Security Threat Prediction in a Local Area Network Using Statistical Model

Somak Bhattacharya, S.K. Ghosh
School of Information Technology
Indian Institute of Technology, Kharagpur
West Bengal 721302, India
somakb@sit.iitkgp.ernet.in, skg@iitkgp.ac.in

Abstract

In today's large and complex network scenario vulnerability scanners play a major role from security perspective by proactively identifying the known security problems or vulnerabilities that exists across a typical organizational network. Identifying vulnerabilities before they can be exploited by malicious user often helps to test, maintain, and assess the risk of the existing network. Still there are many problems with currently available state of the art vulnerability scanners like hampering system resource. One possible solution to this problem might be reducing the number of vulnerability scans, along with the quantitative approach towards different vulnerability category in order to identify which class of vulnerability should enjoy preference in the risk mitigation procedure. This paper introduces a model that predicts vulnerabilities that will occur in near future on a Local Area Network (LAN) by using statistical measures and vulnerability history data. Two case studies have also been presented to validate the model.

1. Introduction

An organizational Local Area Network (LAN), provide different services over the network, namely, FTP, HTTP. These services and applications contribute to a significant level of vulnerabilities towards the system. Vulnerability scanners are used to identify those vulnerabilities.

At present there are several vulnerability scanners available both as commercial and freeware, like, Nessus[1], SAINT[2], ISS[3] etc. In this work, Nessus security scanner has been used since it is a freeware and its client–server architecture also provides more operational flexibility from end user perspective. Though these vulnerability scanners are able to identify the potential flaws in the system, most of them suffering from some common problems like disruption of services.

The vulnerability prediction (VP) may be defined as an attempt to identify potential vulnerable areas on hosts across a network and the extent to which such areas on hosts will be vulnerable over a specific period of time in the near future. In this paper the principal aim of VP is, therefore, to predict the number of known vulnerabilities that could occur and may be a ranking of these vulnerabilities depending on their severity and impact which could lead to an efficient risk management procedure as well as eradication of repeated scans.

One of the significant works in this area has been carried out using fuzzy expected interval (FEI) [1]. A trend analysis of vulnerabilities [2] has been done by aggregating information on the vulnerabilities from publicly available sources, such as, ICAT[4], Bugtraq[5] and CVE[6].

A trend Analysis of exploitations [3] has been conducted on empirical study on number of computer security exploits and determined the rates at which incidents involving the exploit are reported to (CERT[7]) can be modeled using a common mathematical framework $C = I + S * M^{1/2}$.

Alhazmi et al. [4] fitted discovered vulnerabilities of Windows-98 and WindowsNT-4 in an "S" curve to simply predict future values. Arbach et al. [5] have examined several systems using the incidents and vulnerability data reported by CERT. Venter et.al. [6] had done a comprehensive study on different vulnerability scanners.

Alhazmi et.al [7] has defined "vulnerability density" as the number of vulnerabilities present in unit size of code. The vulnerability density of the previous versions of software is used for prediction of the number of vulnerabilities in a future release of software system.

Rescorla [8] has examined vulnerability discovery rates to determine the impact of vulnerability disclosures. Anderson [9] has proposed a model for a vulnerability-finding rate using a thermodynamics analogy. Tim

[1] http://www.nessus.org/
[2] http://www.iss.net/
[3] http://www.saintcorporation.com/
[4] http://www.amc.army.mil/amc/ci/matrix/other_documents/icat_new.htm.
[5] http://www.ntbugtraq.com/
[6] http://cve.mitre.org/
[7] Computer Emergency Response Team http:// www.cert.org

1-4244-0910-1/07/$20.00 ©2007 IEEE

Shimeall et.al [10] proposes a framework for conduct of information security trend analyses using the incidents reports to CERT. But most of these works are related to somewhat known facts or reported incidents but predicting total number of vulnerabilities of a LAN varies from organization to organization thus making it dynamic in nature. There is no good way to fit it in any predefined curve for this type of data and the parameters of the model highly depend on the available data feed into the model as input.

The Honeynet project provides [13] a white paper presenting statistical results on malicious activity used to predict future attacks.

The rest of the paper is organized as follows. Section 2 dealt with the vulnerability prediction. Section 3 will give details about our proposed model for vulnerability prediction. Finally we conclude in section 4.

2. Vulnerability prediction

In this paper, a statistical approach has been proposed to predict the total number of vulnerability that will generate in near future on a Local Area Network (LAN) in a timely manner.

The proposed statistical model has two main components, namely, data collector and data analyzer. The data collection is done through vulnerability scanner and the data is analyzed through quantitative forecasting of time series analysis using univariate model. The open source Nessus vulnerability scanner acts as a data collector component in this model and it runs on client-server architecture.

Impact	Total	Percentage
Execution of arbitrary code via network	4116	63%
User access via network	2997	46%
Denial of service via network	2509	38%
Disclosure of user information	2089	32%
Modification of user Information	1432	22%
Disclosure of system information	1430	22%
Disclosure of authentication information	1401	21%
Execution of arbitrary code via local system	1284	20%
Root access via local system	1184	18%
Root access via network	1177	18%
Modification of system information	848	13%
User access via local system	757	12%
Denial of service via local system	452	7%
Host/ resource access via network	404	8%
Modification of authentication information	70	1%

Figure 1. Impact of Different Vulnerabilities Exploits by the Attacker

A time series is a series where data is taken at successive times, spaced apart at uniform time intervals. A time series quantitative forecasting using univariate model is a statistical technique that analyzes the historical data in an attempt to identify a data pattern and assuming it will continue in the future and also which can be extrapolated in order to produce forecasts through identification of trend and other components of a time series [11].

In this paper, it has been tried to identify the total number of remotely exploitable vulnerabilities with unauthorized access that will be detected in near future in our network. The statistics (refer to Figure 1)[8] shows that unauthorized access via network (user access and root/administrator access) jointly leads (46%+18%=64%) to a devastating impact towards the organization. Categorizing vulnerabilities according to their type, impact, and exploit range (local/remote), definitely helps the security administrator to pick up the most critical class of vulnerability during the risk mitigation process. These vulnerabilities may belong to an application level, hardware level or operating system level which in turn determines their severity. For example, a denial-of-service (DoS) attacks posing a much more risk than that of exploiting hardware level vulnerability. Maintaining security and integrity of a large network is a complex and laborious task for the administrator as he needs to detect and close all possible paths for intrusion which is next to impossible whereas malicious user needs only a single path to reach his goal. This leads to prioritize or ranking the vulnerability categories depending upon their impact towards organization and the output of the proposed model. It is admitted that virus and worms attack almost all the networks over the internet but our approach is more oriented to direct attack where a determined attacker target a specific organizational network and his strategy solely depends on the kind of services and the OS is used by the organization and there existing security loopholes. So a general security advisories issued by CERT might not always be at per with individual network's risk.

In this paper, the total number of remotely exploitable vulnerabilities which gives unauthorized access (user or administrator / root access) has been taken into account that present in all the hosts in the LAN, using Nessus reports, on weekly basis. From this information the prediction has been made regarding the total number of remotely exploitable vulnerabilities that will be detected in near future using statistical analysis. The remotely exploitable vulnerabilities generally come from services and application software due to their design, implementation, or configuration error. For example, when an anonymous ftp server's home directory is writable, a remote intruder can creates a *.rhost* file in the ftp home directory. Using this remote login trust relationship between two machines, the intruder logs in from his machine to victim machine, getting a user shell without supplying a password. This operation is usually a legitimate action performed by regular user, but from the intruders' point of view, it is an attack. If the attacker has acquired a user shell on the target machine, he can exploit

[8] www.nccaiim.org/Education/Proceedings/2004/7-Moore-vulnerabilities.ppt.

buffer overflow vulnerability on a *setuid* root file to gain root access. This leads to complete compromise of the system by the remote attacker due to the configuration error of ftp service allowing anonymous user write permission on ftp home directory.

During the scanning procedure the Nessus plug-in database has been regularly updated so that the input data of the time-series model also include all the newly discovered (zero-day) vulnerabilities causing an upward trend in the vulnerability graph. At the same time available patches also has been deployed time-to-time which essentially decreases the total number of vulnerabilities causing a fall in the graph. These help to form a time series with vulnerability count as dependent variable and time itself as an independent variable.

Earlier approach like using Fuzzy Logic [1] states that "It is expected that a range of between x & y network and system information gathering vulnerabilities will be detected when the next scan is conducted". This approach has one serious shortcoming - "When the next scan is conducted" this phase is quite ambiguous. The next scan may be conducted in the very next day or after a week or after a month.

The Data Collector component (Nessus) is also suffering from some problems, namely,

- Lagging Updates: New vulnerabilities are found everyday. So it needs to update the database and perform the scan on regular basis.
- Consumes a large amount of Network resource making it very slow.
- Contains some plug-ins while running causes the target system may crash like denial-of-service (DoS) vulnerability checking plugin. On the contrary allowing "safe checks" increases the number of false positives.
- Produces a mammoth report, interpreting and analyzing them manually is a tedious job for administrator.

All these motivated us to use a time-based approach for vulnerability prediction as well as reduce the number of vulnerability scans to be performed on our network.

3. Proposed model for vulnerability prediction

The Box-Jenkins ARIMA (Auto-regressive integrated moving average) [11] model (refer to equation (1)) is used as a data analyzer component to analyze the data collected by data collector component. The model consists of the following phases:

- Model Identification
- Parameter Estimation
- Prediction

The ARIMA model takes a stationary time series as its input i.e., the statistical properties (mean and variance) of the time series are essentially constant over time. But according to CERT the vulnerability data is non-stationary. Taking the natural logarithm value or differencing between two consecutive values of the original time series are quite a few ways to make a series stationary one.

The autoregressive – moving average model (ARIMA) of order (p, q) can be represented as follows [11] [12].

$$z_t = \delta + \phi_1 z_{t-1} + \phi_2 z_{t-2} + ... + \phi_n z_{t-n} + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - .... - \theta_n a_{t-n} \quad (1)$$

where,

δ = Constant term
ϕ = autoregressive parameters
θ = moving average parameters
p = number of autoregressive terms in model
q = number of moving average terms in model

In equation (1), the current time series values are expressed as function of past time series values z_{t-1}, z_{t-2} etc and past random shocks a_{t-1}, a_{t-2} etc. The random shocks are difference between predicted value and observed value at time *t-1, t-2* respectively.

The autocorrelation function (ACF) and partial autocorrelation function (PACF) can be used of the observed time series data in order to tentatively identify stationary behavior of the time series and determine the order (values of p and q) of the model. The calculation of the ACF and PACF using the equation (2), (3) and (4) are mentioned below.

The autocorrelation at lag *k* is

$$r_k = \frac{\sum_{t=b}^{n-k} (z_t - z_m)(z_{t+k} - z_m)}{\sum_{t=b}^{n} (z_t - z_m)^2} \quad (2)$$

Where $z_m = \frac{\sum_{b}^{n} z_t}{(n - b + 1)}$

b is the differentiation transformation index.

The standard error of r_k is

$$s_{r_k} = \frac{(1 + 2\sum_{i=1}^{k-1} r_i^2)}{\sqrt{(n - b + 1)}} \quad (3)$$

The t-statistics at lag *k* is

$$t_{r_k} = r_k / s_{r_k} \quad (4)$$

Autocorrelation function is a listing or graph of the sample autocorrelations at lag k = 1, 2…n. A spike can exist in an autocorrelation function if absolute value of $|t_{r_k}|$ is greater than 1.6 for lags k =1, 2, 3 and $|t_{r_k}| > 2$ for all lags k>3 [11].

The partial autocorrelation can also be calculated as follows.

The partial autocorrelation at lag k is

$r_{kk} = r_1$ If k=1

$$r_{kk} = \frac{r_k - \sum_{j=1}^{k-1} r_{k-1,j} r_{k-j}}{1 - \sum_{j=1}^{k-1} r_{k-1,j} r_j}$$

If k = 2, 3 … n, where $r_{kj} = r_{k-1,j} - r_{kk} r_{k-1,k-j}$

For j = 1, 2…n (5)

The standard error of r_{kk} is

$$S_{r_{kk}} = \frac{1}{\sqrt{(n - b + 1)}} \quad (6)$$

The t-statistics at lag k is

$$t_{r_{kk}} = \frac{r_{kk}}{S_{r_{kk}}} \quad (7)$$

The partial autocorrelation function is a listing or graph of the partial autocorrelations at lag k =1, 2… n. A spike at lag k exists in PACF if $|t_{r_{kk}}| > 2$.

Both ACF and PACF can be cut–off after lag k if there are no spikes after lag k and dies down if deceases in a steady fashion rather than cut–off. Two case studies have been presented for the support of our model's effectiveness. It is very obvious that the vulnerability reports of an organization LAN are highly confidential. So getting real-time vulnerability data is impossible. Also this data varies from organization to organization to a great extent. So the parameters of our model will vary accordingly the data has been feed to the model as input. Case Study 1 data has been collected from [1]. Case study 2 used the data obtained from our local network.

3.1. Case Study 1

In this case study the data has been collected from a previous research paper [1]. They have given the total number of *network and system information gathering* vulnerabilities detected in their network for the last 10 days. The collected data is shown in Table 1. The time series model has taken 9 days value as its input and tried to predict the 10th day's value.

It can be observed from the Figure 2 that the number of vulnerabilities varies within a time period (here it is 'daily') significantly making it a non-stationary time series. That is because of new applications, services might be installed during that time period with a large number of vulnerabilities related to *network and system information gathering* or new vulnerabilities has been reported of the existing application and services causing an increase in the total number of vulnerabilities. The significant decrease in the vulnerability might cause due to the deployment of patches related to this particular vulnerability. The collected data is then analyzed using data analyzer component in following subsections.

Table 1. Collected Data of network and system information gathering Vulnerability

Time (Days)	Vulnerability Count (VC)
1	19
2	21
3	20
4	71
5	69
6	78
7	24
8	25
9	75
10	79

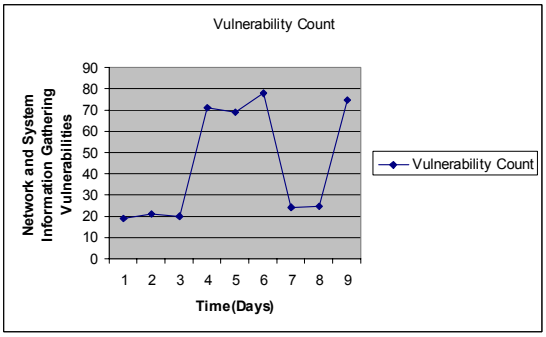

Figure 2. Plot of Original Time Series Data

Table 2. Converted Collected Data of network and system information gathering Vulnerability

Time (Days)	VC	$Log_{10}(VC)$	$Log_{10}(VC_{i+1}) - Log_{10}(VC_i)$
1	19	1.278754	-
2	21	1.322219	0.043466
3	20	1.30103	-0.02119
4	71	1.851258	0.550228
5	69	1.278754	-0.01241
6	78	1.838849	0.043466
7	24	1.892095	0.053246
8	25	1.380211	-0.51188
9	75	1.39794	0.017729
10	Predicted Value	Predicted Value	Predicted Value

3.1.1. Satisfying stationary condition. Plotting the data against time scale in Figure 2 clearly shows the series is a non-stationary one. The stationary time series is obtained by taking the logarithm of the vulnerability count (VC) and considering the difference of the consecutive values [11]. The Table 2 gives the stationary time series (refer to Figure 3). Figure 4 represents the ACF plot for working time series using the formulas (refer to equations (2), (3) and (4)). This conforms to the property of stationary time series [11]. The PACF (refer to equations (5), (6) and (7))

for the data has also been calculated and shown in Figure 5. The PACF will be required for subsequent calculation.

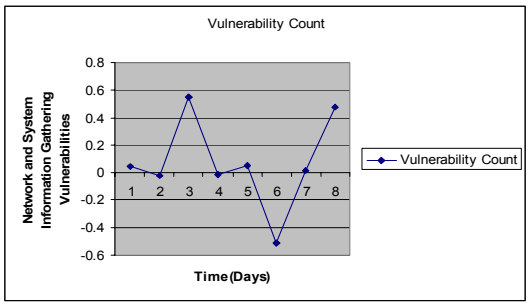

Figure 3. Plot of Converted Time Series Data

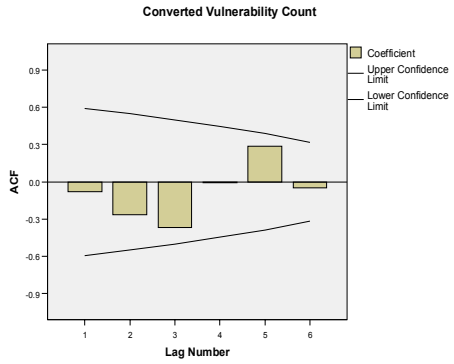

Figure 4. Autocorrelation Function Plot (ACF) with 95% Prediction Interval

3.1.2. Model Identification. After examining the ACF and PACF (Figures 4 and 5) behavior of the working time series, it can be observed that neither ACF nor PACF has spikes at any lag (or cuts off from lag 0). But in ACF lag 3 and 5 is closed to a spike (t-statistics value showed they are closed to 1.6 and 2 respectively). This leads to identify the model as a moving average model of order 1 [11].

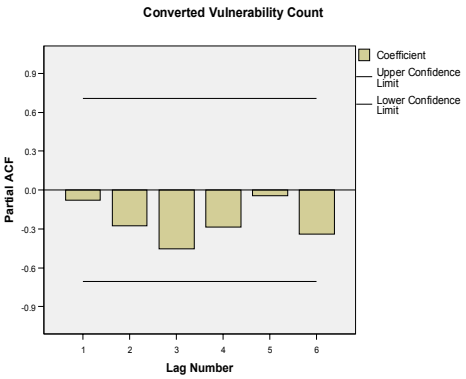

Figure 5. Partial Autocorrelation Function Plot (PACF) with 95% Prediction Interval

The significance constant term (refer to equation (1)) in the proposed model needs to be analyzed. The constant term,δ can be given as

$$\delta = \mu * (1 - \phi_1 - \phi_2 - .... - \phi_9) \quad (8)$$

For this particular case δ = μ as no autoregressive operator has been identified. The sample mean of the stationary time can be given as

$$z_m = \frac{\sum_{t=b}^{n} z_t}{(n - b + 1)} \quad (9)$$

The constant term in the model can only be included if the absolute value of

$\frac{z_m * \sqrt{(n - b + 1)}}{s_z}$ is greater than 2 [4].

The term, s_z, is given by

$$s_z = \sqrt{\frac{\sum_{t=b}^{n} (z_t - z_m)^2}{(n - b + 1) - 1}} \quad (10)$$

Calculating the value of $z_m * \sqrt{(n - b + 1)} / s_z$ shows it is greater than 2, naturally the constant term 0.07453845 has been included in the model. The final working model can be represented as (11)

$$z_t = \delta + a_t - \theta_1 a_{t-1} \quad (11)$$

3.1.3. Parameter Estimation. Here, XLminer 3_0[9] and SPSS 14.0[10] software tools have been used in order to estimate the values of θ_1, θ_2 (moving average parameters) etc. The resulting time series model is shown in equation (12).

$$z_t = 0.07453845 + a_t - 0.75636405\ a_{t-1} \quad (12)$$

3.1.4. Prediction. To predict the total number of remotely exploitable vulnerabilities that will be detected on 10^{th} day the equation (12) used as follows.

$$z_{10} = 0.07453845 + a_{10} - 0.75636405\ a_9 \quad (13)$$

Where, a_{10} is the random shock or the difference between observed and predicted value on 10^{th} day.

Putting all these data in equation (13) the predicted value for 10^{th} day will be -0.054099. Converting the value -0.054099 as $V = \text{Log}_{10}(VC_{10}) = \text{Log}_{10}(VC_9) + (-0.054099)$ and $VC_{10} = 10^V$ the predicted value for 10^{th} day will be $66.2 \approx 66$.

The measured value of the *network and system information gathering* on 10^{th} day was 79, and the predicted value is 66. The detailed analyses of the measured and predicted values are shown in Figure 6. It can be shown as this time based model highly depends on past values so the prediction can be more accurate if more data's (at least for past 30 - 40 time periods) [11] can be

[9] SPSS Evaluation Copy, Web: http://www.spss.com/

[10] XLMiner Evaluation version, Web: http://www.resample.com/xlminer/

incorporated into the model which essentially helps to identify the trends or patterns of vulnerability discovery over a LAN. In case study 2 this point has been validated by making a more accurate prediction with the help of data collected for past 48 time period. Still getting only past 9 days data the model is able to predict the 10th day value with reasonable fidelity.

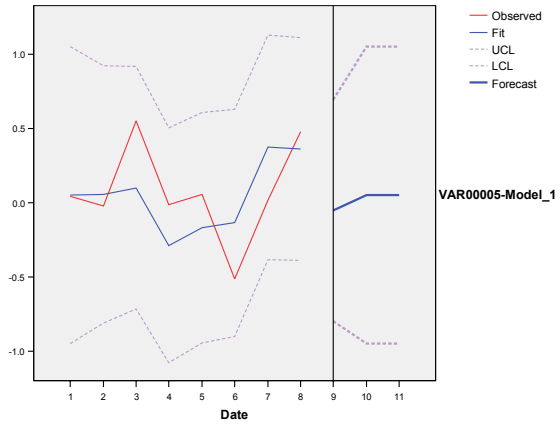

Figure 6. Time Plot of Actual Vs Forecast

3.2. Case Study 2

In this case study, the test data has been collected regarding remotely exploitable vulnerabilities with unauthorized access for a period of 48 weeks from our LAN using data collector component. The LAN is comprised of windows and Linux systems. The collected data is shown in Table 3. It can be observed from the Figure 7 that the number of vulnerabilities varies within a time period (here it is weekly) significantly making it a non-stationary time series. The collected data is then analyzed using data analyzer component in following subsections.

Table 3. Collected Data of remotely exploitable vulnerabilities with unauthorized access.

Time (weeks)	VC	Time (weeks)	VC	Time (weeks)	VC	Time (weeks)	VC
1	27	13	15	25	5	37	15
2	9	14	13	26	9	38	26
3	15	15	12	27	6	39	9
4	22	16	18	28	12	40	13
5	11	17	14	29	15	41	24
6	14	18	12	30	16	42	14
7	13	19	16	31	21	43	36
8	16	20	26	32	12	44	24
9	15	21	11	33	12	45	20
10	15	22	14	34	17	46	14
11	11	23	20	35	9	47	22
12	6	24	19	36	21	48	20

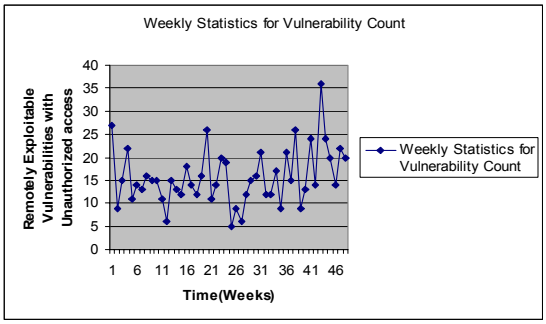

Figure 7. Plot of Original Time Series Data.

3.2.1. Satisfying stationary condition. Plotting the data against time scale clearly shows that in Figure 7 the series is a non-stationary one. So the data has been converted by taking the logarithm of vulnerability count (VC) and considering difference of values [11]. Essentially it yields a stationary time series (refer to Table 4 and Figure 8). Calculating and plotting the ACF (refer to Figure 9) for working time series using the formulas (refer to equations (2), (3) and (4)) shows there is an early cut-off after lag 11 and thus satisfies stationary condition. The PACF (refer to Figure 10) for the data has also been calculated using equations (5), (6) and (7).

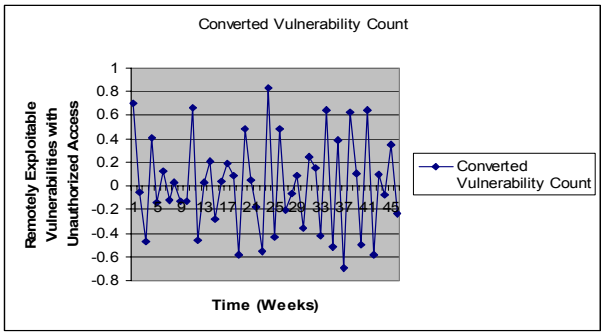

Figure 8. Plot of converted Time Series Data.

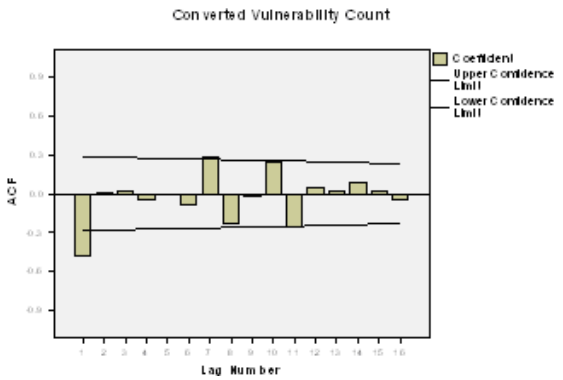

Figure 9. Autocorrelation Plot Function with 95% Prediction Interval.

Table 4. Converted Data of remotely exploitable vulnerabilities with unauthorized access.

Time (weeks)	VC	LOG(VC)	LOG(VC2) – LOG(VC1)
1	27	1.43136376	-
2	9	0.95424251	-0.477121255
3	15	1.17609126	0.22184875
4	22	1.34242268	0.166331422
5	11	1.04139269	-0.301029996
6	14	1.14612804	0.104735351
7	13	1.11394335	-0.032184683
8	16	1.20411998	0.09017663
9	15	1.17609126	-0.028028724
10	15	1.17609126	0
11	11	1.04139269	-0.134698574
12	6	0.77815125	-0.263241435
13	15	1.17609126	0.397940009
14	13	1.11394335	-0.062147907
15	12	1.07918125	-0.034762106
16	18	1.25527251	0.176091259
17	14	1.14612804	-0.109144469
18	12	1.07918125	-0.06694679
19	16	1.20411998	0.124938737
20	26	1.41497335	0.210853365
21	11	1.04139269	-0.373580663
22	14	1.14612804	0.104735351
23	20	1.30103	0.15490196
24	19	1.2787536	-0.022276395
25	5	0.69897	-0.579783597
26	9	0.95424251	0.255272505
27	6	0.77815125	-0.176091259
28	12	1.07918125	0.301029996
29	15	1.17609126	0.096910013
30	16	1.20411998	0.028028724
31	21	1.32221929	0.118099312
32	12	1.07918125	-0.243038049
33	12	1.07918125	0
34	17	1.23044892	0.151267675
35	9	0.95424251	-0.276206412
36	21	1.32221929	0.367976785
37	15	1.17609126	-0.146128036
38	26	1.41497335	0.238882089
39	9	0.95424251	-0.460730839
40	13	1.11394335	0.159700843
41	24	1.38021124	0.266267889
42	14	1.14612804	-0.234083206
43	36	1.5563025	0.410174465
44	24	1.38021124	-0.176091259
45	20	1.30103	-0.079181246
46	14	1.14612804	-0.15490196
47	22	1.34242268	0.196294645
48	20	1.30103	-0.041392685
49	Predicted Value	Predicted Value	Predicted Value

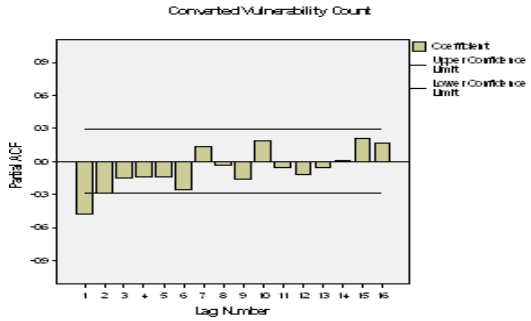

Figure 10. Partial Autocorrelation Function Plot with 95% Prediction Interval.

3.2.2. Model Identification. After examining the ACF and PACF (refer to Figures 9 and 10) behavior of the working time series, it can be shown that ACF has spikes at lag 1, 7, 8, 10, 11(refer to Figure 9) and PACF has spikes at lag 1 and 6 (refer to Figure 10). So PACF cuts off more abruptly after lag 6 than ACF. This leads to tentatively identify that our model is an autoregressive model of order 6. The significance constant term (refer to equation (1)) in the proposed model needs to be analyzed. The constant term, δ can be given as

$$\delta = \mu * (1 - \phi_1 - \phi_2 - .... - \phi_9) \quad (14)$$

For this particular case $\delta = \mu$ as no autoregressive operator has been identified. The sample mean of the stationary time can be given as

$$z_m = \frac{\sum_{t=b}^{n} z_t}{(n - b + 1)} \quad (15)$$

So constant term in the model can only be included if the absolute value of

$$z_m * \sqrt{(n - b + 1)} \Big/ s_z \text{ is greater than 2}$$

The term, S_z, is given by

$$s_z = \sqrt{\frac{\sum_{t=b}^{n} (z_t - z_m)^2}{(n - b + 1) - 1}} \quad (16)$$

In this case, the value of $Z_m * (n - b + 1)^{1/2} / S_z$ is > 2. So the constant term 0.06290293 has been included in the model. The final working model can be represented as

$$z_t = \delta + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \theta_3 a_{t-3} - \theta_4 a_{t-4} - \theta_5 a_{t-5} - \theta_6 a_{t-6} \quad (17)$$

3.2.3. Parameter Estimation. The resulting time series model is shown in equation (18) using θ_1, θ_2 etc.

$$z_t = 0.0629 + a_t - 0.8325a_{t-1} - 0.6235a_{t-2} - 0.4506a_{t-3} - 0.4468a_{t-4} - 0.4194a_{t-5} - 0.33334a_{t-6} \quad (18)$$

3.2.4. Prediction. To predict the total number of remotely exploitable vulnerabilities that will be detected in our network on 49th week the equation (19) has been used as follows.

$$z_{49} = 0.0629 + a_{49} - 0.8325a_{48} - 0.6235a_{47} - 0.4506a_{46} - 0.4468a_{45} - 0.4194a_{44} - 0.3333a_{43} \quad (19)$$

Where, a_{49} is the random shock or the difference between observed and predicted value. Here it is 0.01648399 (approx). Putting all these data in equation (19) the predicted value for 49th week will be -0.032314. Converting the value -0.032314 as $V=LOG(VC_{49}) = LOG(VC_{48}) + (-0.032314)$ and $VC_{49} = 10^V$ the predicted value for 49th week will be 18.56 ≈ 19. The measured value generated on 49th week was 22, and the predicted value is 19. It is evident from the plot (refer to Figure 11) that the

predicted value using the proposed model closely matches with the measured value.

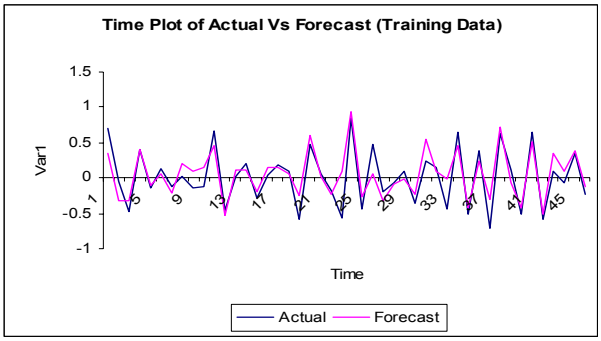

Figure 11. Time Plot of Actual Vs Forecast

4. Conclusion

In this paper we proposed a statistical approach for vulnerability prediction, which will help in identifying the most vulnerable areas for near future in an organizational LAN. As the time series value is highly dependent on the past values so more numbers of inputs feed to the model, the probability of getting a better prediction increases. Two case studies have been presented to demonstrate the applicability of the model. In case of certain peaks or falls in time-series due to unforeseen and unpredictable reasons which can also be modeled as irregular noise-component in the time-series model. This proposed model can be further extended to a causal forecasting model with multiple time series data, where the output series (vulnerability count) predicts on the basis of past values as well one or more related input time series (like patch time series).

5. References

[1] H.S. Venter, J.H.P. Eloff, "Vulnerability Forecasting – a conceptual Model", *Computers & Security*, Elsevier, 2004, Vol. 23 (6), pp. 489-497.

[2] R. Gopalakrishna, E. Spafford, and J. Vitek, "A Trend Analysis of Vulnerabilities", CERIAS TR 2005-06, 2005.

[3] Browne, H.K., Arbaugh, W.A., McHugh, J., Fithen, W.L., "A trend analysis of exploitations," IEEE Symposium on Security and Privacy, 2001, pp.214-229.

[4] Alhazmi, O.H., Malaiya, Y.K., "Quantitative vulnerability assessment of systems software," *Reliability and Maintainability Symposium,* Jan. 24-27, 2005, pp. 615- 620.

[5] W. A. Arbaugh, W.L. Fithen, J. McHugh, "Windows of Vulnerability: A Case Study Analysis", *IEEE Computer*, December 2000, Vol. 33, No. 12, pp. 52-59.

[6] H.S. Venter, J.H.P. Eloff, "Assessment of Vulnerability scanner", Network Security, *Elsevier Science*, February 2003, pp 11-16.

[7] O. H. Alhazmi et.al, "Security Vulnerabilities in Software Systems: A Quantitative Perspective", Annual IFIP WG 11.3 working conference on data and applications security No19, Storrs CT, 20051973, vol. 3654, pp. 281-294.

[8] E. Rescorla, "Is finding security holes a good idea?" Proc. Third Annual Workshop on Economics and Information Security (WEIS04), May 2004, pp. 1-18.

[9] R. Anderson, "Security in Open versus Closed Systems—The Dance of Boltzmann, Coase and Moore", Conference on Open Source Software: Economics, Law and Policy, Toulouse, France, June 2002, pp. 1-15.

[10] Tim Shimeall, "Models of Information Security Trend Analysis", Available at http://www.cert.org/archive/pdf/info-security.pdf

[11] Bowerman, B.L, and R.T. O'Connell, *Time series forecasting, unified concepts and computer implementation*, 2nd ed., Duxbury Press, Boston, 1987.

[12] Kazmier, L.J., *Basic Statistics for Business and Economics*, Mcgraw Hill, New York, 1979.

[13] Honeynet Whitepaper, "Know Your Enemy: Statistics", July 2001, Available at http://www.honeynet.org/papers/stats/.