlar ve işle ilgilenenleri devamlı bilgilendirirler. Akademik veya devlet kuruluşlarında görev yapan araştırmacılar ve bazı şirketler herhangi bir ücret talep

etmeden, yaygın olarak kullanılan ve çok değerli duyuru, alarm ve tavsiye servisleri önerirler. Tablo 2.1' de bu kuruluşların listesi verilmiştir.



Tablo 2.1. Çeşitli alarm ve tavsiye servisleri.

Servis	Servis Tipi
CERT Advisories	Tavsiye
Bugtraq	e-posta grubu
Cassandra	Alarm
Cybernotes	Aylık bülten
RAZOR	Tavsiye
SAFER	Aylık bülten
SANS NewsBites	e-posta grubu
Security Alert Consensus	e-posta grubu
X-Force Alerts	Tavsiye

Açıklık içeren yazılımları yaratan şirketler de bazı çözümler önerebilmektedirler. Çoğunun yazılımdaki hataların düzeltilmesi veya güncellenmesi için kendi yöntemleri vardır. Bundan kısa bir süre öncesine kadar yazılım şirketlerinin çoğu, yamaların ve güncellemelerin dağıtımı konusunda pek istekli değilken artık, IBM, Microsoft, SGI ve Sun Microsystems gibi büyük yazılım devleri bu tip güvenlik çözümleri önermektedir.

2.2 CVE (Common Vulnerabilities and Exposures)

Bundan birkaç yıl öncesine kadar çeşitli açıklık servisi, aracı, veritabanı veya üreticilerin kendi güncellemeleri kullanılmak istenildiğinde büyük bir sorunla karşı karşıya kalınıyordu. Buradaki sorun, yeni bir açıklık bulan kişi veya grubun sanki bu bilginin kaynağının sadece kendileriymiş gibi görmeleridir ve buldukları bilgiyi isimlendirirken, tanımlarken ve paylaşıırken kendi yaklaşımlarını kullanmalarındır. Bundan dolayı bir yazılımdaki tek bir açıklık için birden fazla isim ortaya çıkıyordu. Bu durum, kişilerin hangi açıklıkla karşı karşıya olduklarını anlamalarını ve bunun için hangi aracı kullanmaları gerektiğini bulmalarını zorlaştırıyordu. Bu sebeple MITRE isimli şirket, bu isim karışıklığını önlemek için bir metot geliştirmeye başladı. Bu metot, her bir açıklık için tek bir isimden oluşan bir referans listesi yaratmayı ve bu listedeki elemanları uygun araçlar veya veritabanları ile eşleştirmeyi amaçlıyordu. Daha sonra MITRE şirketi bir makale [Mann and Christey, 1999] yayınlarak CVE inisiyatifinin (Common Vulnerabilities And Exposures Initiative)

temel düşünce ve yaklaşımını açıkladı. CVE önceleri basit olarak açıklık ile ilgili veritabanlarının birleştirilmesi olarak düşünülmüştü. Çeşitli servislerin ve yazılım ürünlerinin ortak isimler kullanması bilgi güvenliği çevreleri için son derece önemlidir. CVE bu konuda artık bir standart olmuştur ve CVE list olarak bilinen standartlar listesi bugün birçok kurum tarafından kabul edilmiştir. CVE' nin en önemli özellikleri şunlardır:

- Her bir açıklığın tek bir ismi olacaktır.
- Her bir açıklığın standart bir tanımı olacaktır.
- CVE listesi bir veritabanından çok bir sözlük niteliğinde olacaktır.
- Liste ve CVE'nin diğer aktiviteleri Internet' ten yayınlanabilecektir.
- CVE uyumlu ürünlerin ve servislerin onaylanması, CVE bünyesindeki bir jüri tarafından yapılacaktır.

CVE uyumunun anlamı bir güvenlik aracının, web sitesinin, veritabanının veya servisin kullandığı CVE isimlerinin diğer CVE isimlerini kullanan kaynaklarına referans verebilmesidir. 2005 yılının Haziran ayı itibariyle CVE uyumlu ürün sayısı 223' tür. Örnek bir CVE ismi ve tanımı Tablo 2.2 'deki gibidir. Eğer açıklık CVE jürisi tarafından kabul edilmiş ise CVE ön eki, değerlendirme aşamasında ise CAN ön eki alır. Daha sonra açıklığın bulunduğu yıl ve o yılki sırası belirtilir.

Tablo 2.2. Örnek CVE ismi ve tanımı.

İsim	CVE-1999-0047
Tanım	MIME conversion buffer overflow in sendmail versions 8.8.3 and 8.8.4.
Referanslar	- CERT: CA-97.05.sendmail - BID: 685 - XF: sendmail-mime-bo2
Tarih	11.09.2005

2.3 Harmanlanmış (Harmonized) Açıklık Sınıfları

Günümüz açıklık tarayıcı sistemlerinin hepsi birbirinin tam olarak aynı açıklıkları taramadığından farklıdırlar. Örneğin, bir açıklık tarayıcısı şifre açıklıkları için on çeşit açıklık tanımlarken bir diğeri dört tane tanımlamış olabilir. Bir kurum için hangi açıklık tarayıcısının daha iyi olduğunu bulmak için açıklık tarayıcılarının

her birinin hangi tip açıklıklara özel olduğunu bilmek gerekir. Bazı firmalar donanımsal açıklıkları daha iyi yakalayan tarayıcıları tercih ederken bazıları yazılım veya ağdaki açıklıkları daha iyi tespit edebilen açıklık tarayıcılarını tercih eder. Bazı kurumlar ise farklı alanlarda başarılı olan tarayıcıları birlikte kullanarak birçok tip açıklığa karşı tarama gerçekleştirmek isteyebilir.

Bir kurumun kendi gereksinimlerine en uygun açıklık tarayıcı sistemi bulmak için çeşitli tarayıcılar arasında bir değerlendirme yapması gerekir. Bunu gerçekleştirebilmek için tarafsız metotlar sağlayan harmanlanmış (harmonized) açıklık sınıfları kullanılabilir. Harmanlanmış açıklık sınıfları çeşitli açıklık tarayıcıların taradığı tüm açıklıkların sınıflandırılmasıdır. Bu sınıflandırma gelecekte yeni açıklıklar bulundukça değişime uğrayabilir fakat mevcut açıklıkları sınıflandırmak için yeterli olmaktadır. Tablo 2.3' de harmanlanmış açıklık sınıflarının bir listesi gösterilmektedir. 13 sınıftan oluşan bu liste günümüzdeki açıklık tarayıcı sistemleri hakkındaki tüm literatür ayrıntılı şekilde taranarak oluşturulmuştur [Venter and Eloff, 2003].

Tablo 2.3. Harmanlanmış Açıklık Sınıfları.

Kategori No	Kategorinin Tanımı
1	Şifre kırma ve çalma
2	Ağ ve sistem bilgisi elde etme
3	Kullanıcı listesi ve bilgileri
4	Arka kapılar, truva atları ve uzaktan kontrol
5	Uzaktaki bağlantı ve servislere yetkisiz erişim
6	Kullanıcıları ve önceliklerini değiştirme
7	Kimlik gizleme ve yanıltma
8	Yanlış konfigürasyonlar
9	Servisin inkârı ve tampon bellek taşmaları
10	Virüsler
11	Donanımsal özellikler
12	Yazılımsal özellikler ve güncellemeler
13	Güvenlik politikası ihlalleri

2.4 Açıklık Veritabanları

Bir tarayıcının bir açıklığı tespit edebilmesi için öncelikle o açıklığın tarayıcının veritabanında olması gerekir. O halde, taramanın ne kadar kapsamlı olacağı açıklık veritabanının büyüklüğü ile ilgilidir. Açıklık veritabanı taramanın

sonucunda hazırlanacak rapordaki bilgileri depolar. Bu bilgiler şu şekilde sıralanabilir:

- Açıklığın kimlik numarası (CVE ismi vb.)
- Açıklığın başlığı
- Açıklığın tanımı
- Açıklığın bulunma tarihi
- Risk faktörü yani açıklığın verebileceği zararın boyutu
- Açıklık ile ilgili diğer kaynaklara referanslar
- Hangi işletim sistemlerine veya yazılımlara zarar verebileceği
- Açıklığı düzeltmek için gereken çözüm önerileri

2.5 Açıklık Tarayıcıların Genel Özellikleri

Piyasada farklı teknolojilere, farklı özelliklere, farklı fiyatlara göre birçok açıklık tarayıcı mevcuttur. Taranılan açıklıkların sayısı, taramanın doğruluğu veya üretilen raporların niteliği bir tarayıcıdan diğerine çok farklılık gösterebilir. Açıklık tarama işleminde sistemin güvenlik politikalarına uygun tarayıcıyı seçmek önemli bir adımdır. Tarayıcıları seçerken kriter olarak kullanılabilecek bazı özellikler şöyle sıralanabilir:

- Tarayıcının kullandığı teknoloji
- Veritabanındaki açıklıkların sayısı
- Raporlama özellikleri
- Karşı önlem önerileri
- Analizlerinin doğruluğu
- Performans

Açıklık tarayıcıları karşılaştırırken bakılması gereken ilk nokta tarayıcının esas aldığı teknolojidir. Tarama çözümleri açık portları tarayan port tarayıcılarından güvenlik yamalarını, kayıt dosyalarını tarayan hatta sisteme atak yaparak test eden tarayıcılara kadar değişir. Tarama işlemi aktif veya pasif olarak sınıflandırılabilir. Pasif taramada taranan sistemin çökmesine yol açabilecek ağ trafiği üretilmez. Aktif taramada ise ağda belirli bir trafik üretilir. Açıklıkların tarama biçimi farklılık

gösterir. Çeşitli açıklık tarayıcıları aynı tip açıklıkları farklı yöntemlerle tarıyor olabilir. Mesela, bir tarayıcı şifrelerin güvenliğini ölçmekte sözlükteki kelimeleri deneme yöntemini (dictionary attack) kullanırken diğeri, bütün karakter kombinasyonlarının tek tek denendiği yöntemi (brute-force attack) kullanıyor olabilir.

Veritabanındaki açıklıkların sayısı ne kadar kapsamlı bir taramanın yapılacağına göstergesidir. Bir tarayıcı yüzlerce sonuç döndürebilir fakat sonuç verisi anlaşılır bir şekilde rapor edilmez ise bir işe yaramaz. Açıklıkların taranıp bulunması işin yarısıdır. İyi bir açıklık tarayıcı tarama sonuçlarını iyi analiz edip karşı önlem önerilerini sağlamalıdır. En önemli kriterlerden biri de doğruluk oranıdır. Yanlış pozitiflerin (gerçekte olmayıp rapor edilen) ve yanlış negatiflerin (gerçekte olup rapor edilemeyen) oranının düşük olması gerekir. Bazı tarayıcılar sadece açıklığı bulmakla kalmaz onu istismar etmeye çalışarak açıklığın gerçekten var olduğunu kanıtlamaya ve yanlış pozitiflerin oranını azaltmaya çalışırlar. Fakat bu probleme yol açabilir. Açıklık tarayıcılarının tarama işlemi sırasında sistemin performansını mümkün olduğu kadar az etkilemelidir.

Genel olarak açıklık tarayıcıları iki grupta incelenebilir: Ağ tabanlı (network-based) açıklık tarayıcıları ve konak tabanlı (host-based) açıklık tarayıcıları. Ağ tabanlı tarayıcılar, açık portları tararlar, bu portlar üzerinde çalışan servisleri tanırlar, atakları simule ederler ve bu servislerle ilişkili olası açıklıkları ortaya çıkarmaya çalışırlar [Guirguis, 2003]. Zayıf işletim sistemi servisleri, DNS sunucuları, servisin inkârı atakları ve düşük seviyeli protokol zayıflıkları gibi açıklıklar ağ üzerinden daha rahat araştırılabilirler. Ağ tabanlı açıklık tarayıcıları sistemi kırmaya çalışan dâhili veya harici saldırganların bakış açısı ile tüm ağ yapısını detaylı biçimde analiz ederler. Bu tip tarayıcılar konak yazılımının kurulumuna ihtiyaç duymadıklarından hızla kurulup kullanılabilirler ve konak üzerinde yönetici seviyesinde erişim hakkına ihtiyaç duymazlar. Ancak çeşitli ağ kaynaklarını kullandıklarından ağın yavaşlamasına sebep olurlar. Güvenlik duvarları tarafından engellenebilirler. Konak üzerinde bir program kurmadıklarından yönetici seviyesinde erişim hakları yoktur. Bu nedenle buldukları açıklığı düzeltmek için bir işlem yapamazlar.

Öte yandan konak tabanlı tarayıcılar ise sistem güvenliğini kullanıcı bakış açısı ile değerlendirir. Yerel bir hesaba sahip her kullanıcı yerel sistemin kontrolünü ele geçirmek ve istismar etmek için birçok olasılığı beraberinde getirir. Konak tabanlı bir tarayıcı sistemin doğru yapılandırıldığını ve açıklıkların yamalandığını ve dolayısıyla yerel kullanıcıların yönetici seviyesinde erişim kazanamayacağını garanti eder. Hedef konak üzerinde bir program biçiminde olduklarından yönetici seviyesinde erişim hakları vardır. Bu tip tarayıcıların en büyük dezavantajı sistemi dışarıdan bir saldırgan gibi görememeleridir.

2.6 Konuyla İlgili Diğer Çalışmalar

Günümüzde pek çok açıklık tarayıcı yazılım mevcuttur. Bir kısmı ücretsiz bir kısmı da ticari olan bu yazılımların tümünün kendine has özellikleri vardır. En çok kullanılan açıklık tarayıcılar ve kullanım oranları Tablo 2.4 'te gösterilmiştir.

Tablo 2.4. Açıklık tarayıcıların kullanım oranları (Sunbelt, 2004).

Açıklık Tarayıcı	Kullanım Yüzdesi
Microsoft Baseline Security Analyzer	%20
Nessus	%17
GFI LanGuard	%14
Retina	%13
Internet Security Scanner	%11
Nmap	%6

2.6.1 Nessus

Nessus ücretsiz, açık kaynak kodlu ve ağ tabanlı bir açıklık tarayıcıdır. Nessus'un genel özellikleri şu şekilde sıralanabilir:

- İstemci/sunumcu mimarisi sayesinde esnek bir yapıya sahiptir. Tüm tarama işini yapan merkezi bir sunucu ve tarama işini başlatan ve sonuçların gösterildiği dağıtık istemciler bulunur. Tarama motoru UNIX tabanlı iken istemciler hem UNIX hem de Windows tabanlı olabilir.
- CVE uyumludur. Yayınlanmış açıklıklar hakkında bilgi almak için CVE'ye bağlanır. Ayrıca CERT, Bugtraq ve üretici firmaların güvenlik bültenlerine de referansı vardır.

- Her bir güvenlik testi harici eklentiler olarak yazılmıştır. Böylece kullanıcılar kaynak kodu incelemeden de bu eklentiler yardımıyla kendi testlerini yazabilirler. NASL (Nessus Attack Scripting Language) isimli bir dil sayesinde daha kolay ve hızlı testler yazılmasını sağlar. Ayrıca testler C dilini kullanarak da yazılabilir.
- 37 ayrı kategoriye ayrılmış 4000'den fazla test içerir. Bu kategorilerden bazıları şunlardır: Arka kapılar, CGI suistimalleri, güvenlik duvarları, FTP, Netware, NIS, yetkisiz root hakkı kazanma, RPC, SNMP ve SMTP problemleri, çeşitli işletim sistemlerine özel açıklıklar vs.

Nessus' u çalıştırmak için önce UNIX tabanlı sunumcu kısmının kurulup gerekli konfigürasyonların yapılması gerekir. Daha sonra aynı sisteme veya başka bir sisteme ister UNIX tabanlı ister Windows tabanlı istemciyi kurmak gerekir. İstemci, taramanın parametrelerini belirledikten sonra taramayı başlatır ve sonuçlar rapor halinde tekrar istemciye geri döner. Nessus' un tarama için oldukça kapsamlı parametreleri vardır [Mitchell, 2002]. Daha önce de belirttiğimiz gibi testler harici eklenti şeklindedir. Bazı eklentilerin taramaya dâhil edilmesi hedef konağı çökertebilir. Bu nedenle güvenli tarama seçeneği geliştirilmiştir. Nessus port tarama, işletim sistemi tanıma, bilgi toplama, açıklık tarama ve saldırı simülasyonu safhalarından oluşur. Bazı safhalarda açık kaynak kodlu diğer güvenlik araçlarını da kullanır. Örneğin, işletim sistemini tanıma ve port tarama için Nmap, web server ve CGI atakları için Nikto ve Whisker programlarından yararlanır.

Nessus' un performansını yapılan testlerle gözlemlemek mümkündür. Bu amaçla UNIX tabanlı Fedora Core 2 işletim sistemi çalıştıran bir makineye Nessus sunucusu kurulmuş ve yine aynı makine üzerindeki istemci ile iki hedef makine taranarak elde edilen raporlar html formatında saklanmıştır. İlk hedef, üzerinde Red Hat 9 işletim sistemi kurulu olan bir web ve mail sunumcusudur. Tarama sonucunda oluşturulan raporda açıklık olarak nitelendirilen 9, uyarı olarak nitelendirilen 33 madde yer almaktadır. Sistemde rastlanan 9 açıklık şu şekildedir:

- TCP port 23 üzerinde telnet servisinde Sendmail ile ilgili 2 farklı açıklık bulunmuştur.

- TCP port 22 üzerinde ssh servisinde OpenSSH ile ilgili 1 açıklık bulunmuştur.
- TCP port 139 yani netbios-ssn portu üzerinde 4 farklı açıklık bulunmuştur.
- TCP port 443 üzerinde HTTPS protokolü ile ilgili OpenSSL' den kaynaklanan 1 açıklık bulunmuştur.
- TCP port 993 üzerinde IMAPS servisinde yine OpenSSL ile ilgili 1 açıklık bulunmuştur.

Ortaya konan raporda her madde için ayrıntılı açıklamalar yapılmış ve sorunu gidermek için gereken önlemler sıralanmıştır. Şekil 2.1 'de raporun bir kısmı gösterilmiştir.

Informational	imap (143/tcp)	The remote imap server banner is : * OK [CAPABILITY IMAP4REV1 LOGIN-REFERRALS STARTTLS AUTH=LOGIN] www.bilmuh.gyte.edu.tr IMAP4rev1 2001.315rh at Tue, 8 Jun 2004 14:32:25 +0300 (EEST) Versions and types should be omitted where possible. Change the imap banner to something generic. Nessus ID : 11414
Vulnerability	netbios-ssn (139/tcp)	It was possible to log into the remote host using a NULL session. The concept of a NULL session is to provide a null username and a null password, which grants the user the 'guest' access To prevent null sessions, see MS KB Article Q143474 (NT 4.0) and Q246261 (Windows 2000). Note that this won't completely disable null sessions, but will prevent them from connecting to IPC\$ Please see http://msgsec.securepoint.com/cgi-bin/get/nessus-0204/50/1.html All the smb tests will be done as "f/whatever" in domain BLMH CVE : CAN-1999-0504, CAN-1999-0506, CVE-2000-0222, CAN-1999-0505, CAN-2002-1117 BID : 494, 990 Nessus ID : 10394
Vulnerability	netbios-ssn (139/tcp)	The remote Samba server, according to its version number, may be vulnerable to a remote buffer overflow when receiving specially crafted SMB fragment packets. An attacker needs to be able to access at least one share to exploit this flaw. Solution : upgrade to Samba 2.2.8 Risk factor : High CVE : CAN-2003-0085, CAN-2003-0086 BID : 7106, 7107 Other references : RHSA:RHSA-2003:095-03, SUSE:SUSE-SA:2003:016 Nessus ID : 11398
Vulnerability	netbios-ssn (139/tcp)	The following shares can be accessed using a NULL session : - IPC\$ - (readable?, writeable?)

Şekil 2.1. İlk konaktan elde edilen rapor örneği.

Test için kullanılan ikinci hedef ise Windows 2000 Server işletim sistemine sahip bir domain sunucusudur. Bu taramanın sonucunda ise 13 güvenlik açığı ve 39

güvenlik uyarısı tespit edilmiştir. Tespit edilen 13 açıklık SMTP (TCP/25), HTTP (TCP/80) (6 adet), MSRPC (TCP/135) (2 adet), Netbios-Ssn (TCP/139), LDAP (TCP/389) (2 adet), SNMP (UDP/161) portları üzerindedir.

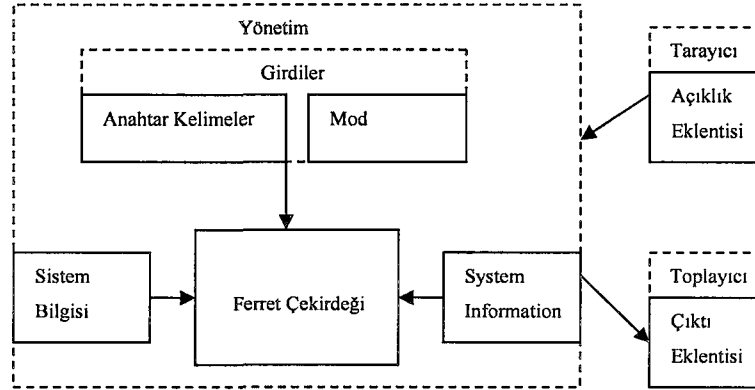
Nessus, sadece açıklıkları tarayıp bulmakla kalmaz aynı zamanda bulduğu açıklıkları isteğe göre istismar etmeye çalışarak sistemde açıklığın olduğunu kanıtlar. Yapılan taramalar sonucunda Nessus' un gerçekten de açık bulduğu portlar üzerinde çalışan servisler hakkında bilgi topladığı görülmüştür. Örneğin ikinci makinede UDP port 161 üzerinde çalışan SNMP servisindeki açıklıktan yararlanarak tüm Samba kullanıcıları listelenebilmiştir.

Öte yandan Nessus, Norton Internet Security programını çalıştıran başka bir konak üzerinde başarılı olamamıştır. Norton Internet Security, IDS özelliği sayesinde Nessus' u çalıştıran sunucu tarafından kendisine port tarama (portscan) atağı yapıldığını rapor etmiştir.

2.6.2 Ferret

Açıklık tarama teknolojisinin bir diğer örneği de açık kaynak kodlu ve konak tabanlı bir açıklık tarayıcı olan Ferret' tir. Perl programlama dili kullanılarak yazılmıştır. Ferret üç büyük parçadan oluşmuştur: Yönetim çekirdeği (management core), açıklık tarama eklentileri (vulnerability checking plug-ins) ve çıktı eklentileri (output plug-ins). Yönetim çekirdeğinin bazı görevleri şunlardır: Konfigürasyon dosyasını okumak, Ferret' in üzerinde çalıştığı sistem hakkında bilgi elde etmek, sistemde bulunan eklentileri belirlemek, seçilen eklentileri sırayla çalıştırmak ve elde edilen sonuçları gerekli yere iletmek [Sharma et al, 2004].

Ferret 'teki açıklık testleri nessus' ta olduğu gibi eklentiler şeklinde yazılmıştır. Linux, Solaris ve HP-UX işletim sistemlerinin açıklıklarını hedef alan 80 civarında eklenti geliştirilmiştir. Ferret' in yapısının genel görünümü Şekil 2.2' de gösterildiği gibidir.



Şekil 2.2. Ferret' in yapısı

Bir sistemi Ferret açıklık tarama aracıyla test ettiğimizde elde edeceğimiz rapor Şekil 2.3 'te gösterilmiştir.

```

Plugins finding Vulnerabilites are:
PLUGINNAME homedir-group
DESCRIPTION Checks the home directory field of the password file for vulnerabilities
RESULT Vulnerability Found
OUTPUT /var/spool/mail is group writable(mode 0775):: /var/gdm is group writable(mode 1770):

PLUGINNAME read-write-devs-etcfstab
DESCRIPTION Checks for world/group writability for resources in /etc/fstab file.
RESULT Vulnerability Found
OUTPUT /dev/shm is world writable (mode 1777):: /dev/shm is group writable (mode 1777):

PLUGINNAME set-umask-etc-profile
DESCRIPTION Checks the umask settings in /etc/profile for world writability
RESULT Vulnerability Found
OUTPUT None

Plugins not finding any vulnerability are:
PLUGINNAME bin-root-ownership
DESCRIPTION Checks whether root owns or not the critical /bin directory
RESULT No Vulnerability Found
OUTPUT None

PLUGINNAME check-access-exports
DESCRIPTION Checks for files in /exports that have been given some access control or not.
RESULT No Vulnerability Found
OUTPUT None

PLUGINNAME check-paths-cshrc
DESCRIPTION Checks for paths inside /.cshrc for world writability
RESULT No Vulnerability Found
OUTPUT File /.cshrc does not exist

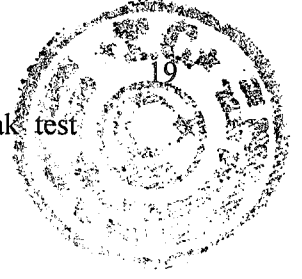
PLUGINNAME cshrc-root-ownership
DESCRIPTION Checks whether root owns or not the critical / cshrc file

```

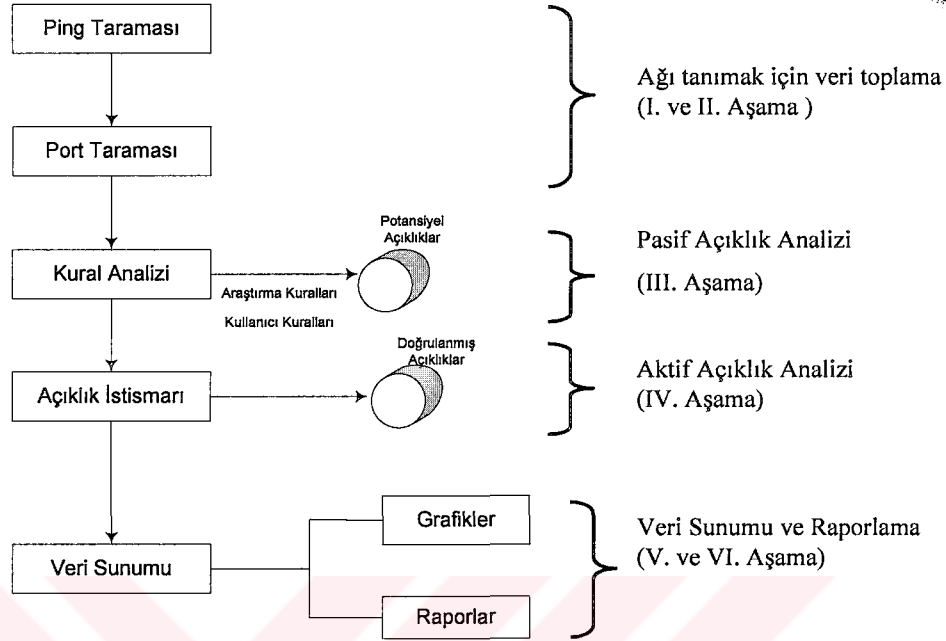
Şekil 2.3. Ferret ile elde edilen rapor örneği.

2.6.3 Cisco Secure Scanner

Cisco firmasının ürettiği bir güvenlik aracıdır. En büyük avantajı detaylı ve düzgün bir rapor sunmasıdır. Dezavantajı ise arka kapılar, truva atları, virüsler ve



yanlış konfigürasyonlar gibi sistem açıklıklarını fazla detaylı olarak test etmemesidir [Venter and Eloff, 2003].



Şekil 2.4. Cisco Secure Scanner' ın fonksiyonları

Cisco Secure Scanner'ın fonksiyonları 6 aşamada incelenebilir. Bu aşamalar Şekil 2.4' te gösterilmiştir. Birinci aşama ağın tanınmasıdır. Belirtilen IP adreslerinde ve portlarda taramalar yaparak ağın bir nevi haritası çıkarılır. Ağdaki iş istasyonlarını, sunumcuları, yönlendiricileri, switchleri, güvenlik duvarlarını vs. içeren bir liste çıkartılır. İkinci aşama veri toplama aşamasıdır. Listesi çıkarılan araçlardan bilgiler toplanır ve veritabanına kaydedilir. Port taraması ile aktif portlardan bilgiler toplanır. Üçüncü aşama analiz aşamasıdır. Bilinen açıklıkların tutulduğu veritabanı kullanılarak potansiyel açıklıklar analiz edilir.

2.6.4 Açıklık Tahmini (Vulnerability Forecasting)

Açıklık tarama teknolojisindeki bir diğer önemli konu da açıklık tahmini (vulnerability forecasting) konusudur. Bu teknoloji açıklık tarayıcılardan elde edilen sonuçları kullanır. Açıklık tahmininin anlamı bir sistemde yakın bir gelecekte olacak açıklıkları tahmin etmektir. Tahmin işlemi geçmiş açıklık verileri kullanılarak yapılır. Venter ve Eloff bu konu ile ilgili kavramsal bir model ortaya koymuştur

[venter and Eloff, 2004]. Açıklık tarayıcıdan elde edilen tarama sonuçları harmanlanmış açıklık kategorileri (HVC) ile eşleştirilir ve bu eşleştirmenin sonuçları HVC veritabanında saklanır. Daha sonra açıklık tahmin motoru (vulnerability scanning engine) HVC' ler bazında eğilimleri ve örüntüleri tahmin etmeye çalışır.



3 AKILLI AÇIKLIK TARAYICI MODELİ VE UYGULAMASI

Akıllı Açıklık Tarayıcı Modeli hem ağ tabanlı hem de konak tabanlı açıklık tarayıcıların avantajlarından faydalanmak, hem de her iki tip tarayıcının dezavantajlarını bertaraf etmek amacıyla tasarlanmıştır. Bu sebeple istemci/sunumcu mimarisinde bir model geliştirilmiş ve bu model Linux işletim sistemi üzerinde C++ dili kullanılarak gerçekleştirilmiştir. Tarama işleminin çeşitli safhalarında birçok açık kaynak kodlu program ve kütüphanelerden yararlanılmıştır.

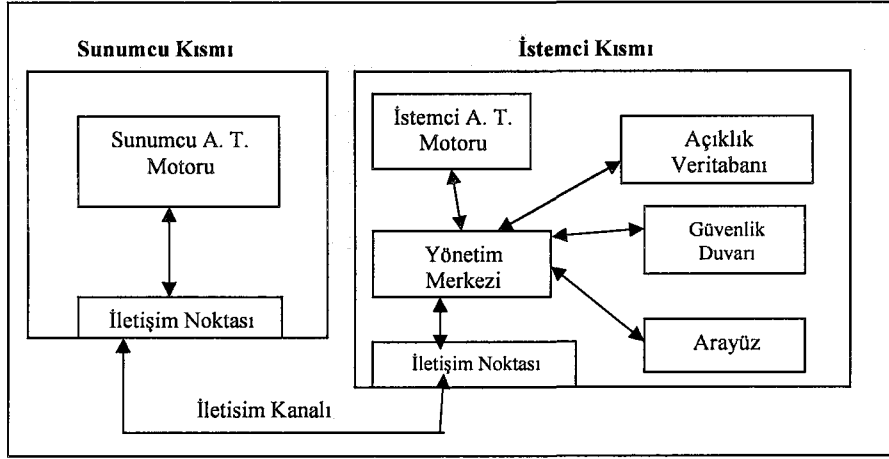
3.1 Modelin Genel Özellikleri

Akıllı Açıklık Tarayıcı Modeli istemci/sunumcu mimarisindedir. Bunun anlamı bir sunumcu ve bir veya birden fazla istemciden oluşabilir. Sunumcu kısmı tarama yapılacak hedef konağı dışarıdan bir saldırgan gibi görür. Temel olarak, hedef konağa belirli bir ağ trafiği göndererek hedef makine hakkında bilgiler elde etmeye ve daha sonra bu bilgileri kullanarak hedef konağın istismar edilebilecek açık noktalara sahip olup olmadığını bulmaya çalışır. Sunumcu kısmının ağ tabanlı açıklık tarayıcıları gibi çalıştığı söylenebilir.

Öte yandan, hedef konak üzerinde kurulan ve çalıştırılan istemci kısmı kritik sistem dosyalarına yanlış erişim hakkı verilmesi, kullanıcı hataları, zayıf şifreler vs. gibi sistem seviyesinde açıklıkları bulmaya çalışır. Bu nedenle istemci kısmı konak tabanlı açıklık tarayıcıların çalışmasına benzer. Ayrıca istemci kısmı taramanın başlatıldığı ve sonuçların görüntülediği kısımdır.

Modelin en önemli avantajlarından birisi, konak üzerinde bir program kurulacağından sistem seviyesinde erişim yetkisine sahip olunmasıdır. Böylece bulunan açıklıkların aktif olarak düzeltilmesi mümkün olabilmektedir. Bulunan açıklıklar rapor halinde kullanıcıya gösterildikten sonra kullanıcı eğer arzu ederse üzerinde açıklık bulunan portu derhal kapatabilecektir.

Akıllı Açıklık Tarayıcı Modelinin genel görünümü Şekil 3.1' te gösterilmiştir.



Şekil 3.1. Modelin genel görünümü.

3.2 İstemci Kısmı

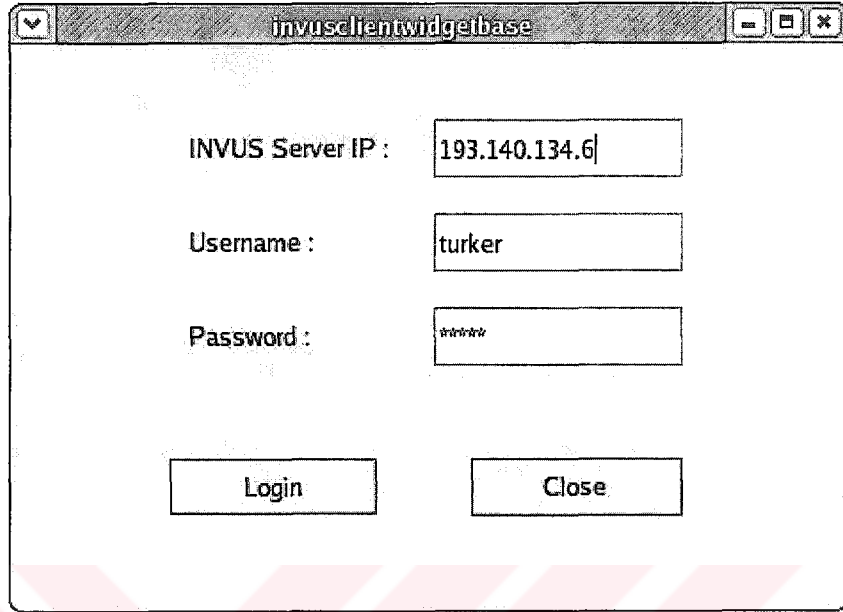
Modelin istemci kısmı taramanın başlatıldığı, yönetildiği ve sonuçların gösterildiği kısımdır. İstemci kısmı Yönetim Merkezi birimi ve bu birimin ilişkili olduğu istemci açıklık tarama motoru, ara yüz, açıklık veritabanı ve güvenlik duvarı birimlerden oluşur.

3.2.1 Yönetim merkezi

Yönetim merkezi birimi adından da anlaşılacağı gibi tarama işleminin yönetildiği birimdir. Tarama işlemi önce istemci kısmının sunumcu kısmına bağlanarak kendini tanıtmayı ve yetkilendirmeyi ile başlar. Bu amaçla istemci kısmındaki ara yüz vasıtasıyla bağlanmak istenilen sunumcunun IP adresi ile geçerli bir kullanıcı adı ve şifre verilir.

Verilen IP adresinde çalışmakta olan bir sunumcu var ise bir bağlantı oluşturulur. Bu bağlantı kullanılarak ara yüzde girilen kullanıcı adı ve şifre sunumcu kısmına gönderilir. Sunumcu bu bilgilerin doğruluğu kontrol eder. Verilen bilgiler doğru değil ise, sunumcu istemciye bir hata mesajı göndererek istenilen bilgilerin tekrar gönderilmesini ister. Eğer gönderilen bilgiler doğru ise, sunumcu bilgilerin doğruluğunu istemci kısmındaki yönetim merkezine teyit eder.

Sunumcu ile bağlantı kurma ve yetkilendirme işlemi için kullanılan ara yüz Şekil 3.2' te gösterilmiştir.

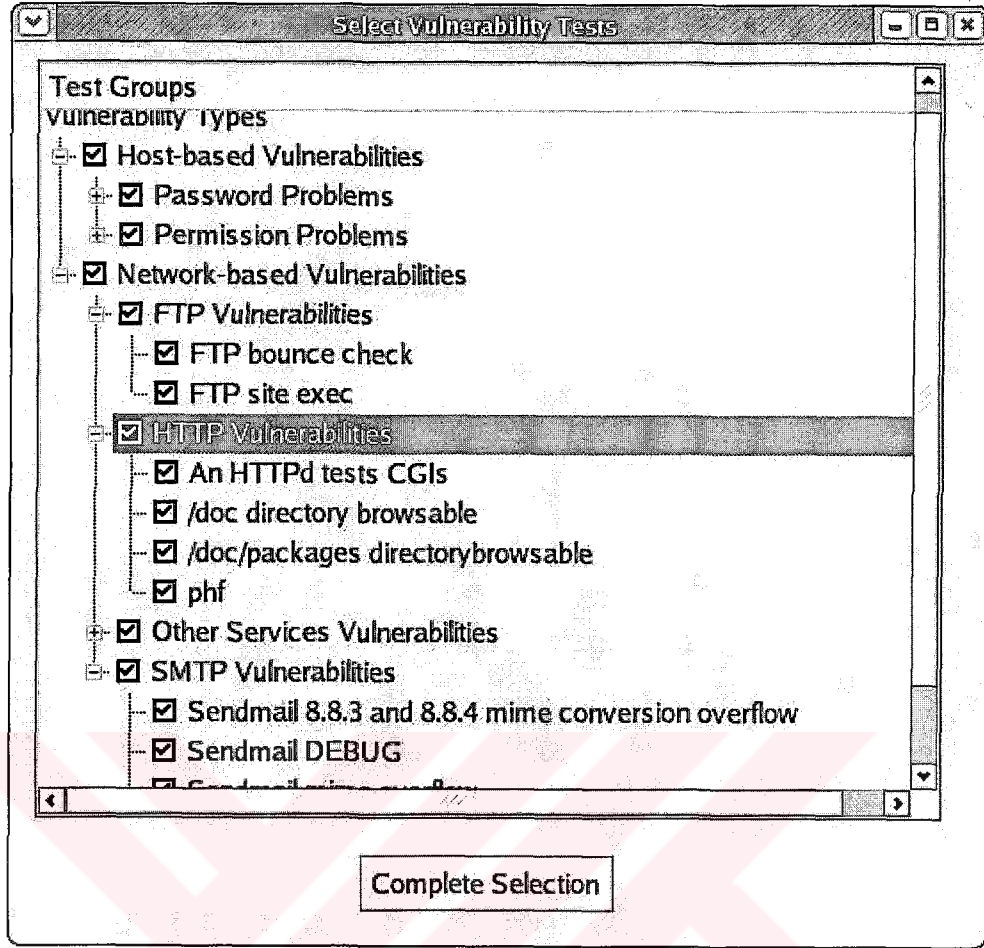


Şekil 3.2. Yetkilendirme ara yüzü.

Eğer yetkilendirme başarılı olursa, yönetim merkezi başka bir ara yüz vasıtasıyla taranacak açıklıkların seçilmesini bekler. Bu ara yüzde hem istemci açıklık tarama motoru hem de sunumcu açıklık tarama motoru tarafından taranabilecek bütün açıklıklar listelenir. Açıklıklar gruplar halinde listelenmiştir. Bu liste temel olarak iki gruba ayrılmıştır: Konak tabanlı açıklık testleri ve ağ tabanlı açıklık testleri.

Konak tabanlı açıklıklar istemci kısmındaki İstemci Açıklık Tarama Motoru tarafından taranan açıklıklardır. Bu kısımdan seçilen açıklık testleri yönetim merkezi tarafından İstemci Açıklık Tarama Motoruna iletilir. Ağ tabanlı açıklıklar ise sunumcu kısmındaki Sunumcu Açıklık Tarama Motoru tarafından taranacak açıklıklardır. Seçilen ağ tabanlı açıklık testleri daha önceden sağlanan bağlantı üzerinden sunumcu kısmına parametre olarak gönderilirler.

Açıklıkların seçimi için kullanılan ara yüz Şekil 3.3' te ayrıntılı olarak gösterilmiştir.



Şekil 3.3. Açıklıkların seçimi.

3.2.2 İstemci Açıklık Tarama Motoru

İstemci Açıklık Tarama Motoru konak tabanlı açıklık tarama işleminden sorumludur. Yönetim merkezi tarafından gönderilen konak tabanlı tarama ile ilgili parametreler doğrultusunda tarama işlemi gerçekleştirilir. Konak tabanlı açıklıklar şu şekilde gruplandırılabilir:

- Kritik sistem dosyalarının veya klasörlerinin erişim hakları ile ilgili açıklıklar,
- Şifre ayarları ile ilgili açıklıklar,
- Kullanıcıların home klasör ayarları ile ilgili açıklıklar,
- Konak üzerinde çalışan bazı servislerin ayarları ile ilgili açıklıklar.

İstemci Açıklık Tarama Motoru konak üzerinde çalıştığından sistem seviyesinde erişim hakkı vardır. Bunun anlamı sistem komutlarını sorunsuz bir şekilde çalıştırabilir. Konak tabanlı açıklıklar taranırken çeşitli sistem komutlarından yararlanılır. Ayrıca bazı açık kaynak kodlu yazılım ve kütüphanelerden de yararlanılmıştır.

İstemci Açıklık Tarama Motoru tarafından taranabilen on yedi adet konak tabanlı açıklık ve bu açıklıklar ile ilgili ayrıntılar şu şekildedir:

1- Kritik izinlerden biri olan /bin dizinin sahibinin root kullanıcısı olup olmadığının test edilmesi.

2- Kritik izinlerden biri olan /dev dizinin sahibinin root kullanıcısı olup olmadığının test edilmesi.

3- Kullanıcı şifrelerinin tutulduğu /etc/passwd dosyasının sahibinin root kullanıcısı olup olmadığının kontrol edilmesi.

4- Kullanıcı gruplarının tutulduğu /etc/group dosyasındaki grup isimlerinin sadece harf ve rakamlardan oluşan isimlere sahip olup olmadığının kontrol edilmesi.

5- /etc/group dosyasında boş satır olup olmadığının kontrol edilmesi. Sistem grup dosyası ile ilgili açıklıkları önlemek amacıyla grup dosyasında boş bir satır istemektedir.

6- /etc/group dosyasındaki grupların grup id' lerinin sadece rakamlardan oluşup oluşmadığının kontrol edilmesi.

7- /etc/group dosyasındaki şifre alanında * veya ! karakterlerinin olup olmadığının kontrol edilmesi.

8- /etc/passwd dosyasında yedi tane alanın olup olmadığının kontrol edilmesi.

9- /etc/passwd dosyasındaki kullanıcı adlarının sadece harf ve rakamlardan oluşan isimlere sahip olup olmadığının kontrol edilmesi.

10- /etc/passwd dosyasında boş şifre olup olmadığının kontrol edilmesi. Boş şifreler büyük bir güvenlik açığıdır, bu sebeple hiçbir kullanıcı boş şifre kullanmamalıdır.

11- /etc/passwd dosyasında boş satır olup olmadığının kontrol edilmesi. Sistem şifre dosyası ile ilgili açıklıkları önlemek amacıyla şifre dosyasında boş bir satır istemektedir.

12- /etc/passwd dosyasındaki şifre alanında * veya ! karakterlerinin olup olmadığını ve /etc/shadow dosyasının var olup olmadığının kontrol edilmesi.

13- Şifre dosyasındaki userid' lerin 8 karakterden uzun olup olmadığının kontrol edilmesi.

14- Şifre dosyasındaki userid' lerin sadece rakamlardan oluşup oluşmadığının kontrol edilmesi.

15- /etc/passwd dosyasının diğer kullanıcı gruplarına ait izinlerinin kontrol edilmesi. Bu dosyaya verilen haklarda bir yanlışlık olması yerel kullanıcıların sistemde gayri meşru haklar kazanmasına neden olabilir.

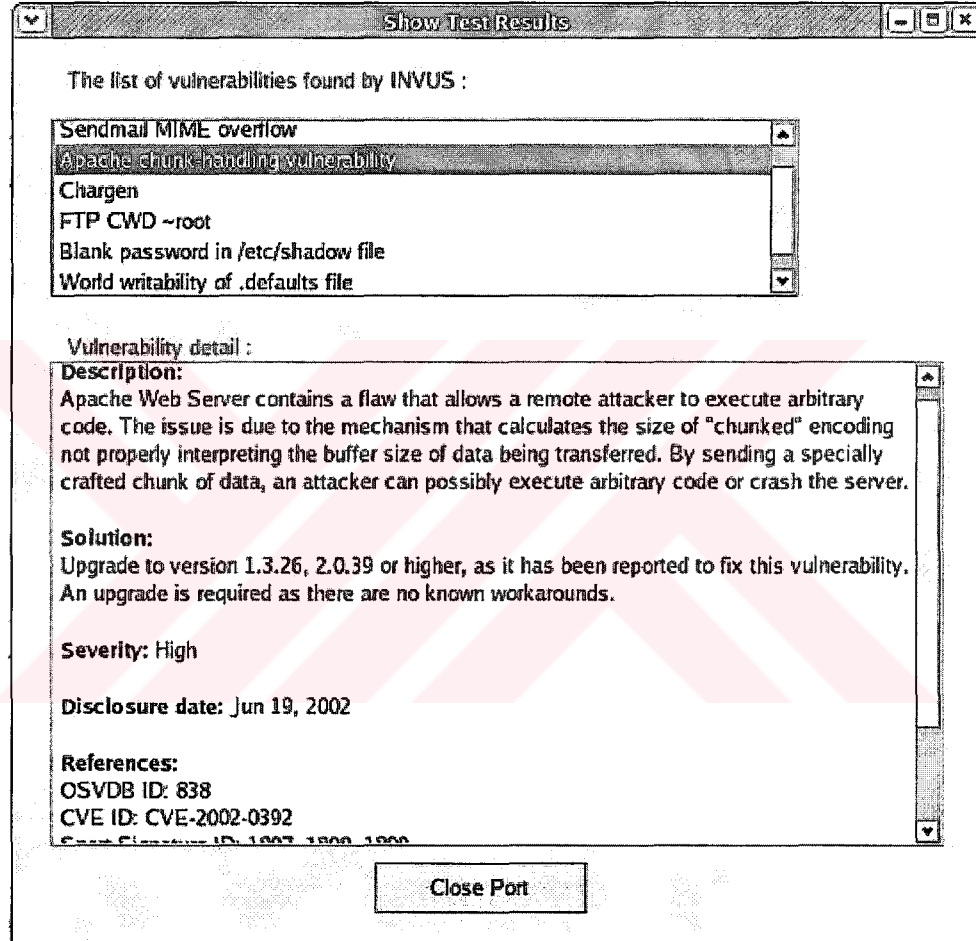
16- Kullanıcıların /home klasöründeki .defaults dosyalarının diğer kullanıcı gruplarına verilen haklarının kontrol edilmesi.

17- /etc/shadow dosyasında herhangi bir kullanıcının boş şifre kullanıp kullanmadığının kontrol edilmesi.

Konak üzerindeki tarama işlemi tamamlandıktan sonra elde edilen sonuçlar değerlendirilmek üzere yönetim merkezine gönderilir.

3.2.3 Raporlama

Tarama işleminin tamamlanmasının ardından hem Sunumcu Açıklık Tarama Motoru hem de İstemci Açıklık Tarama Motoru buldukları sonuçları yönetim merkezine gönderirler. Yönetim merkezi her iki motordan gelen sonuçlar doğrultusunda, açıklık veritabanından gereken bilgileri çekerek son raporu hazırlar. Son rapor örneği Şekil 3.4' te gösterilmiştir.



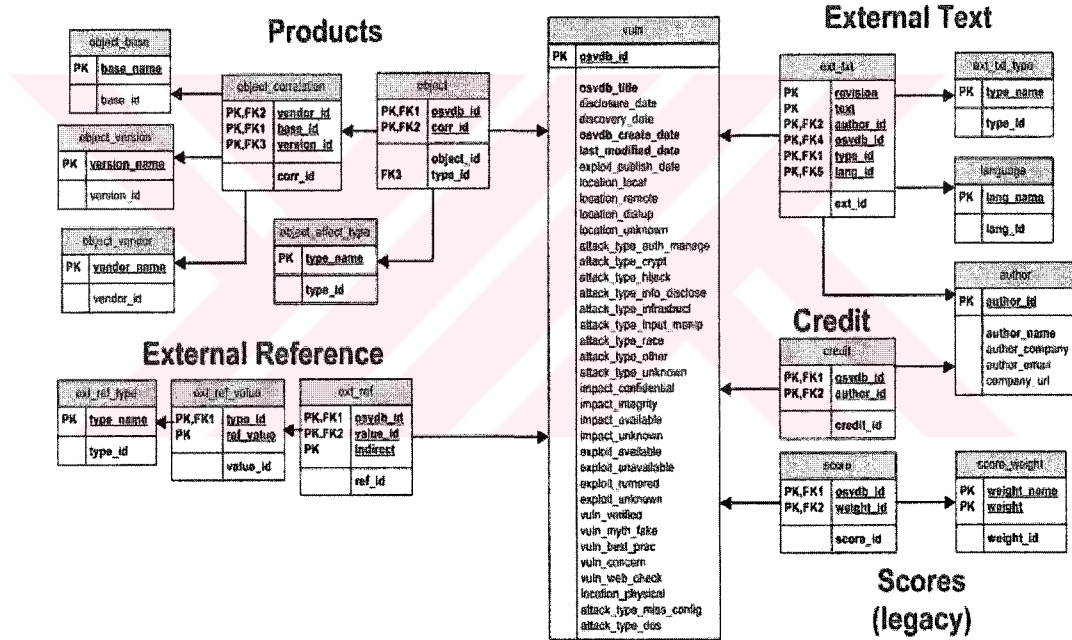
Şekil 3.4. Örnek rapor.

Elde edilen rapor istemci kısmındaki ara yüz vasıtasıyla kullanıcıya gösterilir. Aynı zamanda kullanıcıya gösterilen raporu HTML formatında saklama imkânı da sunulmuştur.

3.2.4 Açıklık Veritabanı

Açıklık tarama işleminin sonunda elde edilen ürün bulunan açıklıkların yer aldığı rapordur. Raporda gösterilen verilerin tutulduğu yer ise açıklık veritabanıdır. Uygulama yazılımında açıklık veritabanı olarak OSVDB (Open Source Vulnerability Database) veritabanı kullanılmıştır. OSVDB bilinen açıklıklar ile ilgili detaylı bilgiler içeren, herkesin kullanabileceği açık kaynak kodlu bir açıklık veritabanıdır. Açıklık tarama araçları geliştirilenler tarafından kolaylıkla kullanılabilir. CVE uyumlu bir veritabanı olması da önemli bir avantajdır.

Şekil 3.5’ te OSVDB açıklık veritabanının tabloları ve tablolar arasındaki ilişkiler gösterilmiştir.



Şekil 3.5. OSVDB veritabanının genel görünümü.

3.2.5 Açıklıkların Düzeltilmesi

Oluşturulan raporun gösterilmesi sırasında, bulunan bazı açıklıkların aktif olarak düzeltilmesi amacıyla kullanıcıya seçenek sunulmaktadır. Kullanıcı eğer dilerse ilgili butona basarak o açıklık ile ilgili portun tamamen kapanmasını sağlayabilecektir.

Bu işlem bir güvenlik duvarı yardımıyla gerçekleştirilmiştir. Eğer kullanıcı portun tamamen kapanmasını seçmişse güvenlik duvarına ilgili kurallar girilerek istenen port kapatılır. Linux işletim sistemi ile beraber gelen IPtables paketi ile bu işlemi yapmak mümkündür.

3.3 Sunumcu Kısım

Modelin sunumcu kısmı taramanın başlatıldığı, yönetildiği ve sonuçların gösterildiği kısımdır. Sunumcu kısmı devamlı çalışan bir program şeklindedir.

3.3.1 Sunumcu Açıklık Tarama Motoru

Sunumcu Açıklık Tarama Motoru ağ tabanlı açıklık tarama işleminden sorumludur. Taranacak açıklıklar istemci kısmındaki yönetim merkezi tarafından bağlantı üzerinden gönderilir. Ağ tabanlı açıklık tarama işlemi aşağıdaki safhalardan oluşur:

- Port tarama
- Servis tanıma
- Servislerin açıklıklarının taranması
- Bulunan açıklıkların kanıtlanması

Sunumcu Açıklık Tarama Motoru birçok açık kaynak kodlu yazılım ve kütüphane kullanmaktadır. Port tarama işlemi için Nmap programından yararlanılmıştır. Nmap açık kaynak kodlu, gelişmiş, popüler bir güvenlik sına aracıdır. Nmap port tarama işlemi için SYN (yarı-açık) tarama, FIN tarama, Xmas tarama vs. gibi TCP tarama tipleri sağlar. Ayrıca, yapılan taramanın zaman ayarları da ayarlanabilmektedir. Bu sayede nüfuz tespit sistemlerine yakalanmadan tarama yapmak mümkün olabilmektedir. Bir atak yeterince yavaşlatılabilirse günlük trafiğin içinde kaybolur ve yakalanma engellenebilir.

Portların açık olup olmadığının belirlenmesinden sonra açık portlar üzerinde çalışan servislerin belirlenmesi gerekir. Çoğu servisin hangi port üzerinde çalıştığı aşağı yukarı bellidir. Örneğin, http servisi 80. portta, FTP servisi 21. portta, SMTP

servisi 25. portta çalışmaktadır. Ancak bazen bu portlar değiştirilebilir. Bu durumda hangi servisin hangi port üzerinde çalıştığından emin olmamız gerekir.

Servislerin belirlenmesinden sonra bu servislerin hangi programlar tarafından kullanıldığı belirlenir. Örneğin SMTP servisinin varsayılan portu olan 25. port üzerinde Sendmail programının herhangi bir versiyonu çalışıyor olabilir. Sadece çalışan programın ve versiyonunun bilinmesi halinde bile birçok açıklığın var olup olmadığı tespit edilebilir. Port üzerinde çalışan program belirlendikten sonra bu program ile ilgili bilinen açıklıklar araştırılmaya başlanır. Bu araştırma her bir açıklık için farklı biçimde yapılır.

Sunumcu Açıklık Tarama Motoru 30 farklı açıklığı tarayabilmektedir. Bu açıklıklar ve bu açıklıklar ile ilgili yapılan testlerin ayrıntılı bilgileri şu şekildedir:

1- Sendmail 8.8.3 and 8.8.4 MIME conversion overflow (CVE-1999-0047): Sendmail sunumcusunun versiyon numarasına göre, yetkisiz birinin root olarak istediği komutu çalıştırabilmesine olanak sağlayan MIME dönüşümü tampon bellek taşması atağına izin vermesidir. Konak üzerinde çalışan sendmail programının versiyonu kontrol edilir. Eğer versiyon 8.8.3 veya 8.8.4 ise açıklık mevcuttur.

2- Sendmail DEBUG (CVE-1999-0095): Mail transfer yazılımının DEBUG modunu kabul etmesidir. Bu mod uzaktaki kullanıcıların yetkilendirmeye gerek duymadan root olarak istedikleri komutu çalıştırabilmelerine olanak sağladığı için oldukça tehlikelidir. Mail transfer yazılımının söz konusu moda izin verip vermediği kontrol edilir.

3- HELO overflow (CAN-1999-0098): SMTP sunumcusu uzaktaki kullanıcıların HELO komutuna çok uzun argümanlar sağlayarak isimsiz mail göndermelerine izin vermesidir. Mail sunumcusunun isimsiz mail gönderip göndermediği kontrol edilir.

4- Sendmail MIME overflow (CVE-1999-0206): Sendmail sunumcusunun versiyon numarasına göre, yetkisiz birinin root olarak istediği komutu çalıştırabilmesine olanak sağlayan MIME tampon bellek taşması atağına izin

vermesidir. Sunumcu üzerinde çalışan sendmail programının versiyonu kontrol edilir. Eğer versiyon 8.8.0 veya 8.8.1 ise açıklık mevcuttur.

5- Sendmail prescan() overflow (CAN-2003-0681): Sendmail sunumcusunun versiyon numarasına göre, uzaktaki kullanıcıların root yetkisi kazanmasına olanak sağlayan tampon bellek taşmasına karşı açık olmasıdır. Sendmail' in 5.79 ile 8.12.9 versiyonları arasındaki tüm versiyonlar bu açıklığı barındırır.

6- Sendmail buffer overflow due to type conversion (CAN-2003-0161): Sendmail sunumcusunun versiyon numarasına göre, uzaktaki kullanıcıların root yetkisi kazanmasına olanak sağlayan tampon bellek taşmasına karşı açık olmasıdır. Sendmail' in 5.79 ile 8.12.8 versiyonları arasındaki versiyonlar bu açıklığı barındırır.

7- Sendmail Parsing Redirection DOS (CVE-1999-0393): Sendmail sunumcusunun uzaktaki saldırganların geniş başlıklı mesajlar göndererek servisin inkarına sebep olmalarına olanak sağlamasıdır. Sendmail' in 8.9.3 versiyonundan önceki versiyonlarında bu açıklık görülebilir.

8- Sendmail Group Permissions Vulnerability (CVE-1999-0129): Sendmail sunumcusunun yerel kullanıcıların bir dosyaya yazma hakkı vermesi ve .forward veya :include: dosyaları vasıtasıyla grup hakkı kazanmalarına izin vermesidir. Sendmail' in 8.8.4 versiyonundan önceki versiyonlarında bu açıklık görülebilir.

9- Sendmail ETRN command DOS (CVE-1999-1109): Sendmail sunumcusunun versiyon numarasına bağlı olarak uzaktaki saldırganların bir dizi ETRN komutu göndererek servisin inkarına sebep olmalarına olanak sağlamasıdır. Sendmail' in 8.10.1 versiyonundan önceki versiyonlarında bu açıklık görülebilir.

10- Sendmail remote header buffer overflow (CVE-2001-1349): Sendmail sunumcusunun versiyon numarasına göre, uzaktaki kullanıcıların root yetkisi kazanmasına olanak sağlayan tampon bellek taşmasına karşı açık olmasıdır. Sendmail' in 5.79 ile 8.12.7 versiyonları arasındaki versiyonlar bu açıklığı barındırır.

11- phf (CVE-1999-067): phf isimli CGI programının kurulu olduğu anlamına gelir. Bu program, saldırganın root seviyesinde istediği komutu çalıştırmasına izin verir. /cgi-bin/phf ' in sistemde var olup olmadığı kontrol edilir.

12- /doc/packages directory browsable? (CVE-2000-1016): /doc/packages dizinin görüntülenebilir olmasıdır. Bu dizinin içeriği saldırganlara konak üzerinde kurulmuş paketlerin versiyonları hakkında bilgi verebilir.

13- /doc directory browsable? (CVE-1999-0678): /doc dizinin görüntülenebilir olmasıdır. Bu dizin /usr/doc dizinin içeriğini gösterir, bu sebeple konak üzerinde hangi programların kurulduğu görülebilir.

14- Apache < 1.3.29 (CAN-2003-0542): Web sunumcusunun Apache' nin 1.3.29 versiyonundan daha düşük bir versiyonunu kullanmasıdır. Kullanılan web sunumcu yazılımının belirtilen versiyondan daha düşük olup olmadığı kontrol edilir.

15- Apache < 2.0.51 (CVE-2004-0786): Konağın Apache2 nin 2.0.51 versiyonundan daha düşük bir versiyonu kullanmasıdır. Bu versiyonların servisin inkârına açık olduğu bilinmektedir. Kullanılan web sunumcu yazılımının belirtilen versiyondan daha düşük olup olmadığı kontrol edilir.

16- mod_rewrite in Apache 1.3.12 (CVE-2000-0913): Apache' nin 1.3.12 versiyonu ve bundan daha önceki versiyonlarında, eğer RewriteRule komutu içinde regular expression içeren dosya ismine genişletilirse uzaktaki saldırganların istedikleri dosyaları okuyabilmelerine izin verebilir.

17- Apache chunk-handling vulnerability (CVE-2002-0392): Apache web sunucusunun 1.3' ten 1.3.24' e ve 2.0' dan 2.0.36 versiyonları arasındaki versiyonlarında saldırganların servisin inkârı ve keyfi kod çalıştırmalarına izin veren bir açıklık mevcuttur.

18- PHP breaks safe mode (CVE-2001-1246): PHP 4.0.5 ve 4.0.10 arasındaki versiyonlar güvenli modda mail() fonksiyonundaki bazı parametreleri tam olarak

temizleyemedikleri için saldırganların shell vasıtasıyla uzaktan komut çalıştırmalarına yol açabilir.

19- PHP socket integer overflow (CVE-2003-0172): PHP' nin 4.3.2 versiyonundan daha düşük versiyonlarında, eğer kullanıcının belli fonksiyonlara parametre göndermesine izin veriyorsa kullanıcı servisi çökertebilir.

20- Finger (CVE-1999-0612): Finger servisinin herhangi bir versiyonunun çalışması ve kullanıcılar ile ilgili bilgileri ağ üzerinden açığa çıkarması.

21- Chargen (CVE-1999-0103): Chargen servisinin çalışır durumda olması saldırganların sunumcuya devamlı paketler göndererek meşgul etmesine olanak sağlayabilir.

22- Netstat (CAN-1999-0650): Netstat servisi sistemdeki aktif bağlantıları gösterdiği için, saldırganlara faydalı bilgiler sağlayabilir.

23- Systat (CVE-1999-0103): Systat servisi sistemde hangi proseslerin çalıştığını ve bunları hangi kullanıcıların çalıştırdığını gösterdiği için saldırganlara önemli bilgiler sağlayabilir.

24- FTP CWD ~root (CVE-1999-0082): FTP sunumcusundaki bir bug sayesinde herhangi biri CWD komutu ile root olarak sisteme girebilir. Söz konusu komutun başarılı olup olmadığı test edilir.

25- FTP Bounce Check (CVE-1999-0017): PORT komutunu kullanarak FTP sunumcusu üçüncü parti konaklara bağlanmaya zorlanabilir. Bu da saldırganın ağ kaynaklarını kullanarak diğer konakları taramalarına hatta güvenlik duvarını aşabilmeleri olanağı sağlar.

26- FTP site exec (CVE-1999-0080): FTP' deki 'site exec' problemini kullanarak konak üzerinde istenilen kod çalıştırabilir. Bu komut test edilerek bu açıklığın var olup olmadığı test edilir.

27- Linux FTP backdoor (CAN-1999-0452): Linux' un eski FTP deamonlarında uzaktaki kullanıcıların boş kullanıcı adı ve şifre girerek FTP üzerinden root hakkı kazanmalarını sağlayan bir arka kapı vardır.

28- Tftpd overflow (CVE-2002-0813): Tftp sunucusu çok büyük bir UDP datagramı aldığı zaman çökebilir. Bir saldırgan bu açıklığı kullanarak sunucuyu etkisiz hale getirebilir veya sistemde keyfi komut çalıştırabilir.

29- SSH overflow (CVE-1999-0834): SSH 'in versiyonu 1.2.27' den düşükse, uzaktaki saldırganın sistemde root hakkı kazanma ihtimali oldukça yüksektir.

30- rsh with null username (CVE-1999-0180): Eğer rsh doğru yapılandırılmamış ise in.rshd kullanıcıların boş kullanıcı adı ile sisteme girebilmelerine izin verebilir.

Sunumcu Açıklık Tarama Motoru tarama işlemini tamamlandıktan sonra tarama sonuçlarını içeren veriyi daha önce açılan iletişim kanalı yoluyla hedef konak üzerindeki yönetim merkezine gönderir.

4 DENEYSEL SONUÇLAR VE ANALİZ

Önerilen model ve uygulama yazılımı performans ve kullanılabilirlik bakımından test edilmiştir. Yapılan testler sonucunda önerilen modelin hız ve doğruluk bakımından çok iyi sonuçlar verdiği gözlenmiştir.

4.1 Deney Ortamı

Daha önce de anlatıldığı gibi önerilen model istemci/sunumcu mimarisine sahiptir. Yapılan testlerde sunumcu kısmının çalıştığı sistemin özellikleri şu şekildedir: Intel Pentium IV 3.06 Ghz işlemci, 1024 MB bellek ve Fedora Core 2 işletim sistemi. Yapılan testlerde programın istemci kısmı için ise 3 farklı makine kullanılmıştır. Bu makinelerin birincisi Intel Pentium IV 1.4 Ghz işlemci, 512 MB bellek ve Fedora Core 3 işletim sistemi; ikincisi Intel Pentium IV 3.06 Ghz işlemci, 512 MB bellek ve Red Hat 9 işletim sistemi diğeri ise Intel Celeron 1000 Mhz işlemci, 512 MB bellek ve Suse 9.2 işletim sistemine sahiptir. Sunumcu için kullanılan makine ve istemci için kullanılan makinelerin hepsi aynı 100 megabit' lik switche bağlıdır.

Gerek sunumcu program gerekse istemci programlar root seviyesinde erişim hakkına ihtiyaç duyduklarından, programlar root kullanıcısı tarafından oluşturulmuşlardır. Ayrıca sunumcu kısmında çalışan program tarafından kullanılan çeşitli açık kaynak kodlu program ve kütüphaneler de sunumcu makinesi üzerine kurulmuş ve gerekli ayarlamaları yapılmıştır.

4.2 Deneysel Sonuçlar

Uygulama yazılımı 30 adet ağ tabanlı ve 17 adet konak tabanlı açıklığı taramak üzere test edilmiştir. 30 adet ağ tabanlı açıklık testinin 10 tanesi SMTP açıklıkları, 9 tanesi SMTP açıklıkları, 5 tanesi FTP açıklıkları ve 6 tanesi de diğer servisler(finger, systat, netstat, chargen, ssh ve rsh) ile ilgili açıklıklardır. Hem ağ tabanlı hem de konak tabanlı açıklıkları test edebilmek için test edilen konaklar üzerinde çeşitli ayarlamalar yapılmıştır. Örneğin açıklık testinin pozitif cevap vermesi için konak

üzerinde çalışan test edilen servise ait yazılımın versiyonu veya gereken konfigürasyonları değiştirilmiştir.

Her 3 konak üzerinde yapılan çeşitli testler sonucunda Akıllı Açıklık Tarayıcı Modeli uygulama yazılımının bulması gereken tüm açıklıkları bulduğu ve rapor ettiği gözlenmiştir.

4.3 Diğer Açıklık Tarayıcılar ile Karşılaştırma

Önerilen modelin diğer bazı açıklık tarayıcı modelleri ile karşılaştırılması Tablo 4.1’ de sunulmuştur. Tablodan da görülebileceği gibi önerilen modelin en önemli özellikleri hem ağ hem de konak tabanlı açıklıkları bulabilmesi ve bulunan açıklıkları aktif olarak düzeltebilmesidir.

Tablo 4.1. Açıklık tarayıcıların karşılaştırılması.

Özellik	Nessus	Retina	Ferret	GFI LanGuard	Önerilen Model
İşletim Sistemi	UNIX	Windows	UNIX	Windows	UNIX
Konak tabanlı / Ağ Tabanlı?	Ağ tabanlı	Ağ tabanlı	Konak tabanlı	Ağ tabanlı	Her ikisinde
Aktif düzeltme	Yok	Yok	Yok	Yok	Var
Saldırı Simülasyonu	Var	Var	Yok	Yok	Var
Şifreli Haberleşme	Var	Yok	Yok	Yok	Yok
CVE Uyumu	Var	Var	Yok	Yok	Var
Grafik Arabirim	Var	Var	Yok	Var	Var
Protokol Analizi	Var	Var	Yok	Var	Var

Akıllı Açıklık Tarayıcı Modeli uygulaması çalışma zamanı bakımından da test edilmiştir. Her üç konak üzerinde yapılan testlerin sonucunda, 30 adet ağ tabanlı ve 17 adet konak tabanlı açıklığın test edilmesi için gereken ortalama zamanın yaklaşık 35 saniye olduğu görülmüştür. Ağ tabanlı bir açıklık tarayıcı olan Nessus ile geliştirilen uygulamanın taradığı 30 ağ tabanlı açıklığın test edilmesi yine aynı üç konak için ortalama 50 saniye dolayındadır. Nessus yazılımının aynı testi daha uzun

sürede yapmasının en önemli nedeni taramanın başında seçilen açıklıkların hangi port ile ilgili olduğuna bakmadan hedef konak üzerindeki tüm portları taramasıdır.



5 SONUÇ VE ÖNERİLER

Dünya üzerinde tek başına bir bilgisayar sistemini tam anlamıyla güvenli hale getirebilecek veya güvenli kalmasını sağlayabilecek herhangi bir araç bulunmamasına rağmen, saldırıya hedef olma riski bazı araçlar kullanılarak azaltılabilmektedir. Bu araçlardan biri de açıklık tarayıcılarıdır. Açıklık tarama işleminin en önemli amacı sistemde bulunan açıklıkların istismar edilmeden önce bulmak ve bu doğrultuda gereken önlemlerin alınmasını sağlamaktır. Bu sebeplerden dolayı açıklık tarayıcı araçlar sistem yöneticileri tarafından yaygın şekilde kullanılan güvenlik araçlarıdır.

Bu çalışmada açıklık tarayıcılar için yeni bir model önerilmiştir. Önerilen model bilgisayar sistemlerini hem konak hem de ağ tabanlı açıklıklar için tarayıp detaylı bir rapor hazırlayabilmektedir. Hazırlanan rapor bulunan açıklıklar ile ilgili detaylı bilgileri gösterdiği gibi bu açıklıkları düzeltmek amacıyla kullanılabilecek çözüm önerilerini de belirtir. Ayrıca bu raporları kullanarak sistem yöneticileri bulunan açıklıkları aktif olarak düzeltebilmektedir. Bu düzeltme işlemi güvenlik duvarı ile beraber çalışarak gerçekleştirilmektedir.

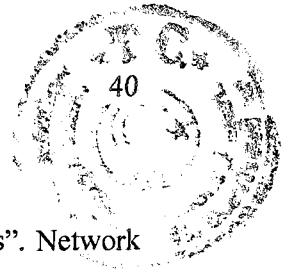
Önerilen model Linux işletim sistemi üzerinde KDevelop geliştirme ortamı üzerinde C++ programlama dili kullanılarak gerçekleştirilmiştir. Uygulamanın kullanımı kolay bir ara yüzü ve bulduğu açıklıklar hakkında ayrıntılı rapor verebilen raporlama sistemi mevcuttur.

Akıllı Açıklık Tarayıcı Modeli uygulama yazılımının taradığı açıklık sayısı 47 ile sınırlıdır. Gelecekte yapılacak olan çalışmalarda uygulama yazılımının taradığı açıklıkların sayısı artırılabilir. Mevcut uygulamada sunumcu ile istemciler arasındaki iletişim şifrelenmemiş metin şeklinde yapılmaktadır. Aradaki iletişimi şifrelemek modelin gelişimine katkıda bulunabilir. Önerilen modelde, bulunan açıklıkları düzeltmek için güvenlik duvarı kullanılmaktadır. Güvenlik duvarı ile açıklığın bulunduğu port tamamen kapatılabilir ancak bu her zaman istenen bir durum olmayabilir. İstemci kısmında bulunan açıklıkların düzeltilmesi için hazırlanan raporda kullanıcıya daha farklı seçenekler de sunulabilir.

Önerilen modeli açıklık tahmini konsepti ile birleştirmek de gelecekte yapılabilecek çalışmalardan biri olabilir. Açıklık tarama programından elde edilecek veriler açıklık tahmin aracı tarafından girdi olarak kullanılabilir.



KAYNAKLAR



- [1] H.S. Venter, J.H.P. Eloff, "Assessment of Vulnerability Scanners". Network Security, February 2003, volume 2003, issue 2, pp 11-16.
- [2] J.W. Humphries, C.A. Carver, U.W. Pooch, "Secure Mobile Agents for Network Vulnerability Scanning". IEEE Workshop on Information Assurance and Security, 2000, pp 19-25.
- [3] A. Sharma, J.R. Martin, N. Anand, M. Cukier and W.H. Sanders, "Ferret: A Host Vulnerability Checking Tool" Proceedings of the 10th IEEE Pacific Rim International Symposium on Dependable Computing, March 2004 pp 389-394.
- [4] H.S. Venter, J.H.P. Eloff, "Vulnerability forecasting- a conceptual model", Computers & Security, September 2004, volume 23, issue 6, pp 489-497.
- [5] Robert A. Martin, "Managing Vulnerabilities in Networked Systems", IEEE Computer, November 2001, volume 34, no 11, pp 32-38.
- [6] Symantec Corporation, "Symantec Internet Threat Report VII", March 2005.
- [7] H.T. Tian, L.S. Huang, Z. Zhou, Y.L. Luo, "Arm up Administrators: Automated Vulnerability Management" IEEE International Symposium on Parallel Architectures, Algorithms and Networks, May 2004, pp 587-593.
- [8] S.C. Lee, L.B. Davis, "Learning from Experience: Operating System Vulnerability Trends", IEEE IT Pro, January 2003, volume 5, no 1, pp 17-24.
- [9] E.E. Schultz, "The MSBlaster worm: going from bad to worse", Network Security, October 2003, volume 2003, issue 10, pp 4-8.
- [10] Sunbelt Software, "Vulnerability Assessment Scanning", February 2004.
- [11] R. Guirguis, "Network- and Host-Based Vulnerability Assessments: An Introduction to a Cost Effective and Easy to Use Strategy", SANS InfoSec Reading Room, June 2003.
- [12] I. Krsul, "Software Vulnerability Analysis", Purdue University, West Lafayette, IN, 1998.



- [13] Chris McNab, "Network Security Assessment", ISBN 0-596-00618-1, O'Reilly, 2004.
- [14] S. Mollerhaug, "Useful Vulnerability Assessment", Information Security Technical Report, April 2003, volume 8, no 4, pp 78-84.
- [15] Symantec Corporation, "Progressive Scanning Technology White Paper", July 2001.
- [16] J. Mitchell, "Proactive Vulnerability Assessments with Nessus", SANS InfoSec Reading Room, April 2002.
- [17] E. Wales, "Vulnerability Assessment Tools", Network Security, July 2003, volume 2003, pp 15-17.
- [18] J. Forristal, G. Shipley, "Vulnerability Assessment Scanners", Network Computing, 2001.
- [19] CERT/CC. Cert/cc statistics 1988-2005. <http://www.cert.org/stats/>, 1988-2005.

ÖZGEÇMİŞ



1980 yılında Kırklareli’nde doğdu. İlk ve orta öğretimini 1986-1998 yılları arasında sırasıyla Kocahıdır İlkokulu ve Kırklareli Anadolu Lisesi’ nde tamamladı. 1998 yılında girdiği Marmara Üniversitesi Bilgisayar Mühendisliği Bölümü’nden 2002 yılında mezun oldu. Aynı yıl Gebze Yüksek Teknoloji Enstitüsü Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü’nde yüksek lisans eğitime başladı. Halen Gebze Yüksek Teknoloji Enstitüsü Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü’nde Araştırma Görevlisi olarak çalışmaktadır.

