Evaluating the Readiness of Cyber Resilient Bangladesh

Touhid Bhuiyan[1], Delwar Alam[1], Tanjila Farah[2]
Daffodil International University[1], North South University[2]
Dhaka, Bangladesh

Abstract

Digitization is a fundamental driver of development and economic growth of the world in both urbanized and emerging markets. Vision "Digital Bangladesh by 2021" marks the countries journey towards becoming a developed nation and embracing the knowledge-based economy as means of achieving it. However the dependency on digitization increases vulnerabilities and risks including cybercrimes such as hacking, intrusion, fraud, harassment and more. Bangladesh government is aware of these threats and has initiated various measures to prevent them. In this paper we discuss the mechanism Bangladesh government has taken to make Bangladesh cyber resilient and the present condition of that initiative. This paper also presents a statistical analysis of various attacks and their results on the websites of Bangladesh.

1. Introduction

Modern world is driven by information and communication technology (ICT). The world is evolving everyday with new knowledge and technology. As a developing country Bangladesh is not that far from realizing the potential of ICT in developing a nation [1]. Yet Bangladesh has a lot of catching up to do [3]. Bangladesh government has taken various initiatives to enter the race of development with ICT. Bangladesh government has initiated "Vision 2021" to empower the people the knowledge based economy by making a "digital Bangladesh". Now emphasis on knowledge based economy is the key development priority for Bangladesh. The use of digital information systems and digital services throughout most of the public and private organizations has been made compulsory by the government. Most of the organizations have started shifting their services towards digitized systems. This change is already affecting the social, educational, and economical growth of the country. However the reliance on digitization bring rising vulnerabilities and threats to the critical private national information infrastructure (CPNII) [4]. Security is one of the main problems of digitization. As with digitization the CPNII's governances and control become digitized, it also become vulnerable to various cyber threats. Acknowledging the increasing cyber threat on the digitization process Bangladesh government has formed "The National Cyber-security Strategy of Bangladesh" [4]. To make this strategy successful the government has joined hand with ITU-ABI research's Global Cyber security Index (GCI) project [3]. This project measures the commitment of countries to cyber security and overlays the path for improvement. This paper discusses the Bangladesh government's initiative to implement GCI and its current condition. It also presents a statistical analysis of the current condition of few well known cyber vulnerabilities present in web applications of Bangladesh.

The paper is organized as follow. In section II we discuss the current condition of cyber readiness of Bangladesh. In section III, we discuss the existing penetration testing methodology and the methodology we have used in this study. In section IV, the steps of various injection techniques are discussed. In section V statistical analysis of the data collected through out with study is presented. Then we conclude in section VI.

2. Current status of Cyber readiness of Bangladesh

Cyber crime can broadly be defined as a criminal activity involving an information technology infrastructure, including illegal access (unauthorized access), illegal interception, data interference (unauthorized damaging, deletion, deterioration, alteration or suppression of computer data), systems interference (interfering with the functioning of a computer system by inputting, transmitting, damaging, deleting, deteriorating, altering or suppressing computer data), misuse of devices, forgery (theft), and electronic fraud. Unfortunately, cyber-security is not yet at the core of many Bangladesh's national and industrial technology strategies [2]. Government and private organizations need to be aware of impact of cyber security threats and identify areas where cyber security needs to be enhanced. The government needs to acquire the tools

and infrastructure to prevent, detect, prosecute, and make people aware of the cyber threats.

Bangladesh Government has started taking initiatives to make Bangladesh cyber resilient. "Cyber Security Act 2015" and collaboration with ITU-ABI research's Global Cyber-security Index (GCI) project are two of the major steps taken by the government. The GCI project measures the commitment of countries to cyber security. To corroborate the GCI project it is important to have institutional structures to deal with cyber incidents and attacks is a genuine problem in responding to cyber threats. ITU, in collaboration with IMPACT, is helping countries to establish their National Computer Incident Response Team (CIRT), which serves as a national focus point for coordinating cyber security incident response to cyber attacks in the country. The objective of the CIRT assessment is to define the readiness to implement a national CIRT. To collaborate with GCI project Bangladesh government has formed bdCERT. It is the Computer Emergency Response Team of Bangladesh. ITU completed a CIRT assessment for Bangladesh at Dhaka, Bangladesh from November to May 2010 and recognized bdCERT as an official CIRT. From this assessment a "Cyber-wellness Profile" for Bangladesh has been generated. According to this profile more the 6.5% of the population of Bangladesh uses Internet [3].

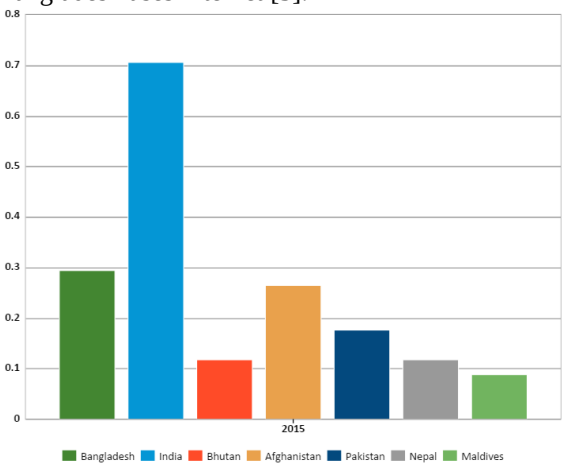

Figure 1. Cyber security Index of South Asian Countries

The GCI assessment is done based on five parameters: Legal, Technical, Organizational, Capacity Building, and Cooperation. Based on these parameters a cyber-security index and rank is given to each collaborating country. This assessment is done over 195 countries all over the world. According to the GCI results of 2014, Bangladesh scored 0.5000 in legal, 0.3333 in technical, 0.1250 in organizational, 0.2500 in capacity building, and 0.3750 in cooperation. Combining these results Bangladesh's cyber security readiness index became 0.2941 and Asia pacific regional rank became 11[4]. Bangladesh ranked 11 globally over cybersecurity preparedness [5]. Figure 1 shows the comparison of GCI cyber security index of South Asian countries (i.e. India, Pakistan, Bangladesh, SriLanka, Nepal, Bhutan, Afghanistan, and Maldives). Among the eight countries India scored the best and Bangladesh is the second best [4].

Figure 2 shows the comparison of global ranking of the South Asian countries. Similar results as cyber security index can been seen in here. Bhutan and Nepal ranked the same in global ranking as shown in Figure 2. This ranking indicates that they have the same level of readiness [4].

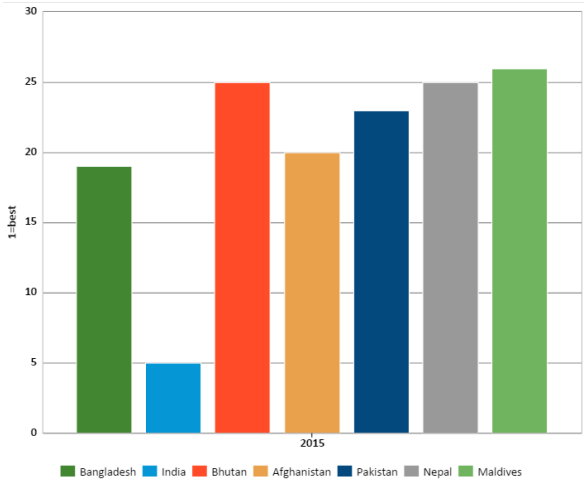

Figure 2. Global ranking

CRET assessment suggests that Bangladesh does not have any officially approved national (and sector specific) cyber security frameworks for the certification and accreditation of national agencies and public sector professionals. Bangladesh also does not have any officially recognized national or sector-specific research and development (R&D) programs/projects or certified government and public sector agencies for cyber security standards, best practices and guidelines to be applied in either the private or the public sector. bdCERT is working on developing manpower by organizing educational and professional training but they are not sufficient. To facilitate sharing of cyber security assets across borders or with other nation states, Bangladesh has official recognized partnerships with the following organizations: ITU, APCERT, and OIC-CERT. Bangladesh is a member of the ITU-IMPACT initiative and has access to relevant cyber security services. In addition, bdCERT has joined Asia Pacific CIRT (APCERT) cyber Security Drills. It also attends regional events organized by other CERTs like Japan CIRT (JPCERT) and Korean CIRT (KRCERT) [3].

With all these initiatives Bangladesh is still in its initial steps of providing cyber security to the nation. Bangladesh government needs to take fast actions implement all the strategies and CERT guidelines. People's awareness towards the cyber vulnerabilities is a big step toward the foundation of cyber

resilience. In this study to measure the current condition web applications and people's awareness to government's initiative for creating cyber resilience, we have implemented few very basic attacks using well known techniques on the web applications of Bangladesh. For attack or injection purpose we have used known penetration testing methodology. In next sections we will discuss the techniques and their findings in detail.

3. Methodology of Penetration Testing

Penetration testing methodologies are systematic approaches of test techniques to corroborate the existence of vulnerabilities. It focuses on locating and targeting exploitable defects in the design and implementation of a web application.

Penetration testing could be performed manually or by using automatic tools [13]. Successful penetration testing depends on the methodology. Various groups follow various methodologies for penetration testing. The existing testing methodologies include 4 to 7 steps. Though the names of these steps are different in various methodologies the functions are the same [11]. NIST penetration testing methodology is used in this research [13]. This methodology includes four phases: planning, discovery, attack, and reporting. These phases are further subdivided to make the steps of testing method accurate as shown in Figure 3.

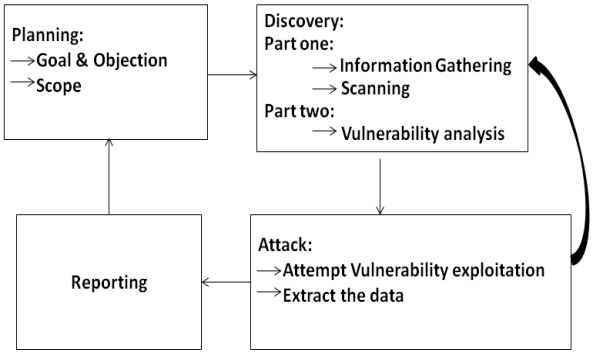

Figure 3. Penetration testing methodology

A. Planning

In the planning phase, rules are identified, management approval is finalized and documented, and testing goals are set. Information gathered in this phase are needed for assessment execution such as the web services to be assessed, the threats of interest against the services, the security controls to be used to mitigate those threats, and to develop the assessment approach [11]. The planning phase sets the groundwork for a successful penetration test. No actual testing occurs in this phase. The assessment process in this phase includes plan to address goals and objectives and scope [13].

1) *Goal and Objective:* All web applications are potentially vulnerable to various attacks. In this step the target web applications are identified and the threats to be examined are determined. For this research the goal was set to check the SQLi vulnerability of the financial web applications of Bangladesh. A financial web application is identified as online software used for financial transaction and storing user information such as: online payment and bank transfer.

2) *Scope:* The scope of penetration testing depends on the information available to the tester. Scope is considered in three stages: white-box testing, black-box testing, and gray box testing [11]. The black-box testing approach is considered
in this paper suggesting that we had no prior knowledge about the application's usages and coding system. The applications were accessed through browser. The attack vectors that are being used include: URL parameter, HTML from parameters, cookies, HTTP header etc [12].

B. Discovery

Primary goals of the discovery phase are to identify and validate vulnerabilities [11]. This phase addresses activities associated with the assessment methods and techniques. Although specific activities for this phase differ by assessment type, upon completion of this phase assessors will have identified system, network, and organizational process vulnerabilities [13]. This phase of penetration testing includes two parts. First part is for information gathering and scanning. The second part is for vulnerability analysis of the web applications.

1) *First Part:*

Information Gathering/ Foot printing: This research focuses on SQLi penetration testing approach. Understanding the underlying SQL query allows to craft correct SQLi statements. For that purpose finding information about the vulnerable web links to identify the input fields, hidden fields and post requests fields are necessary. Google dork command inurl: has been used to find the vulnerable input links [14]. 'inurl:' dork retrieves links where variable is used to access data from database. These variables are the vulnerable input fields used to inject the malicious query and retrieve unauthorized information. Information gathering phase is also known as the survey and assess method to determine complete information of the potential target.

Scanning: Vulnerability scanning is a process to recognize weaknesses of targeted web services, network, and applications. In this step network port and service identification is conducted to identify potential targets. The results of scanning include: the open ports and services running on these ports. Based on these results type of vulnerabilities and suitable attacks are determined. Through this process information about type and version of database, search engine, and user privilege levels could be

retrieved. Identifying the privilege levels increase chance of gaining the access as an authentic user [13]. After scanning next step could be to obtain the password and compromise the system [14]. Scanning can be done with or without tools. Manual scanning has been performed for collecting data in this research.

2) *Second Part:*

This part of discovery phase is for vulnerability analysis. Based on the findings in first part, suitable and potential threats are determined in this step. For this research, the findings of first part led to SQLi threat as all the web applications tested showed vulnerability to this threat. SQLi is a technique that takes advantage of non-validated input vulnerabilities and injects malicious SQL queries to the database through a web application interface. These commands are un-authorized yet executed in the back-end database due to the functional structure of the web applications [13]. Various SQL queries are injected in the input fields found in the first part to generate error messages. If an error message is detected that would suggest a potential SQLi vulnerability is detected. These error messages are also essential for extracting information from the database.

C. Attack

Executing an attack is the primary function of any penetration test. Attack stage verifies the potential vulnerabilities identified in vulnerability analysis of discovery stage by exploiting them. It may also reveal more vulnerabilities of the target web application. Attack phase includes attempt to: gain access, escalate privileges, and system browsing. This stage is divided in two steps: attempting vulnerability exploitation of web applications and extraction of data.

1) *Attempt vulnerability exploitation:* In this phase malicious SQL query is injected in the input field of the web application. Depending on error information found in vulnerability detection phase attack techniques vary. SQL syntax may varies based on the type of back end databases. Thus the SQLi technique varies based on error message and database. There are various SQLi techniques such as: Union based, error based and blind injection. For this research we have focused on Union based SQLi.

2) *Extract the data:* Once the SQL injection attack is successful, the attempt to extract table names, column names, and table data from the target database initiates. Depending on the aim of the testing, interaction with the database to extract are performed. The tester can go further to compromise the whole target network by installing trojans and planting key-loggers [4]. Our goal was to find the table and their column names from database. No harm was done to the databases or the internal networks during the testing process.

This study aims to test the cyber threat resilience of the web applications of Bangladesh. To fulfill the goal, the web applications are tested for basic injection vulnerabilities such as: basic, error based, 64 based, local variable, global variable, and blind injection Structured Query Language injection (SQLi). Though various other vulnerabilities exist, the scope of this paper covers only various types of SQLi vulnerabilities. The testing is performed using black-box approach. All tests are implemented manually. No tools were used.

4. Injection Vulnerabilities

Due to various coding malpractice the web applications are vulnerable to various injection attack. The goals of these attacks are to get unauthorized access to the web applications. The focus of this study is SQLi. Web browsers platform is used for employing this attack. The language used for this attack is structured query language (SQL). This language is also used for communicating with database in backend of the web applications. Users write the URL of the web application in their browser. This URL is converted to SQL query or request by the web application server and sent to the database server as shown in Figure 4. The database retrieves the requested data and returns the output to the browser.

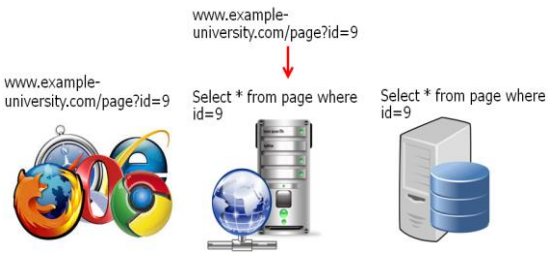

Figure 4. URL to query conversion

When the URL input is crafted to return unauthorized output, The query is called SQLi. The crafted input is converted into query in the same process shown in Figure 1. This type of Vulnerability occurs when the user input parameters are not verified before forming the query.There are various types of SQLi techniques such as:

A. Basic SQLi / Union base
B. Error Based
C. Error based double query
D. 64 based
E. Local variable/ Global variable
F. Blind injection

A. Basic SQLi/ Union Base

This is the basic of any SQLi techniques. The first step of SQLi is vulnerability checking and balancing the query. The vulnerability checking is done by adding single quote (') after the URL as shown in

Figure 5. If adding single quote returns an error message [1] it confirms the existence of SQLi vulnerability. This query is balanced by adding --+ and at end of the URL. This syntax comments out any query written after --+. Thus the authorized query is blocked [7]. The next step is injection query to get un authorized data. In Figure 5 the query written between single quote and --+ the injection query to retrieve database table names. This query is rewritten to retrieve column name and data from the database.

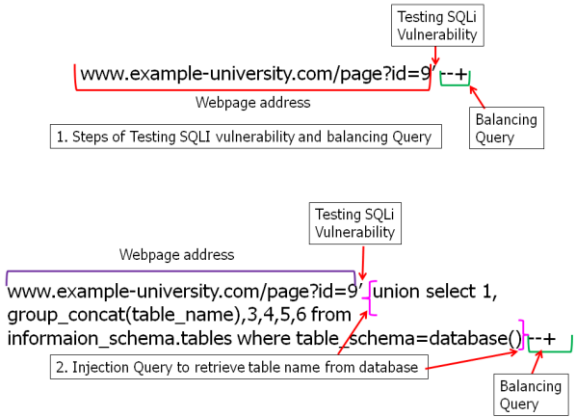

Figure 5. Steps of Normal SQLi

B. Error Baeed SQLi

Error based sql injection takes advantage of poor error handling in the codes of web application. Error based injection is attempted when basic inject is unable to retrieve all the data. Error based injection is usually attempted when the "union" query fails in normal SQLi [9]. "Union" query retrieves the vulnerable columns in the database. Sometimes this command is blocked in the backend database. In such scenario error based injection is performed. Error based injection forces the database to return errors messages. Invalid SQL statements are injected to database via HTTP request to generate error messages. The goal is to retrieve useful data from the database with these errors. The Steps of Error based Injection are as follow:

1) *Checking for error based injection:* First two steps of error based injection are SQLi vulnerability checking and balancing. These steps are the same as basic SQLi. Then the "order by" or "group by" query is used to find the number of columns in the database of the web application. Next the "union select" query is written to find the vulnerable columns. This query is considered as a failed query when the website runs the injection query but no output is shown. When "union select" query fails it is considered that the website has error based injection. Other errors such as: "Unknown Column 1", "Select statement have different columns in Query", and "error #1604" are also identified as error based injection vulnerability.

2) *Injecting query:* Once the error based injection vulnerability is identified step by step database version, name, table names, column names and the data in the columns are retrieved. The error based injection query is written between single quote (') and --+ in Figure 6.

```
Retrieving MySQL version:
http://www.example.edu.bd/show.php?id=84' or 1 group by
concat_ws(0x3a,version(),floor(rand(0)*2)) having min(0) or 1
--+
```

Figure 6. Error based injection to retrieve database version number

This error based injection is for retrieving database version. The query analysis is as follow:

1. Though the query is injected in a where clause in the database "or 1" condition is added (which is equivalent to "or true") at both end of the query is used to cancel out the previous conditions (i.e. "id=84") and display as many rows as possible.
2. Aggregate function GROUP BY is used to group columns that has two identical values on different rows.
3. Concat_ws() function is used to Concatenate multiple functions.
4. 0x3a is the hex value of ":". This is used as separator.
5. version() is used for retrieving the string of MySQL version number.
6. Rand(0)*2 – rand() returns a random floating-point value between the range 0 to 1. The rand(0) represents random function with seed value. This returns a repeatable sequence of random values. Then this sequence is multiplied by 2.
7. floor() returns the largest integer value not greater than the number specified as argument. In this query the following sequence of numbers: 0, 1, 1, 0, 1, 1... are generated based on the random seed argument 0. This function is chosen to generate the error statement quickly.
8. having min(0) - By itself, this is illegal as the having clause requires a condition. Here the having clause is used to satisfy the aggregate function "group by".

The query shown in Figure 6 concatenates the version string and a sequence of numbers across a group. The sequence of numbers are generated by floor() and rand() functions. The "group by" clause requires unique group keys. Since the version() might return the same value each time, concatenating that and the output of FLOOR(rand(0)*2) results in two different numbers (0, 1) and then a second instance of 1. This 1 is a duplicate, which causes an error (duplicate entry for group key), which is displayed back to the user. That error returned by the injection query is shown in Figure 7. This error reveals the database version number.

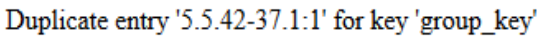

Figure 7. Error reveling database version

3) *Retrieving database name:*
The process of retrieving the database name is the same as retrieving the version number. Only change is that the version() function is replaced by database() function as shown in Figure 8.

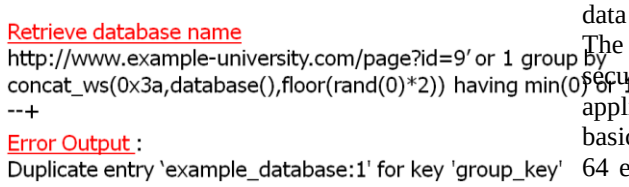

Figure 8. Error based injection query for retrieving database name and the output

4) *Retriving table name:*
The injection queries for retrieving table name and the outputs are shown in Figure 6. The query is the same as basic SQLi. This query is re-written to retrieve column name and data. These queries are also same as basic SQLi injection.

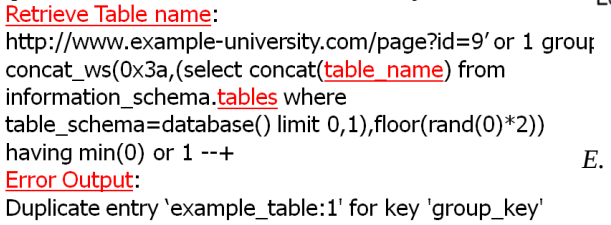

Figure 9. Retrieving table name with error based injection

C. Error based Double query injection

When error based injection vulnerability is identified by but the injection steps fail to return output, double query injection is attempted [9]. Double query injection works with similar principal as the error based injection. It also aims to force database to provide information by generating error messages. In double query injection multiple queries are concatenated into one query to confuse the database. This confusion forces the database to generate error and reveal information about the database. Error based double query injection is similar to error based injection with some variations. Example for double query injection to retrieve the database name is shown in Figure 10. The query shown in Figure 10 is used to retrieve database name.

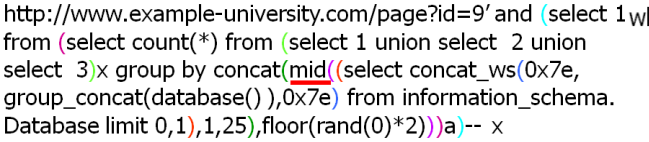

Figure 10. Double query injection

D. 64 based injection

Base64 schemes represent binary data in an ASCII string format by translating it into a base-64 representation. Base64 encoding schemes are commonly used when there is a need to encode binary data that needs to be stored and transferred over media that are designed to deal with textual data [10]. This is to ensure that the data remains intact without modification during transport. In URL the get data is usually encoded with 64 based encryption. The goal of URL encoding in 64 based is to provide security and prevent basic SQLi. To attack a web application with base 64 injection, similar steps as basic SQLi is used except the injection codes are base 64 encrypted before injecting. Example of 64 based injection is shown in Figure 11. Query for testing and balancing SQLi are shown in the first line of the Figure. The injection query is 64 based encrypted as shown in second line of the Figure 11.

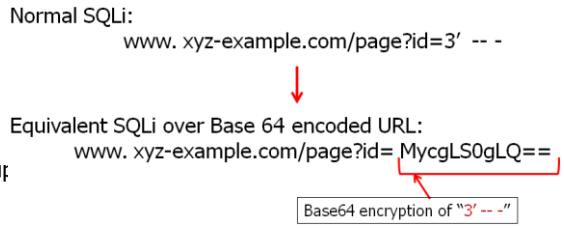

Figure 11. Evading 64based encoding

E. Local variable injection/ Global Variable injection

This type of injection is possible in web applications using MySQL database. Local variable is user defined variable. MySQL syntax "@variable name:=" is used to assign a variable. This injection is used for WAFs that block structured SQL commands such as union select, group concat etc. from executing [10]. So these commands are sent to the database wrapped in a locally defined variable. An example of this attack is shown in Figure 12. The example website xyz-example has two vulnerable columns 1 and 4. In column 1 a variable x is locally defined and then in this variable, union select command to find the vulnerable table name is assigned in this available. The server then processes the query by converting it to SQL command. The values printed in column 4 by running the command in @x.

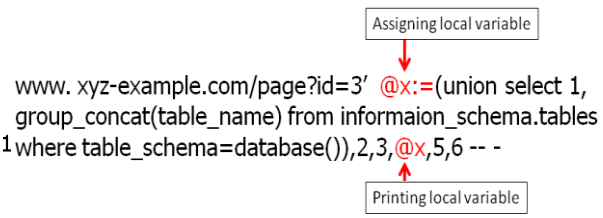

Figure 12. Local variable SQLi

Global variable are universal syntax of MySQL database such as @@PORT, @@HOSTNAME, @@ HAVE_OPENSSL, @@version etc [10]. If local

variable assignment is blocked by WAF then global variable assignment is used as shown in Figure 13.

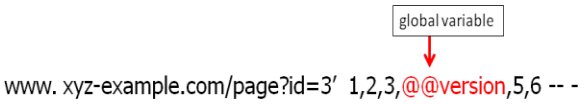

Figure 13. Global variable SQLi

F. Blind injection

When vulnerability testing steps ensure the presence or point of SQLi but the basic or error based SQLi does not retrieve the expected results, Blind Injection is attempted. Blind injection never returns data directly such as other SQLi techniques [9]. In Blind inject data is retrieved one letter at a time. For retrieving each correct letter, the database is attacked with all possible values through injection query. Binary search technique is also used to optimize the search for correct information. The steps of blind injection are as follow:

1) Checking for Blind injection: Blind injection is detected by the Boolean response received (True or False) from the database after inserting attack query. After vulnerability checking and query balancing steps are performed as show in Figure 2. To check if blind injection is possible a true condition such as 1=1 as shown in Figure 14 and a false condition such as 1=0 shown in Figure 10 are entered in consecutive queries. The true condition returns the same page. The query shown in Figure 14 return everything there is in the page with id=9 and where 1=1. In this query page with id 9 is always true and 1=1. TRUE and TRUE implies true in Boolean.

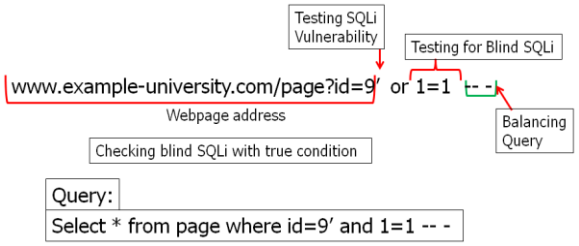

Figure 14. Checking true condition for blind injection

Next if the FALSE condition returns error or blank page, this confirms the presence of Blind SQL Injection. As shown in Figure 15 the query finds the page with id 9 but the condition 1=0 is false. True and false implies false in Boolean and thus return the error page.

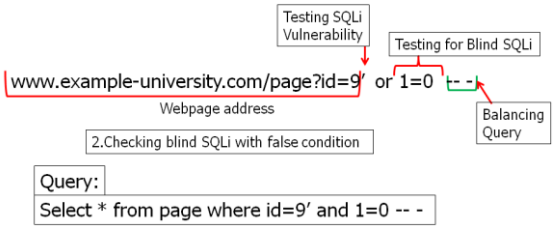

Figure 15. Checking false condition for blind injection

2) Injection Query: If both true and false condition return results accordingly then the possibility of existence of blind injection is confirmed [9]. The next step is to extract other information such as: database version, database name, table name, column name, and data. For example let us assume the back end database is using MySQL version 5.1.3 and we want to retrieve the database name which is "abc" in this example. To retrieve information with blind injection ASCII and Substring Functions are used. Generally ASCII is the numerical representation of letters or characters. For example ASCII value n is 110. And the Substring or str function returns the letter or value at a specific position. For example:

substring('abc',1,1) will return a.
substring('abc',2,1) will return b.
substring('abc',3,1) will return c.
substring('abc',4,1) will return null.

To retrieve any information the query is written for each letter at a time. It is a trial and error process. It in continued until the true answer is revealed. In the example here, to retrieve the database name, the query is written as shown in Figure 16.

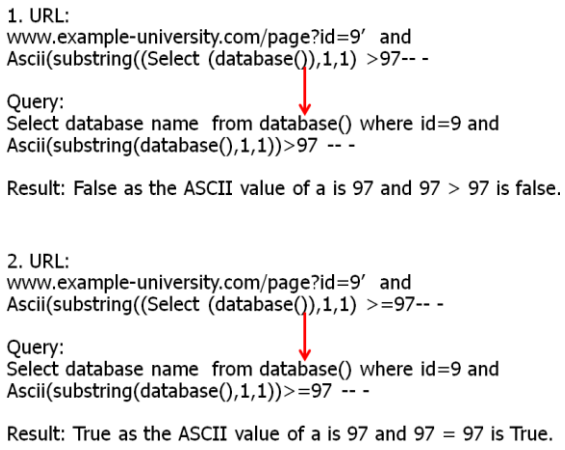

Figure 16. Retrieve the first letter of database name

As mentioned in the example database name here is "abc". To retrieve the name, the checking starts from the first letter "a". The ASCII value of "a" is 97. Initially the ASCII condition is written as "grater the 97" shown in the first query in Figure 11. This returns error as the condition is false. Then the query is

written for ASCII ">=97". This returns true value. Then it is checked for ACSII ">98" and the query returns false. This confirms that, the first letter of the database name is ASCII value 97 and letter 'a'.
The substring function is closed by a bracket and separated from ASCII function. Inside substring function any query could be written for retrieve the expected data. In the example shown in Figure 17, we expected to retrieve the database name, and thus "select database()" is written inside substring function. After the "database()" function 1,1 is written to identify the position of the letter to be retrieved. The first 1 indicates to select/extract the first character of the database name and the second 1 indicates comparing the selected value with the given ASCII value condition. In the example shown in Figure 17 it is compared to ">=97". And then it returns true or false.

2. URL:
www.example-university.com/page?id=9′ and
Ascii(substring((Select (database()),2,1) >=98-- -

Result: True as the ASCII value of b is 98 and 98 = 98 is True.

3. URL:
www.example-university.com/page?id=9′ and
Ascii(substring((Select (database()),3,1) >=99-- -

Result: True as the ASCII value of c is 99 and 99 = 99 is True.

4. URL:
www.example-university.com/page?id=9′ and
Ascii(substring((Select (database()),4,1) >=99-- -

Result: Null as the database name here has 3 letters only.
There is no 4th character/letter here.

Figure 17. Retrieve the rest of the database name

The same process is used to retrieve the rest of the letters in the database name as shown in Figure 18. The substring functions position value changes from 1,1 to 2,1 and 3,1 as per the position of the letter in the name. This process of letter searching is similar to binary search techniques. In a black box testing approach, testers have no idea about the length of the name. To confirm the length of the database name in the example shown in Figure 17, the same query is written again with position value changed to "4,1". This will return null value as the length of database name is 3 character. Thus it is confirmed that the database name is retrieved. This process is repeated for retrieving all the information from the database. To retrieve data that are not associated with a direct MySQL keyword such as table name, column name and data, the query is written as shown in Figure 18.

www.example-university.com/page?id=9′ and
Ascii(substring(Select column_name from
information_schema.columns where
table_name='abc',1,1)>=99-- -

Figure 18. Retrieving column name from example table

5. Vulnerability Analysis

In this study we have attempted to test the vulnerabilities of around 2500 web applications of Bangladesh region. These web applications include: government, private, educational, bank, financial, health care, and ecommerce web applications. All web applications are not vulnerable to all kind of injection attacks. The results are analyzed based on the dataset.

A. Dataset 1: Basic SQLi

In our first dataset, we have tested 600 web applications of Bangladeshi domain ".bd" for Basic SQLi vulnerability. In this dataset 400 web applications were found vulnerable to Basic SQLi. About 65% of the web applications were vulnerable to single quote (') attack. Another reason of this vulnerability is the development language used to build the web applications [8]. In the dataset 33% of the vulnerable web applications were built using php version 4.9 and older then this as shown in Figure 19. 54% were built using version php 5 or higher. From the rest od the dataset, 4% were built using joomla and 9% were built using Microsoft asp.net.

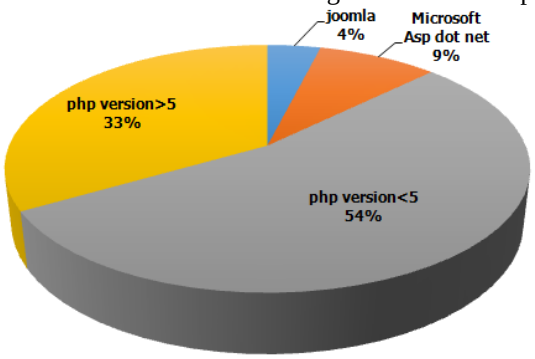

Figure 19. Web application vulnerability level based on development language

B. Dataset 2:union based, error based, and double query injection

In dataset 2, around 359 Educational web applications (i.e. school, college, university, training center e.t.c.) were tested for basic, error base, error based double query injections. Among them 309 websites are found vulnerable to various types of SQLi attacks.

a) Analysis of data-set based on type of vulnerability

In the 309 vulnerable website 199 websites are vulnerable to basic SQLi. From the rest 101 websites 37 are vulnerable to error based SQLi and 50 websites are vulnerable to double query injection. Rest of the website shown vulnerability but the type of vulnerability is not assured. The statistics of various vulnerabilities are shown in Figure 20. The graph implies that most of the educational websites aren't build using proper coding techniques and don't have protection from basic SQLi attacks.

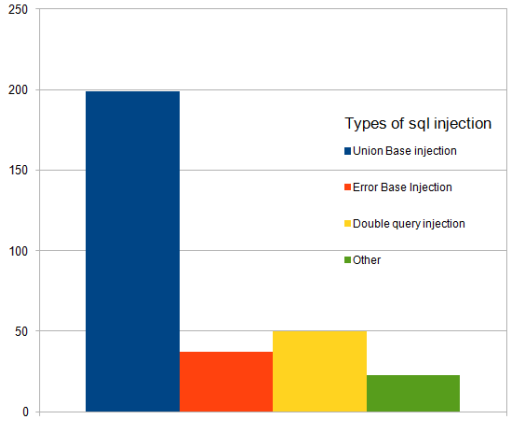

Figure 20. Types of SQLi and their aggression

b) Analysis of data-set based on vulnerability level

The vulnerability level is identified by the type and amount of data that could be retrieved from attacking the web applications [8]. The statistics of vulnerability level of educational websites of Bangladesh in shown in Figure 21. In the dataset 63% wesites are highly vulnerable as we could retrieve nearly any data and also change the data. 18% websites have vulnerability but that can't be severely harmed. The rest 19% are vulnerable but can't be harmed.

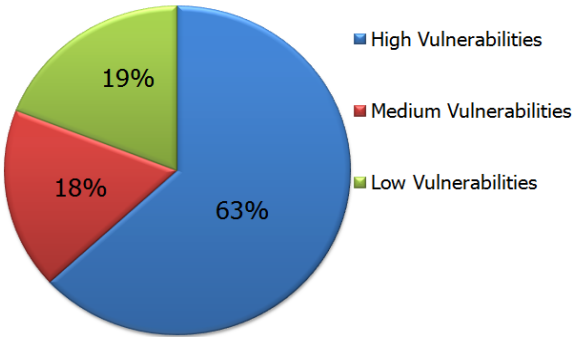

Figure 21. Vulnerability level of educational web applications

C. Dataset 3:Financial web applications

In this dataset we have tested the vulnerability of the financial web applications of Bangladesh. This is an ongoing research. This analysis is based on manual penetration testing of 108 financial web applications based on Bangladesh region. This is a small subset of the intended web application to be tested. For analysis purpose web applications of various genres are selected which include: bank, e-commerce website, brokerage house, insurance company, educational institute. In current data set of vulnerable web applications 67 vulnerable web applications are based on E-commerce, 21 are banking web applications. The rest of the applications include 3 brokerage houses, 2 insurance companies, 15 others. In financial web applications have various signature based Intrusion detection system (IDSs) installed. Yet these IDS's could be breached. The statistics in Figure 22 indicates that in our dataset, 60% of the web applications are with no or insufficient security system installed. They are vulnerable to basic SQLi. 30% of the web applications have MOD security IDS that are also vulnerable to SQLi. The rest of the 10 percent vulnerable web applications have forbidden 404 BAD request securities in them.

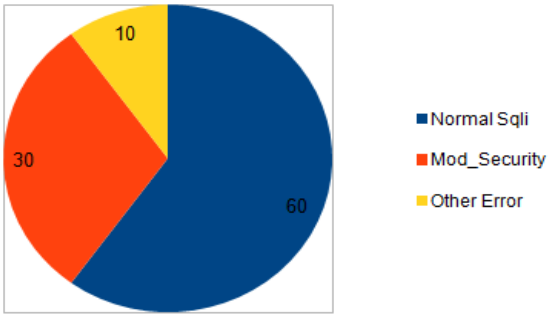

Figure 22. Vulnerability types of financial web applications

We have sub-divided the financial web applications based on the companies they belong to. Then the type of vulnerabilities and web applications are listed in Table 1. As shown in the table 80% of the tested web applications of Banking site of Bangladesh have no security systems installed and they are vulnerable to basic SQLi vulnerability

Table 1. Type of financial web applications and vulnerabilities

Sector	Basic SQLi	MOD security	Others
Bank	80%	20%	0%
E-commerce	65%	30%	5%
Brokerage House	67%	33%	0%
Insurance Company	100%	0%	0%

D. Dataset 4:64 based, local, and global variable injection

In our next dataset, we have tested 526 web applications for 64 based local and global variable injections. In this dataset 379 web applications showed vulnerablety to these web applications. The statistics is shown in Figure 23. In the dataset 61% are vulnerable to 64based SQLi technique, 34% is vulnerable to local variable SQLi and 5% are vulnerable to global variable SQLi.

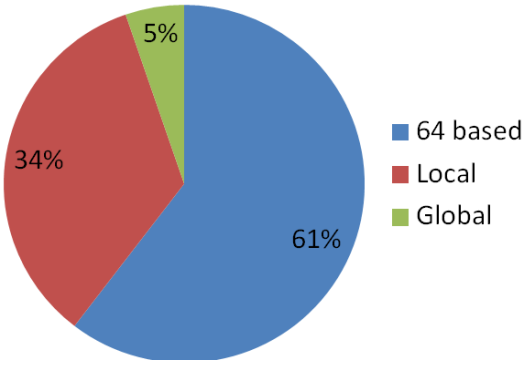

Figure 23. 64 based, local and global vulnerabilities

6. Conclusion

As a developing nation, Bangladesh's one of the biggest achievement is "Digitization". The government is taking all necessary steps for significant step towards achieving e-governance. The government is planning to bring the whole governance system under one public network. The primary objective of the project is to ensure internet connectivity in all the districts of Bangladesh. The plan includes a National Data Centre, which is also going to be established to host all the government websites and administrative offices and to connect the offices with each other. Keeping the security aspects of virtual establishment in mind, the government has also decided to train and build cyber security experts. To ensure that, government has also joined hands with ITU-ABI researches. All this steps are supposed to make Bangladesh cyber resilient. The statistics of the web applications we have tested for this study indicates that these steps are all in their primary stage. Though our study is also in its initial stage and we need to analyze more data to come to a conclusion. Yet the present Bangladesh is not as cyber resilient as expected. The reason behind this is less awareness of people towards cyber security, developer's malpractice, and lack of proper cyber security framework. The vulnerabilities we have tested are very basic and should not exist if proper attention is paid. That's what makes our study crucial. Web applications of Bangladesh are vulnerable to basic SQL injection. The web developers need to take these vulnerabilities into serious considerations while building a web application. The government needs to make the people aware of the cyber threats and build a framework to be followed to ensure cyber security of the web applications. The web applications also need constant monitoring and update. The steps needed to fulfill the initial goals of forming a cyber resilient Bangladesh are not that difficult. They just need awareness of the people. Once we get the people to think about and including cyber security it won't be much far to achieve the goal of "Cyber resilient Bangladesh".

7. References

[1] T. Farah, D. Alam, M.A.Kabir, and T. Bhuiyan, "SQLi Penetration Testing of Financial Web Applications: Investigation of Bangladesh Region, " World Congress on Internet Security (WorldCIS 2015), dublin, Irland, 19-21 Oct. 2015.

[2] W.Halfond, G.J. Viegas, and A. Orso, "A classification of SQL-injection attacks and countermeasures," Proceedings of the IEEE International Symposium on Secure Software Engineering, 2006.

[3] Cyberwellness profile Bangladesh. [Online]. Available: https://www.itu.int/en/ITU-D/Cybersecurity/ Documents/ Country_Profiles/Bangladesh.pdf (Access date 7 Feb. 2016)

[4] Global Cybersecurity Index, 2015. [Online]. Available: http://knoema.com/GCSI2015/global-cybersecurity -index-2015?regionId=PW (Access date 7 Feb. 2016)

[5]2014 Results for Asia Pacific Region. [Online]. Available:http://www.itu.int/en/ITU-D/Cybersecurity/ Documents/GCI_2014_Results_for_Asia_Pacific.pdf (Access date 7 Feb. 2016)

[6] Global 2014 results. [online]. Available: http://www.itu.int/en/ITU-D/Cybersecurity/Documents /GCI_Global_2014_results.pdf (Access date 7 Feb. 2016)

[7] D. Alam, T. Farah, and M.A.Kabir, "Exploring the SQL injection vulnerabilities of .bd domain web applications," 3rd International Conference on Advances in Computing, Electronics and Communication (ACEC 2015), Zurich, Switzerland, 10-11 Oct. 2015, pp. 110 – 114.

[8] D. Alam, M.A.Kabir, T. Bhiuyan, and T. Farah, "A case study of SQL injection vulnearbilities assessment of .bd domain web applications," 4th International Conference on Cyber Security, Cyber Welfare, and Digital Forensic (CYBERSEC 2015), Jakarta, Indonesia, 29-31 Oct. 2015.

[9] D. Alam, T. Bhiuyan, M.A.Kabir, and T. Farah, "SQLi vunlerability in education sector websites of Bangladesh," 2nd International Conference on Information Security and Cyber Forensics (InfoSec2015), Cape Town, South Africa, 15-17 Nov. 2015.

[10] T. Farah, D. Alam, M. N. B. Ali, and M. A. Kabir, "Investigation of Bangladesh region based web applications: a case study of 64 based, local, and gobal SQLi vulnerability," IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE 2015), Dhaka, Bangladesh, 19-20 Dec. 2015.

[11] M. Mirjalili, A. Nowroozi, and M. Alidoosti,"A survey on web penetration test," International Journal in Advances in Computer Science, Los Alamitos, CA, 2014, vol. 3, no.6.

[12] U. Sachin, K. Mandeep, and G.K. Govinda, "Vunnerability assessment and penetration testing," International Journal of Computer and Communication Technology, April 2011, vol.3, issue 8, pp. 1-4.

[13] P. Mittal, A. Kamra, and S.K. Jena, “Technical guide to iinformation security testing and assesment,” NIST publication, 800,115, 2008.

[14] Certified Ethical Hacker (CEH) Module 12: Web Application Vulnerabilities, [online]. [Available]: http://www.infosecinstitute.com/courses/cehcertified ethical hacker.html (Access date 7 Feb. 2016).

[15] A. Tajpour, A. Kamra, and M.J.Z. Shooshtar, "Evaluation of SQL Injection Detection and Prevention Techniques," Second International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN 2010), 28-30 July 2010, pp. 216-221.

[16] R. Johari, P. Sharma, "A Survey on Web Application Vulnerabilities (SQLIA, XSS) Exploitation and Security Engine for SQL Injection," International Conference on Communication Systems and Network Technologies (CSNT 2012), 11-13 May 2012, pp. 453-458.

[17] V. Sunkari, C.V.G. Rao, "Preventing input type validation vulnerabilities using network based intrusion detection systems," International Conference on Contemporary Computing and Informatics (IC3I 2014), 27-29 Nov. 2014, pp.702-706.

[18] The Open Web Application Security Project (OWASP). [online]. Available: www.owasp.org/index.php/Mainpage (accesses 7 Feb. 2016).

[19] A. Razzaq, A. Hur, S. Shahbaz, M. Masood, and H.F. Ahmed, “Critical analysis on web application firewall solution,” 11th IEEE international symposium on autonomous decentralization systems (ISADS), Maxico city Mar. 6-8 2013, pp. 35-40.

[20] N. Teodoro, and C. Serrao,”Assessing the Portuguese Web applications security,” World Congress on Internet Security (WorldCIS), 21-23 Feb. 2011, pp. 21-26.

