1. INTRODUCTION

1.1 PROJECT SUMMARY

This project is solely aimed to protect website against malicious attacks & secure web application against web vulnerabilities using priWAF (web application firewall) in today's cyber world. It completely revolves around how to overcome the vulnerabilities of web applications and secure it as much as possible for the betterment of web application against attacker of cyber world.

The idea is to use an online portal for carrying out the required configuration related to priWAF (web application firewall) so that the client can better interact and know the needs independently through the web portal and more importantly, the underlying tasks can be carried out by the priWAF admin in a completely new and different manner which is easier and advantageous at its best.

It contains the details of clients and all configuration of WAF (web application firewall) by WAF admin side. Client can monitor attacks which will be done on their URLs or IPs after register their self on the WAF portal. WAF admin one who able to manage the whole details about client's personal information like URLs IPs Proxys Configuration of mod Security & signature for the specific customer's vulnerabilities which they found after scanning & crawling customer's URLs & IPs.

Thus, the main advantage of using this approach to manage the activities is that it collapses the cumbersomeness and complexity experienced while working with the manual WAF box on client's side web server & system and it also reduces rather significantly reduces the time needed to complete and manage the same tasks otherwise. Besides, the information becomes more available to the needed using this solution.

1.2 PURPOSE

The main purpose of implementing this web application firewall & WAF portal is to make as simple as possible for WAF admin to setup WAF box for multiple client's web server.

Develop a user friendly framework that will help WAF admin to setup WAF box for any client easily and maintain any client's details like Attack details / Monitor their web application and develop signature for the new vulnerabilities which found in scanning of the URLs.

Scope

The scope of this project includes the clients who want to protect their web application using priWAF (web application firewall) against vulnerabilities & wants to virtual patch those vulnerabilities. After whitelisting client's IP client able to access priWAF portal from his machine to register himself to be a part of priWAF registration process. After registration he can able to access his dashboard on his machine after allowed by priWAF admin.

Apart from this, priWAF admin has to update every client's details like URLs, IPs and all other information on the portal.

1.3 OBJECTIVE

1. Easier to setup WAF (web application firewall) for multi URLs / IPs.
2. Storage of client's details and manipulation of web application.
3. Analysis of client's vulnerabilities details by web application scanner.
4. Understand and implement the functional requirements of the users.
5. Application to a new concept architecture of WAF for multiusers.
6. Daily scan updates of new vulnerabilities and related to patching reports by the priWAF admin to the clients.

Thus, the whole project revolves around the main objective of how to ease to maintain virtual patching of web application using priWAF (web application firewall) and proxy server.

1.4 TECHNOLOGY AND LITERATURE REVIEW

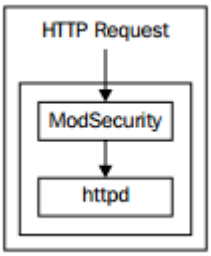
The core technology that has been used to patch any web application is Mod Security.
The technology used as the Back-end to build priWAF portal in this project is PHP.
The technology which was used to the whole database in this project is MySQL.

1.5.1 Introduction to Mod Security:

- Web application firewalls come to the rescue — and what else is better than mod_security.
- It is designed as an Apache module that adds intrusion-detection and prevention features to the Web server. In principle, it's similar to an IDS that analyses network traffic, but it works at the HTTP level, and really well, at that. This allows you to do things that are difficult in a classic IDS. This difference will become clearer when we examine several examples. The attack prevention feature stands between the client and server; if it finds a malicious payload, it can reject the request, performing any one of a number of built-in actions.

- Some of the features of mod_security are audit logging, access to any part of the request (including the body) and the response, a flexible regular expression-based rule engine, file-upload interception, real-time validation and also buffer-overflow protection.

1.5.2 Why Mod Security?



ModSecurity is a toolkit for real-time web application monitoring, logging, and access control. I like to think about it as an enabler: there are no hard rules telling you what to do; instead, it is up to you to choose your own path through the available features. That's why the title of this section asks what ModSecurity can do, not what it does.

The freedom to choose what to do is an essential part of ModSecurity's identity and goes very well with its open source nature. With full access to the source code, your freedom to choose extends to the ability to customize and extend the tool itself to make it fit your needs. It's not a matter of ideology, but of practicality. I simply don't want my tools to restrict what I can do.

Back on the topic of what ModSecurity can do, the following is a list of the most important usage scenarios:

- **Real-time application security monitoring and access control**

  At its core, ModSecurity gives you access to the HTTP traffic stream, in real-time, along with the ability to inspect it. This is enough for real-time security monitoring. There's an added dimension of what's possible through ModSecurity's persistent storage mechanism, which enables you to track system elements over time and

perform event correlation. You are able to reliably block, if you so wish, because ModSecurity uses full request and response buffering.

- **Virtual patching**

  Virtual patching is a concept of vulnerability mitigation in a separate layer, where you get to fix problems in applications without having to touch the applications themselves. Virtual patching is applicable to applications that use any communication protocol, but it is particularly useful with HTTP, because the traffic can generally be well understood by an intermediary device. ModSecurity excels at virtual patching because of its reliable blocking capabilities and the flexible rule language that can be adapted to any need. It is, by far, the activity that requires the least investment, is the easiest activity to perform, and the one that most organizations can benefit from straight away.

- **Full HTTP traffic logging**

  Web servers traditionally do very little when it comes to logging for security purposes. They log very little by default, and even with a lot of tweaking you are not able to get everything that you need. I have yet to encounter a web server that is able to log full transaction data. ModSecurity gives you that ability to log anything you need, including raw transaction data, which is essential for forensics. In addition, you get to choose which transactions are logged, which parts of a transaction are logged, and which parts are sanitized.

- **Continuous passive security assessment**

  Security assessment is largely seen as an active scheduled event, in which an independent team is sourced to try to perform a simulated attack. Continuous passive security assessment is a variation of real-time monitoring, where, instead of focusing on the behavior of the external parties, you focus on the behavior of the system itself. It's an early warning system of sorts that can detect traces of many abnormalities and security weaknesses before they are exploited.

- **Web application hardening**

  One of my favorite uses for ModSecurity is attack surface reduction, in which you selectively narrow down the HTTP features you are willing to accept (e.g., request methods, request headers, content types, etc.). ModSecurity can assist you in enforcing many similar restrictions, either directly, or through collaboration with other Apache modules. They all fall under web application hardening. For example, it is possible to fix many session management issues, as well as cross-site request forgery vulnerabilities.

- **Something small, yet very important to you**

  Real life often throws unusual demands to us, and that is when the flexibility of ModSecurity comes in handy where you need it the most. It may be a security need, but it may also be something completely different. For example, some people use ModSecurity as an XML web service router, combining its ability to parse XML and apply XPath expressions with its ability to proxy requests. Who knew?

1.5.3 Deployment Options

ModSecurity supports two deployment options: embedded and reverse proxy deployment. There is no one correct way to use them; choose an option based on what best suits your circumstances. There are advantages and disadvantages to both options:

- **Embedded**

  Because ModSecurity is an Apache module, you can add it to any compatible version of Apache. At the moment that means a reasonably recent Apache version from the 2.0.x branch, although a newer 2.2.x version is recommended. The embedded option is a great choice for those who already have their architecture laid out and don't want to change it. Embedded deployment is also the only option if you need to protect

hundreds of web servers. In such situations, it is impractical to build a separate proxybased security layer. Embedded ModSecurity not only does not introduce new points of failure, but it scales seamlessly as the underlying web infrastructure scales. The main challenge with embedded deployment is that server resources are shared between the web server and ModSecurity.

- **Reverse proxy**

Reverse proxies are effectively HTTP routers, designed to stand between web servers and their clients. When you install a dedicated Apache reverse proxy and add ModSecurity to it, you get a â€œproperâ€• network web application firewall, which you can use to protect any number of web servers on the same network. Many security practitioners prefer having a separate security layer. With it you get complete isolation from the systems you are protecting. On the performance front, a standalone ModSecurity will have resources dedicated to it, which means that you will be able to do more (i.e., have more complex rules). The main disadvantage of this approach is the new point of failure, which will need to be addressed with a high-availability setup of two or more reverse proxies.

1.5.4 Functionalities of Mod Security:

- Parsing: Security-conscientious parsers extract bits of each request and/or response and store them for use in the rules.
- Buffering: In a typical installation, both request and response bodies will be buffered so that the module usually sees complete requests (before they are passed to the application for processing), and complete responses (before they are sent to clients). Buffering is important, because it is the only way to provide reliable blocking.
- Logging: Allows you to record complete HTTP traffic, logging all response/request headers and bodies.

- Rule engine: Works on the data from the other components, to assess the transaction and take action, as necessary.

1.5.5 Introduction to PHP:

- PHP (recursive acronym for *PHP: Hypertext Preprocessor*) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML. PHP reduces the effort required on user's part and makes programming enjoyable.
- Instead of lots of commands to output HTML (as seen in C or Perl), PHP pages contain HTML with embedded code that does "something". The PHP code is enclosed in special start and end processing instructions <? Php and?> that allow you to jump into and out of "PHP mode."
- What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server, generating HTML which is then sent to the client. The client would receive the results of running that script, but would not know what the underlying code was. You can even configure your web server to process all your HTML files with PHP, and then there's really no way that users can tell what you have up your sleeve.
- The best thing in using PHP is that it is easy and simple for a beginner, but offers many advanced features for a professional programmer. Its advantage lies in the fact that one can easily catch the flow of programming in PHP and can use it to complete a particular task in comparatively less time.

There are three main areas where PHP scripts are used.

- **Server-side scripting**. This is the most traditional and main target field for PHP. You need three things to make this work. The PHP parser (CGI or server module), a web server and a web browser. You need to run the web server, with a connected PHP installation. You can access the PHP program output with a web browser, viewing the

PHP page through the server. All these can run on your local machine if you are just experimenting with PHP programming

- **Command line scripting**. You can make a PHP script to run it without any server or browser. You only need the PHP parser to use it this way. This type of usage is ideal for scripts regularly executed on a unix platform or Task Scheduler (on Windows). These scripts can also be used for simple text processing tasks.
- **Writing desktop applications**. PHP is probably not the very best language to create a desktop application with a graphical user interface, but if you know PHP very well, and would like to use some advanced PHP features in your client-side applications you can also use PHP-GTK to write such programs. You also have the ability to write cross-platform applications this way. PHP-GTK is an extension to PHP, not available in the main distribution.

1.5.6 Advantages of PHP

- The beauty of PHP lies in its simplicity. It is easy to understand and learn, especially for those with backgrounds in programming such as C, javascript and HTML. The language is similar to C and Perl so that anyone with a background in either C or Perl programming will feel comfortable using and understanding PHP. PHP also runs on just about every platform including most UNIX, Macs and Windows versions.

- PHP doesn't use a lot of the system resources so it runs fast and doesn't tend to slow other processes down. It is typically used as an Apache module, written in C, so it loads and executes quickly. It works well with other software and can be quite fast. PHP is also fairly stable and since it is open source, the PHP community works together to fix any bugs. The community offers technical support and continuously updates the code further expanding PHP's capabilities.

- PHP offers many levels of security to prevent malicious attacks. These security levels can be adjusted in the php.ini file.

- Another key advantage of PHP is its connective abilities. PHP uses a modular system of extensions to interface with a variety of libraries such as graphics, XML,

encryption, etc. In addition, programmers can extend PHP by writing their own extensions and compiling them into the executable or they can create their own executable and load it using PHP's dynamic loading mechanism.

- One of the strongest and most significant features in PHP is its support for a wide range of databases. Writing a database-enabled web page is incredibly simple using one of the database specific extensions (e.g., for mysql), or connect to any database supporting the Open Database Connection standard via the ODBC extension. Other databases may utilize CURL.

- PHP is most suited for the implementation of such systems due to its better technical compatibility according to the type of needs of such systems.

1.5.7 Introduction to MYSQL:

- A MYSQL is a web hosting database that is used to store web site information like blog posts or user information. A MYSQL database is the most popular type of relational database on the web today. This is partly because it is completely free but also very popular.
- The MYSQL database has become the world's most popular open source database because of its high performance, high reliability and ease of use. It is also the database of choice for a new generation of applications built on the LAMP stack (Linux, Apache, MYSQL, PHP / Perl / Python.) Many of the world's largest and fastest-growing organizations including Facebook, Google, Adobe, Alcatel Lucent and Zappos rely on MySQL to save time and money powering their high-volume Web sites, business-critical systems and packaged software.
- MySQL runs on more than 20 platforms including Linux, Windows, Mac OS, Solaris, IBM AIX, giving you the kind of flexibility that puts you in control. Whether you're new to database technology or an experienced developer or DBA, MySQL offers a comprehensive range of database tools, support, training and consulting services to make you successful.

1.5.8 Advantages of MYSQL

- **Open Source**: MYSQL is an open source database system which means that anyone can use it for free. Developers can amend its code to suit their requirements which means that MYSQL is highly customizable as well. Another edge of using MYSQL over other database systems is that; it is available widely in the market with no ownership cost.

- **Fast Development**: A lot of people around the globe are continuously developing new modules for integration with MYSQL. This means that it has a wider and faster development circle. Patches, upgrades and fixes are developed fast and become available in forums, blogs and developer sites on the internet.

- **Better for Small Businesses**: This relational database system is free so it reduces the cost of overall database solution for small businesses and companies. This database is relatively easy to learn and operate, so operational cost is reduced substantially which is again an important factor in classifying MYSQL as an applicable and practical tool for small businesses.

- **Cross Platform Operability**: MYSQL is easily installable and operable on different platforms including Windows, Linux, OS2 and Solaris. Cross platform operability makes it a favourable choice for development companies. MYSQL database system also contains APIs for integration with C, C++, PHP, Java, Perl, Python, Tcl, and Ruby etc. You can connect it easily with different development platforms so you can actually integrate applications developed in different OS and development platforms.

- **Security:** MYSQL as a relational database is secure as all access passwords are stored in an encrypted format restricting any unauthorised access to the system. It also encrypts the transactions so eavesdroppers and data harvest tools cannot replicate or regenerate the database transactions once they are processed.

- **Connectivity:** MYSQL clients can access this relational database through standard TCP/IP sockets, named pipes, UNIX sockets and many more. Standard ODBC 2.5 and above functions and commands are also supported in MYSQL.