IMPLEMENTATION OF X.509 CERTIFICATE AS A WEB CLIENT

" OLYNX SSL "

PROJECT REPORT

Submitted by

ANEESH K S : TJAOECS008

ATHULJITH A P : TJAOECS012

in partial fulfillment for the award of the degree

Of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

THEJUS ENGINEERING COLLEGE, VELLARAKKAD

UNIVERSITY OF CALICUT

MARCH 2018

Thejus Engineering College,
Vellarakkad, Thrissur- 680584
Computer Science & Engineering

BONAFIDE CERTIFICATE

Certified that this Project titled **"OLYNX SSL"** is the bonafide work of **ANEESH K S , ATHULJITH A P** of final year B.Tech in partial fulfillment of the requirement for the award of Bachelor of Technology Degree in Computer Science and Engineering awarded by the University of Calicut during the year 2017-2018 under my guidance.

Mrs. Soumya P
Project Guide

Dr. Saju P John
Head of Department

Thrissur
Date……

ACKNOWLEDGEMENT

The project on topic “OLYNX SSL” was taken as a part of the curriculum for the award of B.Tech degree in Computer Science and Engineering.

We sincerely express our gracefulness to Dr. K.Satheesh Kumar , Principal of Thejus Engineering College, Vellarakkad, and Dr. Saju P. John, Associate Professor and Head OfDepartment in Department of Computer Science, for encouraging and supporting us throughout the project.

We are also thankful to our project guide, Mrs. Soumya P. Assistant Professor in Computer Science & Engineering Dept. who through her expert supervision, encouragementand constructive criticism gave us constant supportamidst her busy schedule.

We acknowledgetheliberal help provided by the members of thecollegelibrary and the institutional staffs. We also thank our classmates for their support. Last but not the least, we thank God the almighty, for leading us straight to success of the project with his blessings.

ABSTRACT

"OLYNX SSL" proposes a secure approach based on the concept of SSL security ,An SSL certificate is necessary to create SSL connection. You would need to give all details about the identity of your website and your company as and when you choose to activate SSL on your web server. Following this, two cryptographic keys are created - a Private Key and a Public Key.

The next step is the submission of the CSR (Certificate Signing Request), which is a data file that contains your details as well as your Public Key. The CA (Certification Authority) would then validate your details. Following successful authentication of all details, you will be issued SSL certificate. The newly-issued SSL would be matched to your Private Key. From this point onwards, an encrypted link is established by your web server between your website and the customer's web browser.

On the apparent level, the presence of an SSL protocol and an encrypted session is indicated by the presence of the lock icon in the address bar. A click on the lock icon displays to a user/customer details about your SSL. It's to be remembered that SSL Certificates are issued to either companies or legally accountable individuals only after proper authentication.

An SSL Certificate comprises of your domain name, the name of your company and other things like your address, your city, your state and your country. It would also show the expiration date of the SSL plus details of the issuing CA. Whenever a browser initiates a connection with a SSL secured website, it will first retrieve the site's SSL Certificate to check if it's still valid. It's also verified that the CA is one that the browser trusts, and also that the certificate is being used by the website for which it has been issued. If any of these checks fail, a warning will be displayed to the user, indicating that the website is not secured by a valid SSL certificate. By using Olynx SSL client any website owner can have security with free of cost and it's easy to implement for every person.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 BACKGROUND

An SSL certificate is necessary to create SSL connection. You would need to give all details about the identity of your website and your company as and when you choose to activate SSL on your web server. Following this, two cryptographic keys are created - a Private Key and a Public Key.The next step is the submission of the CSR (Certificate Signing Request), which is a data file that contains your details as well as your Public Key. The CA (Certification Authority) would then validate your details. Following successful authentication of all details, you will be issued SSL certificate. The newly-issued SSL would be matched to your Private Key. From this point onwards, an encrypted link is established by your web server between your website and the customer's web browser.

On the apparent level, the presence of an SSL protocol and an encrypted session is indicated by the presence of the lock icon in the address bar. A click on the lock icon displays to a user/customer details about your SSL. It's to be remembered that SSL Certificates are issued to either companies or legally accountable individuals only after proper authentication.

An SSL Certificate comprises of your domain name, the name of your company and other things like your address, your city, your state and your country. It would also show the expiration date of the SSL plus details of the issuing CA. Whenever a browser initiates a connection with a SSL secured website , it will first retrieve the site's SSL Certificate to check if it's still valid. It's also verified that the CA is one that the browser trusts, and also that the certificate is being used by the website for which it has been issued. If any of these checks

fail, a warning will be displayed to the user, indicating that the website is not secured by a valid SSL certificate.

1.2 OBJECTIVE

SSL or TLS (Transport Layer Security) certificates are data files that bind a cryptographic key to the details of an organization. When SSL/TLS certificate is installed on a web server, it enables a secure connection between the web server and the browser that connects to it. The website's URL is prefixed with "https" instead of "http" and a padlock is shown on the address bar. If the website uses an extended validation (EV) certificate, then the browser may also show a green address bar.

1.3 Problem Statement

The internet has spawned new global business opportunities for enterprises conducting online commerce. However, that growth has also attracted fraudsters and cyber criminals who are ready to exploit any opportunity to steal consumer bank account numbers and card details. Any moderately skilled hacker can easily intercept and read the traffic unless the connection between a client (e.g. internet browser) and a web server is encrypted.

The SSL protocol is used by millions of online business to protect their customers, ensuring their online transactions remain confidential. A web page should use encryption when it expects users to submit confidential data, including personal information, passwords, or credit card details. All web browsers have the ability to interact with secured sites so long as the site's certificate is issued by a trusted CA.

2. SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

Since applications can communicate either with or without TLS (or SSL), it is necessary for the client to indicate to the server the setup of a TLS connection. One of the main ways of achieving this is to use a different port number for TLS connections, for example port 443 for HTTPS. Another mechanism is for the client to make a protocol-specific request to the server to switch the connection to TLS; for example, by making a STARTTLS request when using the mail and news protocols.

Once the client and server have agreed to use TLS, they negotiate a stateful connection by using a handshaking procedure.The protocols use a handshake with an asymmetric cipher to establish not only cipher settings but also a session-specific shared key with which further communication is encrypted using a symmetric cipher. During this handshake, the client and server agree on various parameters used to establish the connection's security

Fig 2.1.1 SSL security on website

2.2 LITERATURE SURVEY

SSL or TLS (Transport Layer Security) certificates are data files that bind a cryptographic key to the details of an organization. When SSL/TLS certificate is installed on a web server, it enables a secure connection between the web server and the browser that connects to it. The website's URL is prefixed with "https" instead of "http" and a padlock is shown on the address bar. If the website uses an extended validation (EV) certificate, then the browser may also show a green address bar.

What is SSL used for?

The SSL protocol is used by millions of online business to protect their customers, ensuring their online transactions remain confidential. A web page should use encryption when it expects users to submit confidential data, including personal information, passwords, or credit card details. All web browsers have the ability to interact with secured sites so long as the site's certificate is issued by a trusted CA.

Why do I need SSL certificate?

The internet has spawned new global business opportunities for enterprises conducting online commerce. However, that growth has also attracted fraudsters and cyber criminals who are ready to exploit any opportunity to steal consumer bank account numbers and card details. Any moderately skilled hacker can easily intercept and read the traffic unless the connection between a client (e.g. internet browser) and a web server is encrypted.

How Does SSL Work?

The following graphic explains how SSL Certificate works on a website. The process of how an 'SSL handshake' takes place is explained below:

- An end-user asks their browser to make a secure connection to a website (e.g.https://www.example.com)

- The browser obtains the IP address of the site from a DNS server then requests a secure connection to the website.
- To initiate this secure connection, the browser requests that the server identifies itself by sending a copy of its SSL certificate to the browser.
- The browser checks the certificate to ensure:
- That it is signed by a trusted CA
- That it is valid - that it has not expired or been revoked
- That it confirms to required security standards on key lengths and other items.
- That the domain listed on the certificate matches the domain that was requested by the user.
- When the browser confirms that the website can be trusted, it creates a symmetric session key which it encrypts with the public key in the website's certificate. The session key is then sent to the web server.
- The web server uses its private key to decrypt the symmetric session key.
- The server sends back an acknowledgement that is encrypted with the session key.

From now on, all data transmitted between the server and the browser is encrypted and secure.

How do I implement SSL on my website :

Implementing SSL for a website is quite easy! A typical installation of SSL certificate involves the following steps:

Step 1. Acquire SSL certificate :

To implement SSL/TLS security on your website, you need to get and install a certificate from a trusted CA. A trusted CA will have its root certificates

embedded in all major root store programs, meaning the certificate you purchase will be trusted by the internet browsers and mobile devices used by your website visitors.

You should also decide which type of certificate suits you best.

Single domain certificates allow you to secure one fully qualified domain name (FQDN).

Wildcard certificates secure a single domain and unlimited subdomains of that domain. For example, a wildcard certificate for '*.domain.com' could also be used to secure 'payments.domain.com', 'login.domain.com', 'anything-else.domain.com'

Multi-domain certificates allow website owners to secure multiple, distinct domains on a one certificate. For example, a single MDC can be used to secure domain-1.com, domain-2.com, domain-3.co.uk, domain-4.net and so on.

Extended Validation certificates provide the highest levels of security, trust and customer conversion for online businesses. Because of this, EV certificates contain a unique differentiator designed to clearly communicate the trustworthiness of the website to its visitors. Whenever somebody visits a website that uses an EV SSL, the address bar will turn green in major browsers such as Internet Explorer, Firefox and Chrome.

Step 2. Activate and install your SSL certificate :

When SSL certificate is purchased from a web host, its activation is taken care of by the web host. The administrator of the website can also activate the SSL through Web Host Manager (WHM) or cPanel. In the WHM dashboard select the SSL/TLS option and choose "Generate SSL Certificate and Signing Request". Next, generate your Private Key and fill out the form for Certificate Signing Request (CSR). Ensure that you enter your domain name in the box asking for

"Host to make cert for". You will need to send this CSR to your CA in order to purchase.

Olynx SSL offers detailed guides for installing certificates on various webservers too. See SSL Certificate Installation on Different Web Servers for a full list. The guides provides installation instructions for different software types such as Apache, Apache on Cobalt, BEA, C2Net Stronghold, Ensim, F5, Hsphere, IBM, Microsoft, Netscape / Sun, Novell, Plesk, SSL Accelerator, Website Pro, and Zeus.

Step 3. Update Website from HTTP to HTTPS :

Your website is now capable of HTTPS! You must now configure you website so that visitors who access this site get automatically directed to the "HTTPS" version. Search engine providers like Google are now offering SEO benefits to SSL pages, so the effort to serve all pages on your site over HTTPS is well worth it.

Who issues SSL Certificates :

A certificate authority or certification authority (CA) issues SSL certificates. On receiving an application, the CA verifies two factors: It confirms the legal identity of the enterprise/company seeking the certificate and whether the applicant controls the domain mentioned in the certificate. The issued SSL certificates are chained to a 'trusted root' certificate owned by the CA. Most popular internet browsers such as Firefox, Chrome, Internet Explorer, Microsoft Edge, and others have these root certificates embedded in their 'certificate store'. Only if a website certificate chains to a root in its certificate store will the browser allow a trusted and secure https connection. If a website certificate does not chain to a root then the browser will display a warning that the connection is not trusted.

What details are included in a SSL certificate :

SSL Certificates will contain details of whom the certificate has been issued to. This includes the domain name or common name, serial number; the

details of the issuer; the period of validity - issue date and expiry date; SHA Fingerprints; subject public key algorithm, subject's public key; certificate signature algorithm, certificate signature value. Other important details such as the type of certificate, SSL/TLS version, Perfect Forward Secrecy status, and cipher suite details are included. Organization validated and extended validation certificates also contain verified identity information about the owner of the website, including organization name, address, city, state and country.

2.3 PROPOSED SYSTEM

SSL or TLS (Transport Layer Security) certificates are data files that bind a cryptographic key to the details of an organization. When SSL/TLS certificate is installed on a web server, it enables a secure connection between the web server and the browser that connects to it. The website's URL is prefixed with "https" instead of "http" and a padlock is shown on the address bar. If the website uses an extended validation (EV) certificate, then the browser may also show a green address bar.

To implement SSL/TLS security on your website, you need to get and install a certificate from a trusted CA. A trusted CA will have its root certificates embedded in all major root store programs, meaning the certificate you purchase will be trusted by the internet browsers and mobile devices used by your website visitors.

Proposed system contains eight entities:

1. LINUX ENCRYPTION CERT'S USAGE
2. LINUX API & INTEGRATION
3. FINDING ACCOUNT ID
4. OLYNX SSL USAGE
5. IMPLEMENTATION OF CERTIFICATE

6. USER SIDE VERIFICATION
7. SERVER SIDE IMPLEMENTATION
8. CERTIICATE INSTALLATION

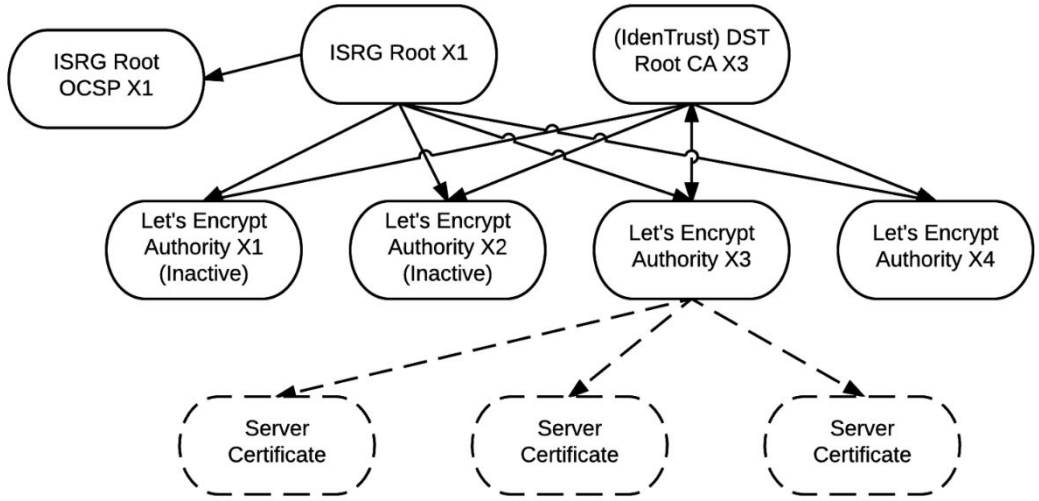

Fig 2.3 Linux Encryption Certificates

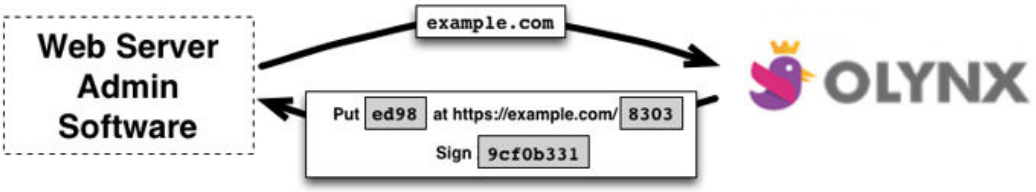

Fig 2.4 Olynx SSL Port Addressing & Certificate Implementation

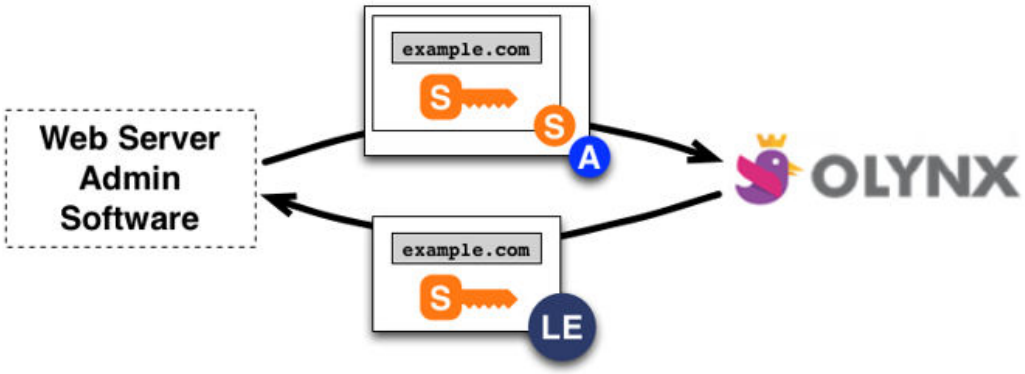

Fig 2.5 Olynx SSL Port Addressing & Certificate Implementation

2.3.1 Linux Encryption Certificates (ACME)

The ACME protocol is the cornerstone of how Linux Encryption Keys works. It is currently a draft standard and not yet a finalized RFC. As the protocol specification evolves over time Linux Encryption Keys will implement updated versions of ACME. When doing so, security will be our primary concern, followed closely by backwards compatibility.

Currently Implemented ACME Version

We currently have the following API endpoints. They do not implement any one fixed draft of the ACME specification as they evolved alongside the draft protocol document. Please see our divergences documentation to compare their implementation to the current ACME draft.

- [Production] https://acme-v01.api.letsencrypt.org/directory
- [Staging] https://acme-staging.api.letsencrypt.org/directory

New Backwards-Compatible ACME Features

From time to time Linux Encryption Keys may implement new backwards-compatible features for existing API endpoints. Typically new

backwards-compatible features are introduced because we've decided to implement a part of the ACME spec that we hadn't implemented before.

When new features are introduced to existing API endpoints, the features will always be clearly specified in a public ACME specification and will not break properly implemented clients.

ACME Security Fixes :

If we become aware of a serious security issue with the ACME protocol (rather than simply our implementation of it) we may be forced to make compatibility-breaking changes to our API endpoints, or to cease operation of existing endpoints and introduce new ones.

ACME has been reviewed by many parties and used successfully in production, but there is always the possibility of undiscovered vulnerabilities. Systems administrators should maintain the ability to deploy timely updates to their ACME clients in response to such vulnerabilities.

New Versions of ACME with Breaking Changes

When we feel it's important to implement new versions of ACME containing breaking changes we'll do so by introducing new API endpoints and maintaining them in parallel with the endpoints for older versions. After making the new version available, we will communicate a deprecation timeline to all users well in advance.This is not going to happen very often since breaking compatibility is so burdensome even if there is plenty of time to transition. We will, however, be doing this once the IETF finishes standardizing ACME. We currently implement a pre-IETF-standardization version of ACME and we feel it's important to be using a formalized standard if possible..

2.3.2 Integrating with Linux API & cert keys

Both OLYNX SSL and the Web PKI will continue to evolve over time. You should make sure you have the ability to easily update all services that use OLYNX SSL. If you're also deploying clients that rely on OLYNX SSL certificates, make especially sure that those clients receive regular updates.

In the future, these things are likely to change:

the root and intermediate certificates from which we issue

the hash algorithms we use when signing certificates

the types of keys and key strength checks for which we are willing to sign end-entity certificatesand the ACME protocol

We will always aim to give as much advance notice as possible for such changes, though if a serious security flaw is found in some component we may need to make changes on a very short term or immediately. For intermediate changes in particular, you should not hardcode the intermediate to use, but should use the Link: rel="up" header from the ACME protocol, since intermediates are likely to change.

Similarly, we're likely to change the URL of the terms of service (ToS) as we update it. Avoid hardcoding the ToS URL and instead rely on the Link: rel="terms-of-service" header to determine which ToS URL to use.

You will also want a way to keep your TLS configuration up-to-date as new attacks are found on cipher suites or protocol versions.

Get Updates

To receive low-volume updates about important changes like the ones described above, subscribe to our API Announcements group. This is useful for both client developers and hosting providers.

For higher-volume updates about maintenances and outages, visit our status page and hit Subscribe in the upper right. This is most useful for hosting providers.

Also, make sure you use a valid email address for your ACME account. We will use that email to send you expiration notices and communicate about any issues specific to your account.

Who is the Subscriber :

Our CPS and Subscriber Agreement indicate that the Subscriber is whoever holds the private key for a certificate. For hosting providers, that's the provider, not the provider's customer. If you're writing software that people deploy themselves, that's whoever is deploying the software.

The contact email provided when creating accounts (aka registrations) should go to the Subscriber. We'll send email to that address to warn of expiring certs, and notify about changes to our privacy policy. If you're a hosting provider, those notifications should go to you rather than a customer. Ideally, set up a mailing list or alias so that multiple people can respond to notifications, in case you are on vacation.

The upshot of this is that, if you are a hosting provider, you do not need to send us your customers' email addresses or get them to agree to our Subscriber Agreement. You can simply issue certificates for the domains you control and start using them.

One Account or Many?

In ACME, it's possible to create one account and use it for all authorizations and issuances, or create one account per customer. This flexibility may be valuable. For instance, some hosting providers may want to use one account per customer, and store the account keys in different contexts, so that an account key compromise doesn't allow issuance for all of their customers.

However, for most larger hosting providers we recommend using a single account and guarding the corresponding account key well. This makes it easier to identify certificates belonging to the same entity, easier to keep contact information up-to-date, and easier to provide rate limits adjustments if needed. We will be unable to effectively adjust rate limits if many different accounts are used.

Multi-domain (SAN) Certificates :

Our issuance policy allows for up to 100 names per certificate. Whether you use a separate certificate for every hostname, or group together many hostnames on a small number of certificates, is up to you.

Using separate certificates per hostname means fewer moving parts are required to logically add and remove domains as they are provisioned and retired. Separate certificates also minimize certificate size, which can speed up HTTPS handshakes on low-bandwidth networks.

On the other hand, using large certificates with many hostnames allows you to manage fewer certificates overall. If you need to support older clients like Windows XP that do not support TLS Server Name Indication (SNI), you'll need a unique IP address for every certificate, so putting more names on each certificate reduces the number of IP addresses you'll need.

For most deployments both choices offer the same security.

Storing and Reusing Certificates and Keys

A big part of OLYNX SSL's value is that it enables automatic issuance as part of provisioning a new website. However, if you have infrastructure that may repeatedly create new frontends for the same website, those frontends should first try to use a certificate and private key from durable storage, and only issue a new one if no certificate is available, or all existing certificates are expired.

For OLYNX SSL, this helps us provide services efficiently to as many people as possible. For you, this ensures that you are able to deploy your website whenever you need to, regardless of the state of OLYNX SSL.

As an example, many sites are starting to use Docker to provision new frontend instances as needed. If you set up your Docker containers to issue when they start up, and you don't store your certificates and keys durably, you are likely to hit rate limits if you bring up too many instances at once. In the worst case, if you have to destroy and re-create all of your instances at once, you may wind up in a situation where none of your instances is able to get a certificate, and your site is broken for several days until the rate limit expires. This type of problem isn't unique to rate limits, though. If OLYNX SSL is unavailable for any reason when you need to bring up your frontends, you would have the same problem.

Note that some deployment philosophies state that crypto keys should never leave the physical machine on which they were generated. This model can work fine with OLYNX SSL, so long as you make sure that the machines and their data are long-lived, and you manage rate limits carefully.

Picking a Challenge Type :

If you're using the http-01 ACME challenge, you will need to provision the challenge response to each of your frontends before notifying OLYNX SSL that you're ready to fulfill the challenge. If you have a large number of frontends, this may be challenging. In that case, using the dns-01 challenge is likely to be easier. Of course, if you have many geographically distributed DNS responders, you have to make sure the TXT record is available on each responder.

Additionally, when using the dns-01 challenge, make sure to clean up old TXT records so the response to OLYNX SSL's query doesn't get too big.

If you want to use the http-01 challenge anyhow, you may want to take advantage of HTTP redirects. You can set up each of your frontends to redirect /.well-known/acme-validation/XYZ to validation-server.example.com/XYZ for all XYZ. This delegates responsibility for issuance to validation-server, so you should protect that server well.

Central Validation Servers :

Related to the above two points, it may make sense, if you have a lot of frontends, to use a smaller subset of servers to manage issuance. This makes it easier to use redirects for http-01 validation, and provides a place to store certificates and keys durably.

Implement OCSP Stapling :

Many browsers will fetch OCSP from OLYNX SSL when they load your site. This is a performance and privacy problem. Ideally, connections to your site should not wait for a secondary connection to OLYNX SSL. Also, OCSP requests tell OLYNX SSL which sites people are visiting. We have a good privacy policy and do not record individually identifying details from OCSP requests, we'd rather not even receive the data in the first place. Additionally, we anticipate our bandwidth costs for serving OCSP every time a browser visits a OLYNX SSL site for the first time will be a big part of our infrastructure expense.

By turning on OCSP Stapling, you can improve the performance of your website, provide better privacy protections for your users, and help OLYNX SSL efficiently serve as many people as possible.

Additionally, OCSP Stapling forms the basis for two other technologies you may want to use in the future: Must Staple, which provides for revocation that actually

works, and Certificate Transparency proofs (which we will be delivering by OCSP, and which must be stapled for Chrome to recognize them).

OLYNX SSL IPs :

OLYNX SSL will validate from a number of different IP addresses in the future, and will not announce which ones in advance. You should make sure your validation server is available to all IPs.

Some people who are issuing for non-HTTP services want to avoid exposing port 80 to anyone except OLYNX SSL's validation server. If you're in that category you may want to use the DNS challenge instead.

Supported Key Algorithms :

OLYNX SSL accepts RSA keys from 2048 to 4096 bits in length, and P-256 and P-384 ECDSA keys. That's true for both account keys and certificate keys. You can't reuse an account key as a certificate key.

Our recommendation is to serve a dual-cert config, offering an RSA certificate by default, and a (much smaller) ECDSA certificate to those clients that indicate support.

HTTPS by default :

For hosting providers, our recommendation is to automatically issue certificates and configure HTTPS for all hostnames you control, and to offer a user-configurable setting for whether to redirect HTTP URLs to their HTTPS equivalents. We recommend that for existing accounts, the setting be disabled by default, but for new accounts, the setting be enabled by default.

Reasoning: Existing websites are likely to include some HTTP sub resources (scripts, CSS, and images). If those sites are automatically redirected to their HTTPS versions, browsers will block some of those sub resources due to Mixed

Content Blocking. This can break functionality on the site. However, someone who creates a new site and finds that it redirects to HTTPS will most likely include only HTTPS sub resources, because if they try to include an HTTP sub resource they will notice immediately that it doesn't work.

OLYNX recommend allowing customers to set an HTTP Strict-Transport-Security (HSTS) header with a default max-age of sixty days. However, this setting should be accompanied by a warning that if the customer needs to move to a hosting provider that doesn't offer HTTPS, the cached HSTS setting in browsers will make their site unavailable. Also, both customer and hosting provider should be aware that the HSTS header will make certificate errors into hard failures. For instance, while people can usually click through a browser warning about a name mismatch or expired certificate, browsers do not allow such a click through for hostnames with an active HSTS header.

When to Renew

We recommend renewing certificates automatically when they have a third of their total lifetime left. For OLYNX SSL's current 90-day certificates, that means renewing 30 days before expiration.

If you are issuing for more than 10,000 hostnames, we also recommend automated renewal in small runs, rather than batching up renewals into large chunks. This reduces risk: If OLYNX SSL has an outage at the time you need to renew, or there is a temporary failure in your renewal systems, it will only affect a few of your certificates, rather than all of them. It also makes our capacity planning easier.

You may want to bulk-issue certificates for all of your domains to get started quickly, which is fine. You can then spread out renewal times by doing a one-time process of renewing some certificates 1 day ahead of when you would normally renew, some of them 2 days ahead, and so on.

If you offer client software that automatically configures a periodic batch job, please make sure to run at a randomized hour and minute during the day, rather than always running at a specific time. This ensures that OLYNX SSL doesn't receive arbitrary spikes of traffic at the top of the hour. Since OLYNX SSL needs to provision capacity to meet peak load, reducing traffic spikes can help keep our costs down.

Retrying failures :

Renewal failure should not be treated as a fatal error. You should implement graceful retry logic in your issuing services using an exponential backoff pattern, maxing out at once per day per certificate. For instance, a reasonable backoff schedule would be: 1st retry after one minute, 2nd retry after ten minutes, third retry after 100 minutes, 4th and subsequent retries after one day. You should of course have a way for administrators to request early retries on a per-domain or global basis.

Backoffs on retry means that your issuance software should keep track of failures as well as successes, and check if there was a recent failure before attempting a fresh issuance. There's no point in attempting issuance hundreds of times per hour, since repeated failures are likely to be persistent.

2.3.3Finding Account ID on LINUX

When reporting issues it can be useful to provide your Linux Encryption account ID. Most of the time, the process of creating an account is handled automatically by the ACME client software you use to talk to Linux Encryption, and you may have multiple accounts configured if you run ACME clients on multiple servers. The account id should be in the form of an URL to the linux encryption websitehttps://acme-v01.api.letsencrypt.org/acme/reg/12345678. You can also provide just the digits at the end of that URL as a short hand. If you're using

Certbot, you can find your account ID by looking at the “uri” field in /etc/letsencrypt/accounts/acme-v01.api.letsencrypt.org/directory/*/regr.json.

If you’re using another ACME client, the instructions will be client-dependent. Check your logs for URLs of the form described above. If your ACME client does not record the account ID, you can retrieve it by submitting a new registration request with the same key. See the ACME spec for more details. You can also find the numeric form of your ID in the Boulder-Requester header in the response to each POST your ACME client makes.

2.3.3 Olynx SSL installation

When SSL certificate is generated from a web Olynx SSL, its activation is taken care of by the web host. The administrator of the website can also activate the SSL through Web Host Manager (WHM) or cPanel. In the WHM dashboard select the SSL/TLS option and choose "Generate SSL Certificate and Signing Request". Next, generate your Private Key and fill out the form for Certificate Signing Request (CSR). Ensure that you enter your domain name in the box asking for "Host to make cert for". You will need to send this CSR to your CA in order to purchase a certificate there have to spend some money, so by using Oynx SSL the cost for purchasing an SSL certificate can be reduced.

3. PROJECT DESCRIPTION

3.1 PROBLEM DEFINITION

The SSL protocol is used by millions of online business to protect their customers, ensuring their online transactions remain confidential. A web page should use encryption when it expects users to submit confidential data, including personal information, passwords, or credit card details. All web browsers have the ability to interact with secured sites so long as the site's certificate is issued by a trusted CA. When SSL certificate is purchased from a web host, its activation is taken care of by the web host. The administrator of the website can also activate the SSL through Web Host Manager (WHM) or cPanel. In the WHM dashboard select the SSL/TLS option and choose "Generate SSL Certificate and Signing Request". Next, generate your Private Key and fill out the form for Certificate Signing Request (CSR). Ensure that you enter your domain name in the box asking for "Host to make cert for". we need to send this CSR to our CA in order to purchase a certificate. Actually those generation and implementation process are very difficult to execute and it will also takes more time. For normal people these generation and implementation of SSL certificates are much difficult.

3.2 OVERVIEW

An SSL certificate is necessary to create SSL connection. You would need to give all details about the identity of your website and your company as and when you choose to activate SSL on your web server. Following this, two cryptographic keys are created - a Private Key and a Public Key.The next step is the submission of the CSR (Certificate Signing Request), which is a data file that contains your details as well as your Public Key. The CA (Certification Authority) would then validate your details. Following successful authentication of all details, you will be issued SSL certificate. The newly-issued SSL would be matched to your Private Key. From this point onwards, an encrypted link is

established by your web server between your website and the customer's web browser.

On the apparent level, the presence of an SSL protocol and an encrypted session is indicated by the presence of the lock icon in the address bar. A click on the lock icon displays to a user/customer details about your SSL. It's to be remembered that SSL Certificates are issued to either companies or legally accountable individuals only after proper authentication.

An SSL Certificate comprises of your domain name, the name of your company and other things like your address, your city, your state and your country. It would also show the expiration date of the SSL plus details of the issuing CA. Whenever a browser initiates a connection with a SSL secured website , it will first retrieve the site's SSL Certificate to check if it's still valid. It's also verified that the CA is one that the browser trusts, and also that the certificate is being used by the website for which it has been issued. If any of these checks fail, a warning will be displayed to the user, indicating that the website is not secured by a valid SSL certificate.

3.3 PHASES

There are 5different phases:

- Identification of Public Key
- Domain explanation & Verification
- Hash code signature generation
- Domain Details Validation
- Issuing & Implementation of SSL certificate

3.3.1 1Identification of Public Key

First, the ACME protocol requires you register a public key and contact informationso you can sign all the requests you make to the API. In this step, you need toput in an email and a public key. The javascript for this section then

converts the public key to a JSON Web Key ([JWK] for more details can use this website address ;(https://tools.ietf.org/html/rfc7517)).

NOTE:currently only RSA 2048 and 4096 bit public keys are accepted by Linux Encrypt.

So if you paste it in this public key:

```
-----BEGIN PUBLIC KEY-----
MIICIjANBgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEA5aok6d72rkrGpOPAICSS
3JPrA0tbVs3mYPWmG7c5tGEY+w1slyI+3V64NsLw8p9YqNLyX/YDsnmkOUMUx6Bu
vx43daBrl//wz3hIOvidXyV4z65Nbrlto9qtLpfi+9lbEEYt2PLhr+KjguqjqOQj
qi2PgqdITGG+BZkU8xIrPzZCR/UPBotV/dGBj9vO1whTGlzpkihvXLf4rEFoJoEE
eOPMtqbxUp1KS41EgX2xFav9JHPVI1hm66K0eqlJrBl407j3xRNlekl4xorwfCkA
xC7xclofg3JZ7RIhv3DdaNe07IZ0QYup9dDufIcCKruAgu0hwYMwDHmZNrrWxMia
GQwagxs61mla6f7c1bvYY92PhfgpkQAN99MXdaTtvBbzDuY018QP+TVzzVH/hpjK
aFx4JlYkcVGqbYamUiP7il4Hldqp6Mm65IH/8nxuZFrN4tJ5VyMeWeZ5sKBBrXZE
1Je8524COYnvljGnaFAVaDRhAcTSEykveY8jx/r6MB95LkWcue7FXIQyX0D3/2lU
KTu/wrBCmhriqNa4FHcccLMyQkiMbs8mEoldNCwYDxvF5lYc19UDlleE855lME00
E/ogStmazzFrNWCzEJ+Pa9JVlTQonKRgWqi+9cWwV+AMd+s2F3wO+H2tlexe8pLo
Vw/42S44tHz4VuZuhpZvn3kCAwEAAQ==
-----END PUBLIC KEY-----
```

This step converts it to this JWK:

```
{
  "alg": "RS256",
  "jwk": {
    "e": "AQAB",
    "kty": "RSA",
    "n": "5aok6d72rkrGpOPAICSS3JPrA0tbVs3mYPWmG7c5tGEY-w1slyI-
3V64NsLw8p9YqNLyX_YDsnmkOUMUx6Buvx43daBrl__wz3hIOvidXyV4z65Nbrlto9qt
Lpfi-9lbEEYt2PLhr-KjguqjqOQjqi2PgqdITGG-
BZkU8xIrPzZCR_UPBotV_dGBj9vO1whTGlzpkihvXLf4rEFoJoEEeOPMtqbxUp1KS41E
gX2xFav9JHPVI1hm66K0eqlJrBl407j3xRNlekl4xorwfCkAxC7xclofg3JZ7RIhv3Dd
aNe07IZ0QYup9dDufIcCKruAgu0hwYMwDHmZNrrWxMiaGQwagxs61mla6f7c1bvYY92P
hfgpkQAN99MXdaTtvBbzDuY018QP-
TVzzVH_hpjKaFx4JlYkcVGqbYamUiP7il4Hldqp6Mm65IH_8nxuZFrN4tJ5VyMeWeZ5s
KBBrXZE1Je8524COYnvljGnaFAVaDRhAcTSEykveY8jx_r6MB95LkWcue7FXIQyX0D3_
2lUKTu_wrBCmhriqNa4FHcccLMyQkiMbs8mEoldNCwYDxvF5lYc19UDlleE855lME00E
```

```
_ogStmazzFrNWCzEJ-Pa9JVlTQonKRgWqi-9cWwV-AMd-s2F3wO-
H2tlexe8pLoVw_42S44tHz4VuZuhpZvn3k"
  }
} …
```

3.3.2 Domain explanation & Verification (CSR)

The user need to specify the domains you want certificates for. That's done through a certificate signing request , for generating an CSR certificate follow ([CSR](https://en.wikipedia.org/wiki/Certificate_signing_request)).The javascript in this section uses the [ASN1.js](https://lapo.it/asn1js/) libraryto parse the CSR and read the domains. NOTE: the private key for the domain certcannot be the same as your account private key, according to ACME.

3.3.3 Hash code signature generation

Linux Encrypt API that you want to register and get certsfor some domains. These requests must be signed with your account private key, sothis steps compiles the request payloads that need signatures. You need to ask forchallenges for each domain, so if you want both `example.com` and `www.example.com`, youneed to make two new-authz calls.

Here's the list of requests that need to be made to the API:

- `/acme/new-reg` - Register the account public key (discarded if already registered)
- `/acme/new-authz` - Asks for challenges for the domain for which you want a cert.
- `/acme/new-authz` - (...needs to be called for each domain)
- `/acme/new-cert` - Asking for your CSR to be signed.

NOTE: Each request also requires an anti-replay nonce, so the javascript get those by making ajax requests to the `/directory` endpoint.

For each request the payload must be signed, and since this website doesn't ask for your private keys, you must copy-and-paste the signature commands into yourterminal.

These commands are structured like this:

- PRIV_KEY=./account.key; \ #set the location of your account private key (change this location if different)
- echo -n "<request_payload_data>" | \ #pipe the payload into open ssl
- openssldgst -sha256 -hex -sign $PRIV_KEY #sign the payload using your private key and output hex …

Once these signatures are pasted back into the inputs, the javascript makes the ajax requests to the above endpoints for `new-reg` and each `new-authz`. If theaccount public key has already been registered the `new-reg` response is a 409Conflict, which is ignored.

3.3.4 Domain Verification

The response for each `/new-authz` has some challenges you need perform to prove you own the domain. The challenge that this website chooses is "http-01",which requires that you host a specific file at a specific location. So, foreach domain, this step shows you the file you need to host and the url you need to host it at.

After the file is being hosted, you need to tell Linux Encrypt to check theverify the challenge for that domain. That request must also be signed sothere's one more signature that must be performed. The reason why this wasn't included in step 3 is because the payload contains something in the response of

`/new-authz`.

There's two options this website offers as copy-and-paste commands: python andfile-based. The python command is a mini server you can copy-and-paste into yourserver's command line (NOTE: this needs sudo permissions!). The file-base option just lists the url where the challenge will check and the file contents that thefile needs to contain. It's up to you to figure out how to make that happen.

When you confirm that you're hosting the files, an ajax request is made to the challenge url to tell Linux Encrypt to verify the domain. Once this is done forall the domains in your CSR, an ajax request is made to `/new-cert` with thepreviously signed payload from step 3.

3.3.5 Issuing & Implementation of SSL certificate

The response from `/new-cert` should be your new certificate! Congrats! Thisstep prints the certificate and also prints the intermediate certificate youneed to chain this certificate to the root certificate. When SSL certificate is generated from a Olynx SSL, its activation is taken care of by the web host. The administrator of the website can also activate the SSL through Web Host Manager (WHM) or cPanel. Go just implement it on the web host that you have.

3.4 ALOGRITHM

SHA -256 Algorithm

A hashing algorithm is a cryptographic algorithm that can be used to provide data integrity and authentication. They are also typically used in password based systems to avoid the need to store plaintext passwords.

There are three common uses of cryptographic hashes:

- **Storage of passwords:**Rather than storing a user's password, a system will typically store the hash of the password instead. When a user enters their password, the hash is then computed and compared with the stored hash. If the hash matches, due to the collision resistance property of hashing algorithms, it implies that the passwords match.
- **Integrity checking**:The sender can hash a file before sending to the recipient. The recipient will then hash the file received and check the hashes match. This can also be used for the storage of files, to ensure files have not been corrupted or modified

Steps followed by the SHA-256 Algorithm:

- **Step 1:** Append Padding Bits….

Message is "padded" with a 1 and as many 0's as necessary to bring the message length to 64 bits fewer than an even multiple of 512.

- **Step 2:** Append Length....

64 bits are appended to the end of the padded message. These bits hold the binary format of 64 bits indicating the length of the original message.

- **Step 3:** Prepare Processing Functions….

SHA1 requires 80 processing functions defined as:

f(t;B,C,D) = (B AND C) OR ((NOT B) AND D)

(0 <= t <= 19)

f(t;B,C,D) = B XOR C XOR D

(20 <= t <= 39)

f(t;B,C,D) = (B AND C) OR (B AND D) OR (C AND D)
(40 <= t <=59)

f(t;B,C,D) = B XOR C XOR D

(60 <= t <= 79)

- **Step 4:** Prepare Processing Constants....

  SHA1 requires 80 processing constant words defined as:

  K(t) = 0x5A827999 (0 <= t <= 19)

  K(t) = 0x6ED9EBA1 (20 <= t <= 39)

  K(t) = 0x8F1BBCDC (40 <= t <= 59)

  K(t) = 0xCA62C1D6 (60 <= t <= 79)

- **Step 5:** Initialize Buffers….

  SHA1 requires 160 bits or 5 buffers of words (32 bits):

  H0 = 0x67452301

  H1 = 0xEFCDAB89

  H2 = 0x98BADCFE

  H3 = 0x10325476

  H4 = 0xC3D2E1F0

- **Step 6:** Processing Message in 512-bit blocks (L blocks in total message)…. This is the main task of SHA-256 algorithm which loops through the padded and appended message in 512-bit blocks.

  Input and predefined functions:

  M[1, 2, ..., L]: Blocks of the padded and appended message f(0;B,C,D), f(1,B,C,D), ..., f(79,B,C,D): 80 Processing Functions K(0), K(1), ..., K(79): 80 Processing Constant Words

  H0, H1, H2, H3, H4, H5: 5 Word buffers with initial values

4. SYSTEM SPECIFICATION

4.1 HARDWARE REQUIREMENTS

- System : Pentium IV 2.4 GHZ / ANY
- Hard Disk : min 20 GB
- Display : 15 VGA Colour/ ANY
- Ram : min 512 MB

4.2 SOFTWARE REQUIREMENTS

- Operating system : Linux / windows / ANY OTHER
- Front End : HTML , CSS , JAVASCRIPT
- Server : apache-tomcat-5.5
- Backend : JAVASCRIPT
- SSH client : Putty SSH
- Client for SSL : Open SSL

5. SOFTWARE DESCRIPTION

5.1 FRONT END –HTML,CSS,JAVASCRIPT

HTML :

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web.[3] Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

The Standard Generalized Markup Language (SGML) was invented as a solution to these problems. By focussing on the logical elements in a document, the recipient of information is freed from the often inappropriate choices of the originator. Users can resize their windows to make optimal use of their screens and printing software can layout documents to match local paper sizes. Other benefits from using SGML include the ability to protect the investment of information providers in the face of short lived proprietary formats, and the availability of tools for authoring and format conversion. SGML is a meta format that allows one to specify a wide range of document formats. The hypertext markup language HTML was developed as a simple non-proprietary delivery format for global hypertext. HTML+ is a set of modular extensions to HTML and has been developed in response to a growing understanding of the needs of information providers. These extensions include text flow around floating figures, fill-out forms, tables and mathematical equations.

Browsers are programs for viewing HTML documents and are now available for most platforms. The best known are the Mosaic family of browsers from the U.S.

National Center for Supercomputing Applications, with browsers for X11, Windows, and the Macintosh. Non-graphical browsers are available for VT100 terminals etc, for example Lynx and Emacs-W3. The appearance of a document will vary from one browser to the next according to the capabilities of each system and the user's preferences.

Currently, most documents are created with conventional text editors. The effort needed to type the markup elements can be reduced by using post-processing tools to ensure that the resulting document complies with the HTML document type definition (DTD). Authors can also use common word processing packages and apply filters to convert to HTML. Filters are available for Frame Maker, LaTeX and Microsoft Word. Other people are using standard SGML tools for document authoring and conversion from proprietary SGML based document formats.

CSS :

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language.[1] Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.[2]

CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colors, and fonts.[3] This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share

formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. It can also display the web page differently depending on the screen size or viewing device. Readers can also specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author specified.

Changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in the CSS file they use, rather than by changing markup in the documents.

The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities (or weights) are calculated and assigned to rules, so that the results are predictable.

JAVASCRIPT :

JavaScript is a multi-paradigm, dynamic language with types and operators, standard built-in objects, and methods. Its syntax is based on the Java and C languages — many structures from those languages apply to JavaScript as well. JavaScript supports object-oriented programming with object prototypes, instead of classes (see more about prototypical inheritance and ES2015 classes). JavaScript also supports functional programming — functions are objects, giving functions the capacity to hold executable code and be passed around like any other object.

5.2 BACK END - JAVASCRIPT

Javascript technology is both a programming language and a platform. The Javascript programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Object oriented
- Distributed
- Multithreaded
- Dynamic
- Architecture neutral
- Portable
- High performance
- Robust
- Secure

In the Java Programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes – the machine language of the Java Virtual Machine (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.

An overview of the software development process:

Because the Java VM is available on many different operating systems, the same .class files are capable of running on Microsoft Windows, the Solaris™ Operating System (Solaris OS), Linux, or Mac OS .This include various tasks such as finding performance bottlenecks and recompiling (to native code) frequently usedsections of code. Through the Java VM, the same application is capable of running on multiple platforms.

Java Platform: A platform is the hardware or software environment in which a program runs. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The Java Virtual Machine
- The Java Application Programming Interface (API)

The API is a large collection of ready-made software components that provide many useful capabilities. It is grouped into libraries of related classes and interfaces; these libraries are known as packages. The next section, What Can Java Technology Can Do highlights some of the functionality provided by the API.

The API and Java Virtual Machine insulate the program from the underlying hardware.As a platform-independent environment, the Java platform can be a bit slower than native code. However, advances in compiler and virtual machine technologies are bringing performance close to that of native code without threatening portability.

Java platform gives the following features:

- Development Tools: The development tools provide everything you'll need for compiling, running, monitoring, debugging, and documenting your applications. As a new developer, the main tools you'll be using are the javac compiler, the java launcher, and the javadoc documentation tool.
- Application Programming Interface (API): The API provides the core functionality of the Java programming language. It offers a wide array of useful classes ready for use in your own applications. It spans everything from basic objects, to networking and security, to XML generation and database access, and more. The core API is very large.
- Deployment Technologies: The JDK software provides standard mechanisms such as the Java Web Start software and Java Plug-In software for deploying your applications to end users.

- User Interface Toolkits: The Swing and Java 2D toolkits make it possible to create sophisticated Graphical User Interfaces (GUIs).
- Integration Libraries: Integration libraries such as the Java IDL API, JDBC™ API, Java Naming and Directory Interface™ ("J.N.D.I.") API, Java RMI, and Java Remote Method Invocation over Internet Inter-ORB Protocol Technology (Java RMI-IIOP Technology) enable database access and manipulation of remote objects.

Java Technology Will Help To Do The Following:

- Get started quickly: Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- Write less code: Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program written in C++.
- Write better code: The Java programming language encourages good coding practices, and automatic garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeansTM component architecture, and its wide-ranging, easily extendible API let you reuse existing, tested code and introduce fewer bugs.
- Develop programs more quickly: The Java programming language is simpler than C++, and as such, your development time could be up to twice as fast when writing in it. Your programs will also require fewer lines of code.
- Avoid platform dependencies: You can keep your program portable by avoiding the use of libraries written in other languages.
- Write once, run anywhere: Because applications written in the Java programming language are compiled into machine-independent bytecodes, they run consistently on any Java platform.

- Distribute software more easily: With Java Web Start software, users will be able to launch your applications with a single click of the mouse. An automatic version check at startup ensures that users are always up to date with the latest version of your software.

5.3 PUTTY SSH

PuTTY SSH is a free and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port. The name "PuTTY" has no official meaning. PuTTY was originally written for Microsoft Windows, but it has been ported to various other operating systems. Official ports are available for some Unix-like platforms, with work-in-progress ports to Classic Mac OS and macOS, and unofficial ports have been contributed to platforms such as Symbian,Windows Mobile and Windows Phone.

PuTTY supports many variations on the secure remote terminal, and provides user control over the SSH encryption key and protocol version, alternate ciphers such as 3DES, Arcfour, Blowfish, DES, and Public-key authentication. It also can emulate control sequences from xterm, VT102 or ECMA-48 terminal emulation, and allows local, remote, or dynamic port forwarding with SSH (including X11 forwarding). The network communication layer supports IPv6, and the SSH protocol supports the zlib@openssh.com delayed compression scheme. It can also be used with local serial port connections.

5.4 OPEN SSL

OpenSSL is a software library for applications that secure communications over computer networks against eavesdropping or need to identify the party at the other end. It is widely used in internet web servers, serving a majority of all web sites. OpenSSL contains an open-source implementation of the SSL and TLS protocols. The core library, written in the C programming language, implements basic cryptographic functions and provides various utility

functions. Wrappers allowing the use of the OpenSSL library in a variety of computer languages are available.

Versions are available for most Unix and Unix-like operating systems (including Solaris, Linux, macOS, QNX, and the various open-source BSD operating systems), OpenVMS and Microsoft Windows. IBM provides a port for the System i (OS/400) and there by its can simply generate Certificate

6. SOURCE CODE

INDEX.HTML

```
/*
THIS PROJECT HAS BEEN MADE BY Mr. ATHULJITH AP ,
ON BEHALF OF HIS FINAL YEAR MAIN PROJECT .
THIS WEBSITE CAN BE USED TO GENERATE OPEN SSL
BASED SSL CERTIFICATES . THERE BY USERS CAN GET
ALMOST FREE OF COST CERTIFICATES FOR THEIR WEBSITE
ON A PERSONAL USE MANNER .
ALSO THE WEBSITE SASS AND LESS CONTENTS ARE MADE BY Mr. ANEESH KS
TOGETHER WITH ATHULJITH . FOR COMPLETENESS OF THE WEBSITE HE DID
AN AMAZING JOB ON FRONT END.
*/
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">
<meta name="description" content="Generate your free SSL certificate
with OLYNX SSL">
<meta name="author" content="Athuljith">
<link rel="icon" href="aj-project/img/favicon.png" type="image/x-
icon">
<title>SSL Automation by Olynx SSL
</title>
<link rel="stylesheet" href="aj-project/css/athuljith.css">
<style>.loader{position:fixed;left:0;top:0;width:100%;height:100%;ba
ckground-color:#f5f8fa;z-index:9998;text-align:center}.plane-
container{position:absolute;top:50%;left:50%}
</style>
<script src="js/asn1js/int10.js"></script>
<script src="js/asn1js/base64.js"></script>
<script src="js/asn1js/hex.js"></script>
<script src="js/asn1js/asn1.js"></script>
</head>
<body>
<div id="loader" class="loader">
</div>
<div id="app" class="athuljith-loading">
```

```
<div class="mini-nav nav-offcanvas nav-offcanvas-desktop"><nav
class="mainnav navbar navbar-default justify-content-between">
</nav>
</div>
<main class="single">
    <section class="text-white has-overlay" data-bg-
possition="center" style="background: url()">
<div class="container">
<div class="p-t-b-80  text-center"><br/><br/>
<p class="subtitle">Web client &amp; Security</p><h1 class="s-
48">Getting Started With Olynx SSL</h1>
<br/><br/>
</div>
</div>
<div class="overlay" data-start="#333A43" data-opacity=".8">
</div></section>
<div class="content-wrapper">
<div class="container">
<div class="col-xl-8 mx-md-auto ">

    <h2 id="digest_error" class="error" style="display:none;">
        ERROR: Your browser is not compatible with this website
(this website
        needs WebCryptoAPI's crypto.subtle.digest()). Please upgrade
to a modern
        browser (Firefox, Chrome, Safari, IE 11+).
    </h2><br/>
    <center><h2>LET'S ANALYSE THE GENERATED PUBLIC
KEY</h2></center><br/><br/>

    <div class="field">
        <input id="email" type="text" class="form-control form-
control-lg" placeholder="what's your e-mail id ? "/>
    </div>
    <br/>
    <div class="field">
        <div class="help-content">
            USE OPEN SSL TO GENERATE ACCOUNT KEY PAIR:<br/>
            <ol>
                <li>
                    MAKE AN ACCOUNT PRIVATE KEY USING :<br/>
                    <code>openssl genrsa 4096 > account.key</code>
```

```
                </li>
                <li>
                    PRINT YOUR KEY ON SHELL:<br/>
                    <code>openssl rsa -in account.key -pubout</code>
                </li>
                <li>
                    LET'S PASTE THE PUBLIC KEY INTO THE BOX.<br/>
                </li>
            </ol>
        </div>
        <label for="pubkey">YOUR PUBLIC KEY:</label><br/>
        <textarea id="pubkey" class="form-control form-control-lg"
placeholder="-----BEGIN PUBLIC KEY----- ..."></textarea>
    </div><br/><center>

    <div class="field">
        <input id="validate_account" class="btn-primary btn-lg btn
blue" type="submit" value="VALIDATE YOUR DETAILS WITH PUBLIC KEY"/>
        <span id="validate_account_status"
style="display:none;"></span>
    </div>
</center>
    <hr/>
<br/>
    <center><h2>EXPLAIN ABOUT YOUR DOMAIN'S</h2></center><br/>

    <div class="field">
        <div class="help-content">
            CREATE YOUR CERTIFICATE SIGNING REQUEST WITH:<br/>
            <ol>
                <li>
                    IF YOU DO NOT HAVE ANY TLS PRIVATE KEY:<br/>
                    <code>openssl genrsa 4096 > domain.key</code>
                </li>
                <li>
                    MAKE CSR FOR THE DOMAINS YOU WANT TO USE:</br>
                    Linux:
                    <pre>
#change "/etc/ssl/openssl.cnf" as needed:
#  Debian: /etc/ssl/openssl.cnf
#  RHEL and CentOS: /etc/pki/tls/openssl.cnf
#  Mac OSX: /System/Library/OpenSSL/openssl.cnf
```

```
openssl req -new -sha256 -key domain.key -subj "/" \
  -reqexts SAN -config &lt;(cat /etc/ssl/openssl.cnf \
  &lt;(printf
"\n[SAN]\nsubjectAltName=DNS:athuljith.com,DNS:www.athuljith.com"))
</pre>
                </li>
                <li>
                    LET'S PASTE THE CSR INTO THE BOX.<br/>
                </li>
            </ol>
        </div>
        <label for="csr">CERTIFICATE SIGNING REQUEST:</label><br/>
        <textarea id="csr" class="form-control form-control-lg"
placeholder="-----BEGIN CERTIFICATE REQUEST----- ..."></textarea>
    </div><br/><center>

    <div class="field">
        <input id="validate_csr" class="btn-primary btn-lg btn blue"
type="submit" value="EXPLAIN ABOUT YOUR DOMAIN DETAILS"/>
        <span id="validate_csr_status" style="display:none;"></span>
    </div>
</center>
    <hr/>
<br/>
    <center>
    <h2>GETTING START WITH SIGNATURE GENERATION<span
id="step3_pending"></span></h2></center>
    <div id="step3" style="display:none;">


        <div class="field">
            <div class="help-content">
                GENERATE NEEDED SIGNATURES YOU WANT:<br/>
                <ol>
                    <li>
                        COPY AND PASTE EACH COMMAND SHOWN BELOW
                    </li>
                    <li>
                        AFTER THAT COPY AND PASTE EACH COMMANDS
OUTPUT YOU GOT .
                    </li>
```

```
                </ol>
            </div>
            <label>RUN THIS HEX-CODE GENERATOR'S IN YOUR
SHELL:</label><br/>
            <span id="step3_commands">
            </span>
            <span id="signing_template" style="display:none;">
                <input class="form-control form-control-lg"
type="text" value="echo ..." readonly/><br/>
                <input id="challenge_sig_foobar_com" class="form-
control form-control-lg" type="text" placeholder='HEX CODE OUTPUT
HERE (e.g. "(stdin)= f2cf67e4...")'/><br/>
            </span>
        </div><br/><center>

        <div class="field">
            <input id="validate_initial_sigs" class="btn-primary
btn-lg btn blue" type="submit" value="VALIDATE HEX CODE WITH A
CLICK"/>
            <span id="validate_initial_sigs_status"
style="display:none;"></span>
        </div></center>
    </div>

    <hr/>
<br/>
    <center>
    <h2>VERIFY YOUR DOMAIN NAME'S WITH "OLYNX"<span
id="step4_pending"></span></h2></center>
    <div id="step4" style="display:none;">


            <div id="challenge_domains">
            </div>

            <div id="challenge_template"
style="display:none;"><br/><br/>
                <center> <h3>Your Domain: <span
class="domain">foobar.com</span></h3></center><br/><br/>

                <div class="field"><div style="display:none">
```

```
                    <label class="howto_sign_label help">(how do I
do this?)</label>
                    <input class="howto_sign help-checkbox"
type="checkbox"/></div>
                    <div class="help-content">
                        GENERATE NEEDED SIGNATURES YOU
WANT:<br/><br/>
                        <ol>
                            <li>
                                LET'S GO TO THE ACCOUNT.KEY PATH
WHERE YOU SAVED
                            </li>
                            <li>
                                COPY AND PASTE EACH COMMANDS
                            </li>
                            <li>
                                WE NEED THE HEX-CODED OUTPUT TO BE
PASTED HERE.
                            </li>
                        </ol>
                    </div>
                    <label>RUN THIS HEX-CODE GENERATOR IN YOUR
SHELL:</label><br/>
                    <span class="step4_commands">
                    </span>
                </div>

                <div class="field tabs">
                    <input type="radio" checked/>
                    <label>Option 1 - python server</label>
                    <input type="radio">
                    <label>Option 2 - file-based</label>

                    <br/>

                    <div class="tab"><div style="display:none">
                        <label class="howto_serve_label help">(how
do I do this?)</label>
                        <input class="howto_serve help-checkbox"
type="checkbox"/></div><br/>
                        <div class="help-content">
```

```
                            LET'S SERVE THE RESPONSE WITH YOUR
DOMAIN:<br/><br/>
                            <ol>
                                <li>
                                    SUDO PERMISSION NEEDED ON SSH
ACCESS NOW:<br/>
                                    <code class="ssh">ssh
ubuntu@foobar.com</code>
                                </li>
                                <li>
                                    PORT 80 RUNNING WEBSERVER SHOULD
BE STOP NOW.<br/>
                                    <code>sudo service nginx
stop</code> &lt;-- example for nginx<br/><br/>
                                    <code>sudo apachectl -k
graceful-stop</code> &lt;-- example for apache
                                </li>
                                <li>
                                    LET'S COPY AND PASTE THE COMMAND
IN YOUR TERMINAL
                                </li>
                                <li>
                                    THIS LINK WILL HELP YOU TO
VERIFY YOUR PATH ON DOMAIN:<br/>
                                    <a target="_blank"></a>
                                </li>
                                <li>
                                    CLICK ON THE FINAL BUTTON BELOW
                                </li>
                            </ol>
                        </div>
                        <label>LET'S RUN THIS ON <span
class="domain">foobar.com</span>'s SERVER:</label><br/>
                        <textarea class="form-control form-control-
lg" readonly></textarea><br/><center>
                        <div>
                            <input class="btn-primary btn-lg btn
blue" type="submit" value="I HAVE COMPLATED THIS FOR foobar.com"/>
                            <span style="display:none;"></span>
                        </div></center>
                    </div>
    <br/><br/>
```

```
                    <div class="tab"><div style="display:none">
                        <label class="howto_file_label help">(how do
I do this?)</label>
                        <input class="howto_file help-checkbox"
type="checkbox"/></div>
                        <div class="help-content">
                            LET'S UPLOAD THE FILE WITH YOUR
DOMAIN:<br/><br/>
                            <ol>
                                <li>
                                    SSH ACCESS NEEDED WITH SUDO
PERMISSIONS:<br/>
                                    <pre class="ssh">ssh
ubuntu@foobar.com</pre>
                                </li>
                                <li>
                                    LET'S MAKE THE ".well-
known/acme-challenge/" DIRECTORY:<br/>
                                    <pre class="wwwdir">mkdir -p
/path/to/www/.well-known/acme-challenge/</pre>
                                </li>
                                <li>
                                    WEB SERVERS CONFIG SHOULD
CONTAIN THAT STATIC FOLDER:<br/>
                                    <pre
class="nginx_location">server {...</pre>
                                </li>
                                <li>
                                    MAKE A FILE PATH INCLUDE NEEDED
CONTENTS:<br/>
                                    <pre class="file_cmd">echo
...</pre>
                                </li>
                                <li>
                                    LET'S MAKE SURE THAT YOUR UPLOAD
WORKS WELL:<br/>
                                    <a target="_blank"></a>
                                </li>
                                <li>
                                    CLICK ON THE FINAL BUTTON WITH
THE PYTHON SERVER ABOVE IF THE FILE BASED CREATED ONECE.
                                </li>
```

```
                            </ol>
                        </div>
                        <label>ON THIS URL LINK:</label><br/>
                        <input type="text" class="form-control form-
control-lg file_url"  value="" readonly/>
                        <div>
                            <label>SHARE THIS CONTENT:</label>
                            <input  type="text" class="form-control
form-control-lg file_data"  value="" readonly/>
                        </div><br/><center>
                        <div>
                            <input  type="submit" class="btn-primary
btn-lg btn blue file_submit" value="I HAVE UPLOADED THIS ON
foobar.com"/>
                            <span style="display:none;"></span>
                        </div></center>
                    </div>
                </div>
            </div>
        </div>

    <hr/>
<br/><center>
    <h2>HOORRAYY !!! INSTALL YOUR SSL CERTIFICATE <span
id="step5_pending"></span></h2></center>
    <div id="step5" style="display:none;">
        <div><center>
            WELL DONE ... AFTER ALL SUCCESFULL COMMENTS AND
VERIFICATION YOU HAVE RECEIVED AND " OLYNX SSL CERTIFICATE " LET'S
UPLOAD AND SECURE YOUR WEBSITE WITH HIGHLY SECURED SSL CERTIFICATE
.</center>
        </div>
<br/>
        <div class="field">
           <center>
            <label for="csr">SIGNED CERFTIFICATE FROM " OLYNX SSL "
: CRT</label></center><br/>
            <textarea id="crt" class="form-control form-control-lg"
readonly></textarea>
        </div>
        <br/>
        <div class="field"><center>
```

```
            <label for="csr">INTERMEDIATE CERTIFICATE OR
CABUNDLE</label></center><br/>
<textarea class="form-control form-control-lg" readonly>
-----BEGIN CERTIFICATE-----
MIIEkjCCA3qgAwIBAgIQCgFBQgAAAVOFc2oLheynCDANBgkqhkiG9w0BAQsFADA/
MSQwIgYDVQQKExtEaWdpdGFsIFNpZ25hdHVyZSBUcnVzdCBDby4xFzAVBgNVBAMT
DkRTVCBSb290IENBIFgzMB4XDTE2MDMxNzE2NDA0NloXDTIxMDMxNzE2NDA0Nlow
SjELMAkGA1UEBhMCVVMxFjAUBgNVBAoTDUxldCdzIEVuY3J5cHQxIzAhBgNVBAMT
GkxldCdzIEVuY3J5cHQgQXV0aG9yaXR5IFgzMIIBIjANBgkqhkiG9w0BAQEFAAOC
AQ8AMIIBCgKCAQEAnNMM8FrlLke3cl03g7NoYzDq1zUmGSXhvb418XCSL7e4S0EF
q6meNQhY7LEqxGiHC6PjdeTm86dicbp5gWAf15Gan/PQeGdxyGkOlZHP/uaZ6WA8
SMx+yk13EiSdRxta67nsHjcAHJyse6cF6s5K671B5TaYucv9bTyWaN8jKkKQDIZ0
Z8h/pZq4UmEUEz9l6YKHy9v6Dlb2honzhT+Xhq+w3Brvaw2VFn3EK6BlspkENnWA
a6xK8xuQSXgvopZPKiAlKQTGdMDQMc2PMTiVFrqoM7hD8bEfwzB/onkxEz0tNvjj
/PIzark5McWvxI0NHWQWM6r6hCm21AvA2H3DkwIDAQABo4IBfTCCAXkwEgYDVR0T
AQH/BAgwBgEB/wIBADAOBgNVHQ8BAf8EBAMCAYYwfwYIKwYBBQUHAQEEczBxMDIG
CCsGAQUFBzABhiZodHRwOi8vaXNyZy50cnVzdGlkLm9jc3AuaWRlbnRydXN0LmNv
bTA7BggrBgEFBQcwAoYvaHR0cDovL2FwcHMuaWRlbnRydXN0LmNvbS9yb290cy9k
c3Ryb290Y2F4My5wN2MwHwYDVR0jBBgwFoAUxKexpHsscfrb4UuQdf/EFWCFiRAw
VAYDVR0gBE0wSzAIBgZngQwBAgEwPwYLKwYBBAGC3xMBAQEwMDAuBggrBgEFBQcC
ARYiaHR0cDovL2Nwcy5yb290LXgxLmxldHNlbmNyeXB0Lm9yZzA8BgNVHR8ENTAz
MDGgL6AthitodHRwOi8vY3JsLmlkZW50cnVzdC5jb20vRFNUUk9PVENBWDNDUkwu
Y3JsMB0GA1UdDgQWBBSoSmpjBH3duubRObemRWXv86jsoTANBgkqhkiG9w0BAQsF
AAOCAQEA3TPXEfNjWDjdGBX7CVW+dla5cEilaUcne8IkCJLxWh9KEik3JHRRHGJo
uM2VcGfl96S8TihRzZvoroed6ti6WqEBmtzw3Wodatg+VyOeph4EYpr/1wXKtx8/
wApIvJSwtmVi4MFU5aMqrSDE6ea73Mj2tcMyo5jMd6jmeWUHK8so/joWUoHOUgwu
X4Po1QYz+3dszkDqMp4fklxBwXRsW10KXzPMTZ+sOPAveyxindmjkW8lGy+QsRlG
PfZ+G6Z6h7mjem0Y+iWlkYcV4PIWL1iwBi8saCbGS5jN2p8M+X+Q7UNKEkROb3N6
KOqkqm57TH2H3eDJAkSnh6/DNFu0Qg==
-----END CERTIFICATE-----
</textarea>
        </div>
<br/><center>
        <div class="field">
            <form
action="https://www.ssllabs.com/ssltest/analyze.html"
target="_blank">
                <input id="ssltest_domain" type="hidden" name="d"
value="foobar.com">
                <input class="btn-primary btn-lg btn blue"
type="submit" value="Test your Olynx SSL install"/>
            </form>
```

```
        </div></center><br/><br/>
    </div>
<footer>
    <center>COPYRIGHT 2018. "OLYNX SSL PROJECT" .</center>
</footer>
</div>
</div>
</div></main>
<script src="aj-project/aneesh/aneesh.js"></script>
<script src="js/index.js"></script>
</body>
</html>
```

7. IMPLEMENTATION RESULTS & PERFORMANCE EVALUATION

We have made the project to generate free of cost ssl certiicates , and there by it will use make custom server certificates for the clients website .

main features :

users need not to be registered

free od cost ssl certificates

steps involved :

1. user should tell about his/her public key for particular server
2. have to explain about which domains are to be protected , [using csr]
3. verification of users web server identity
4. verification of domain names to be certified
5. installation of generated certificate

The SSL protocol is used by millions of online business to protect their customers, ensuring their online transactions remain confidential. A web page should use encryption when it expects users to submit confidential data, including personal information, passwords, or credit card details. All web browsers have the ability to interact with secured sites so long as the site's certificate is issued by a trusted CA.When SSL certificate is purchased from a web host, its activation is taken care of by the web host. The administrator of the website can also activate the SSL through Web Host Manager (WHM) or cPanel. In the WHM dashboard select the SSL/TLS option and choose "Generate SSL Certificate and Signing Request". Next, generate your Private Key and fill out the form for Certificate Signing Request (CSR). Ensure that you enter your domain name in the box asking for "Host to make cert for". we need to send this CSR to our CA in order to purchase a certificate.Actuallythose generation and implementation process are very difficult to execute and it will also takes more time. For normal people these generation and implementation of SSL certificates are much difficult.

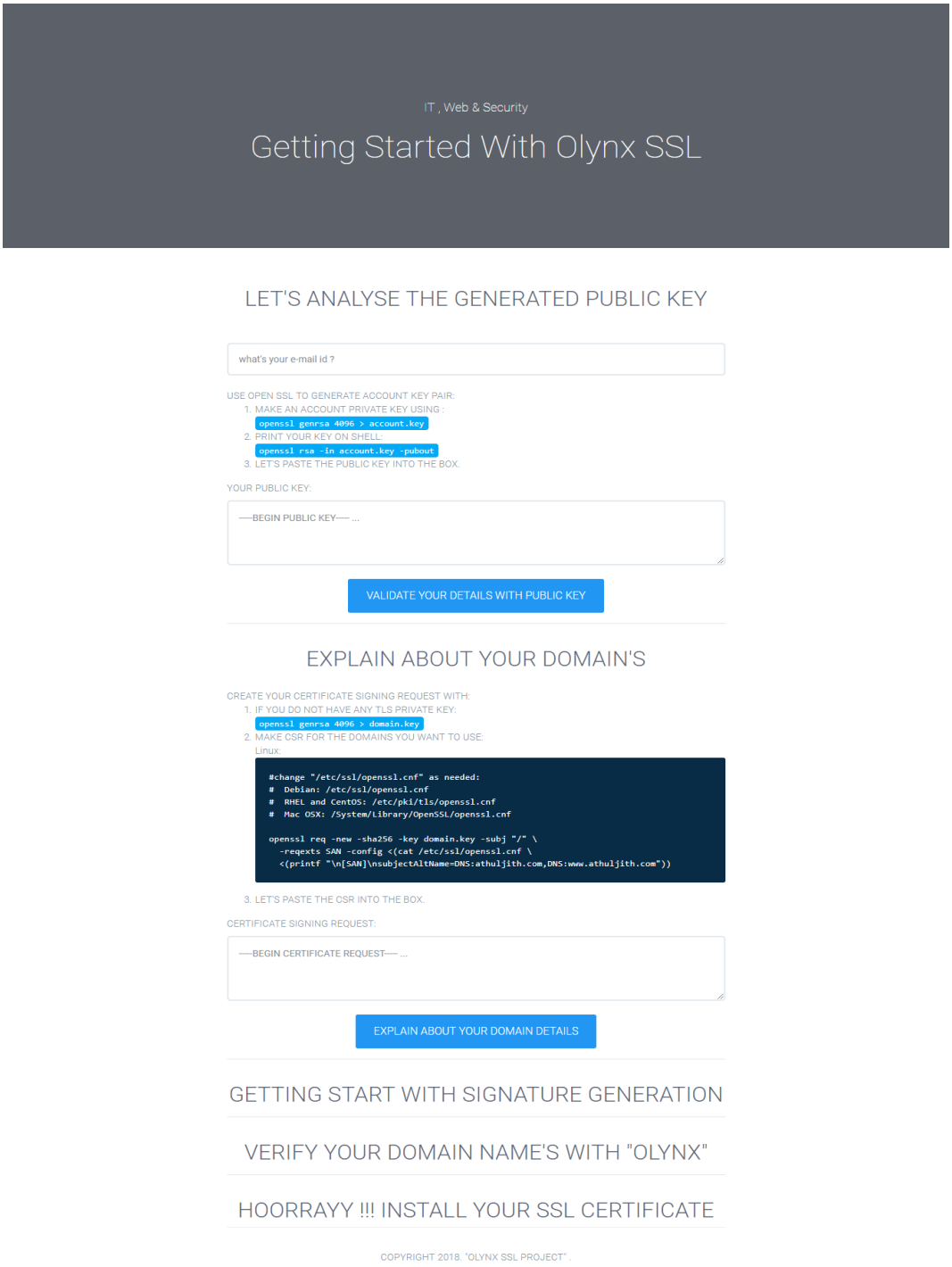

Fig 2.6 Olynx SSL WEBSITE

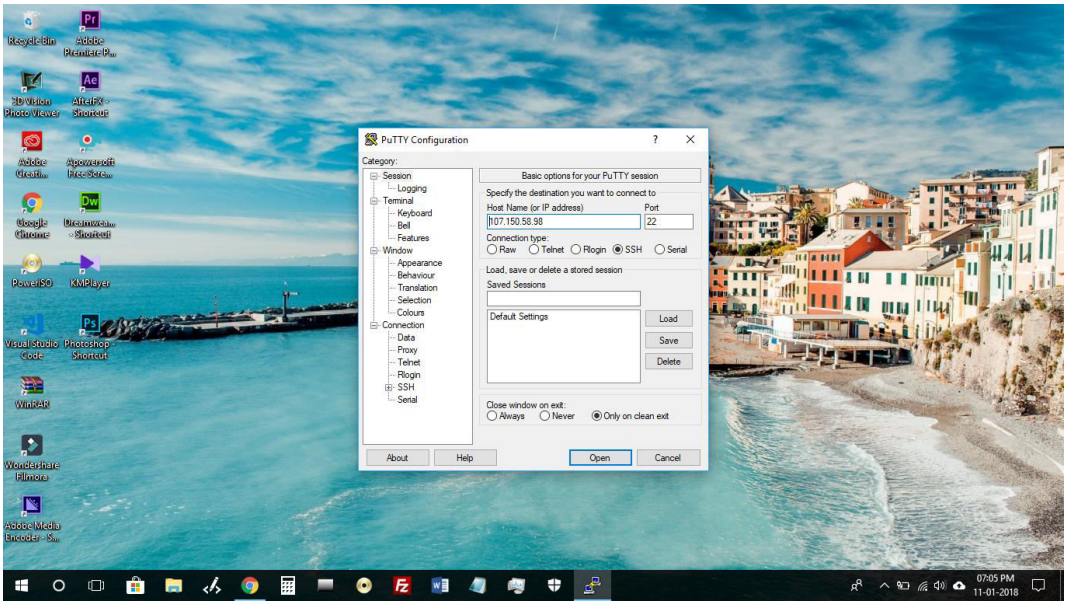

Fig 2.7 Client login using Putty SSH

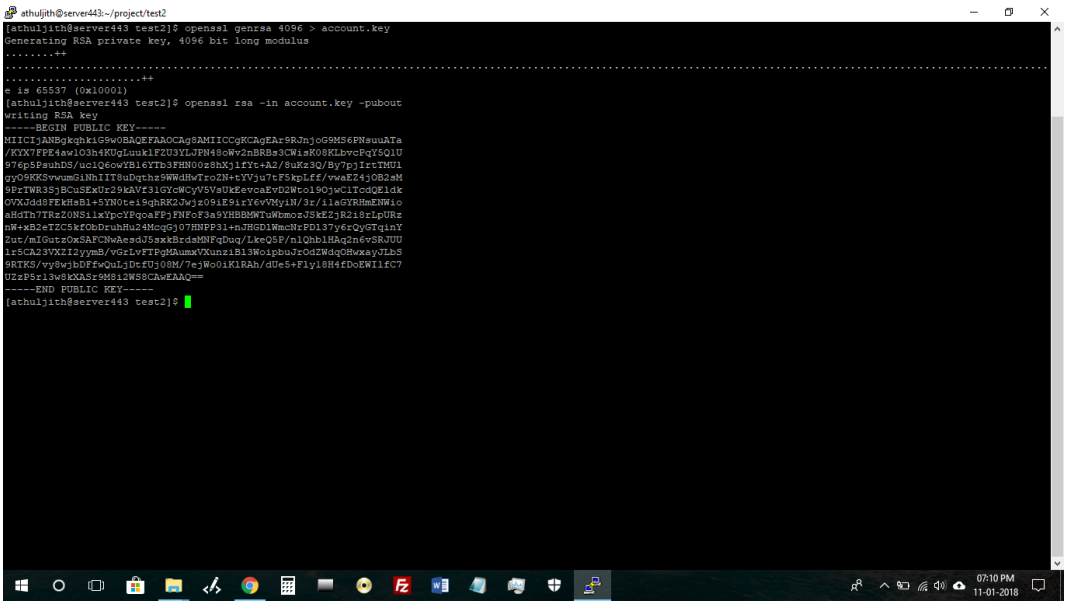

Fig 2.8 Generation of Public key on server

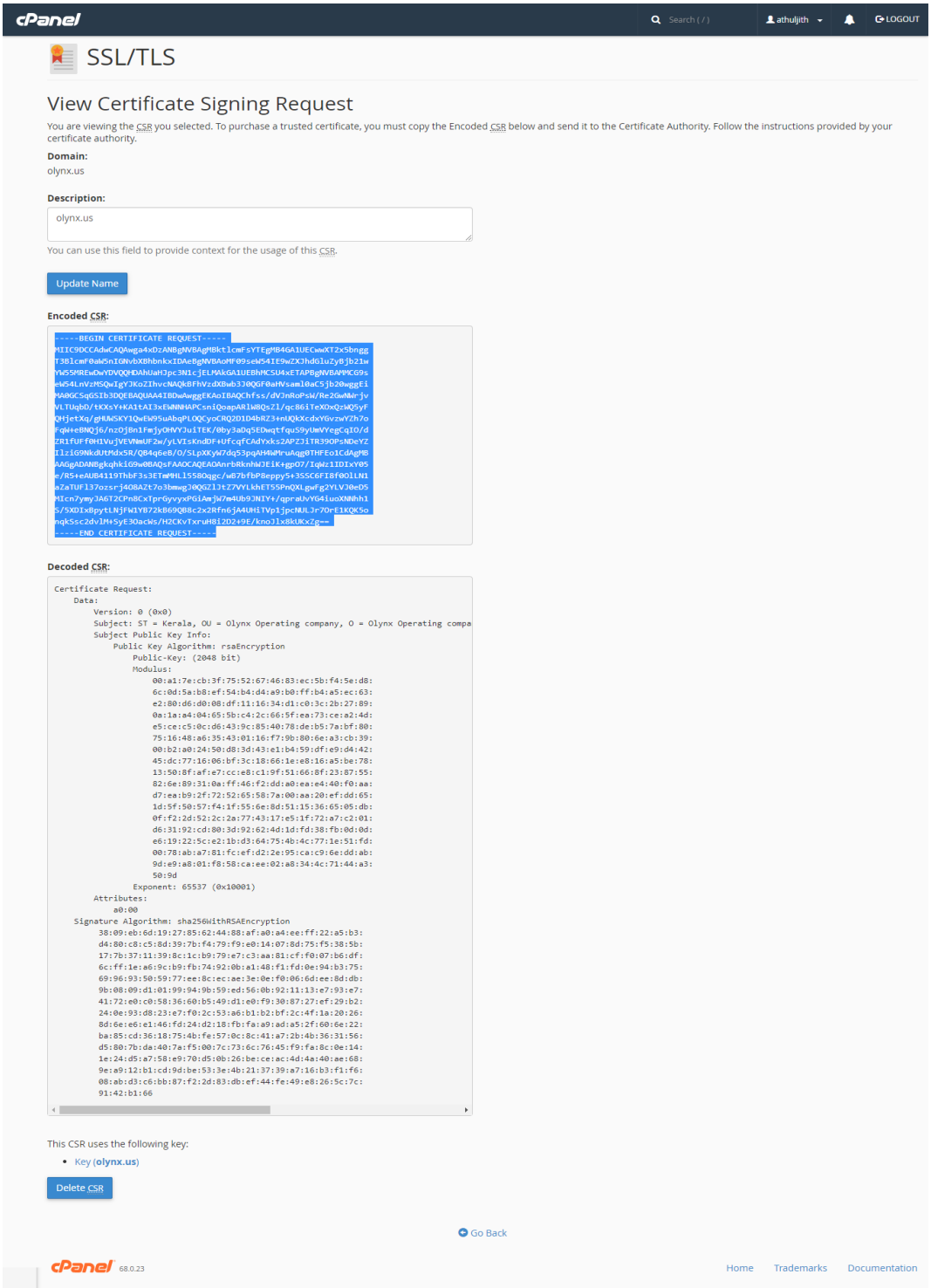

Fig 2.9 CSR generation on CPANEL

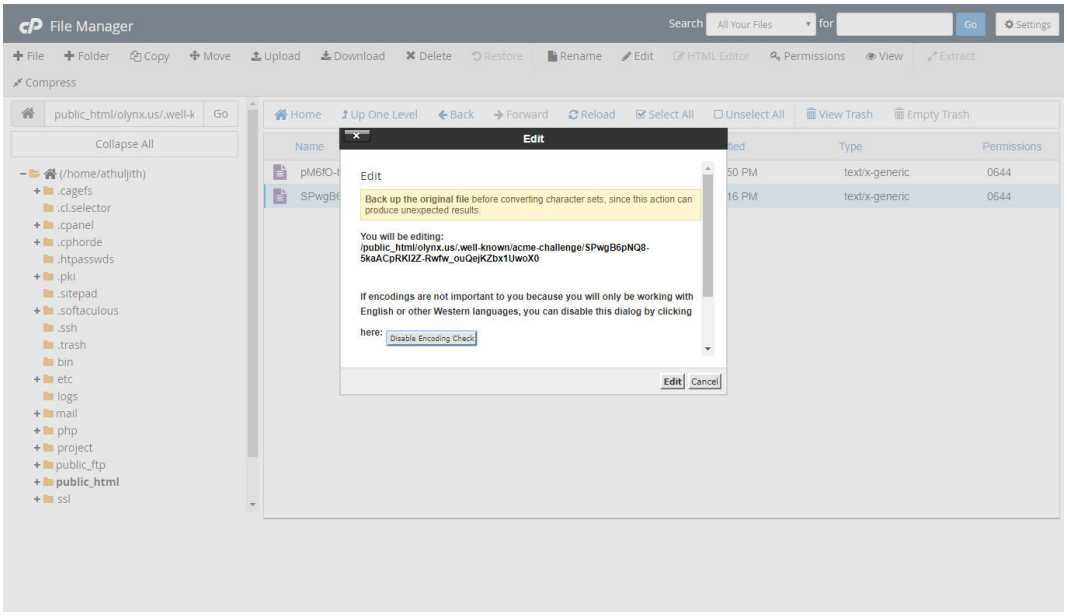

Fig 3 Uploading of verification file

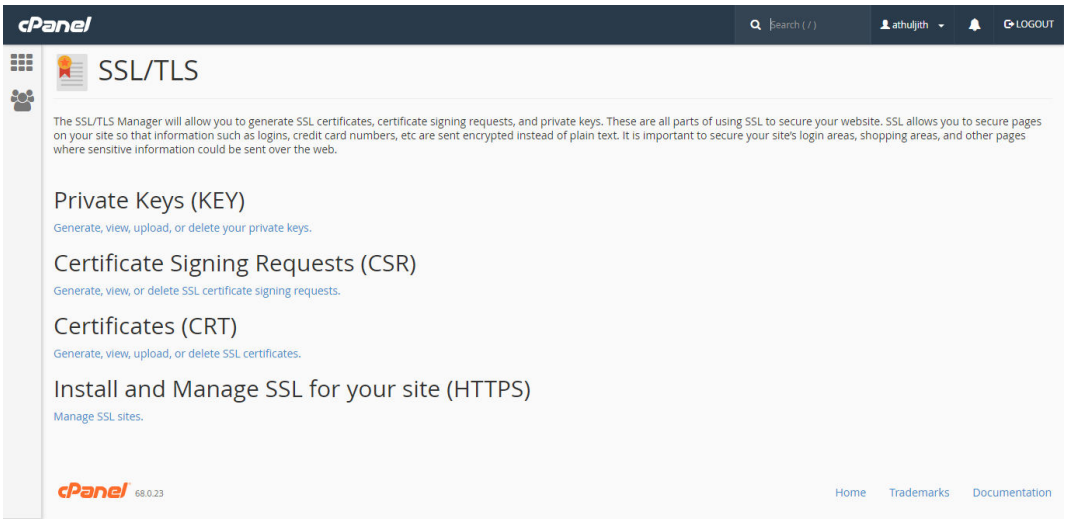

Fig 3.1 Main page on CPANEL to install SSL

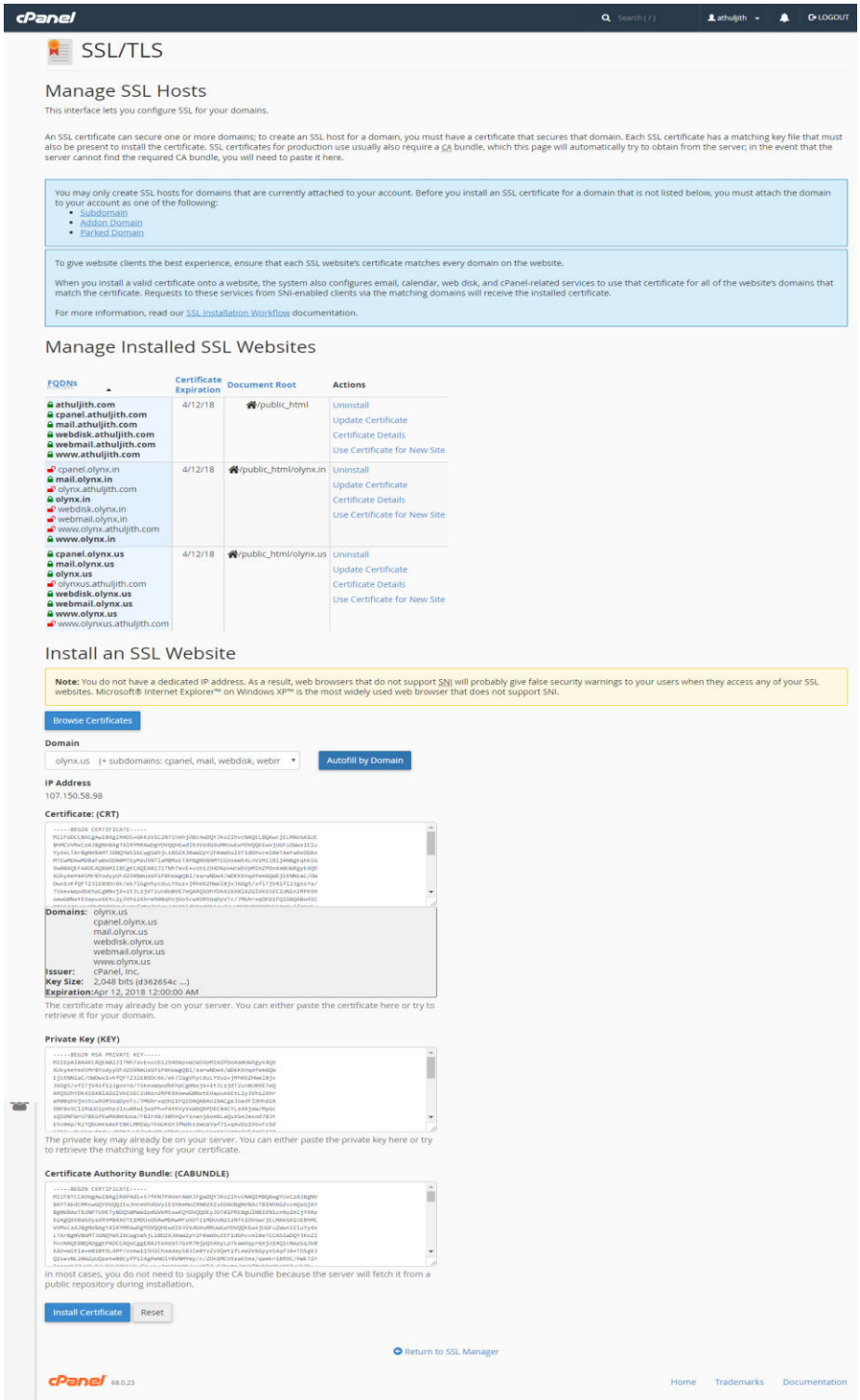

Fig 3.2 Installation of SSL certificate.

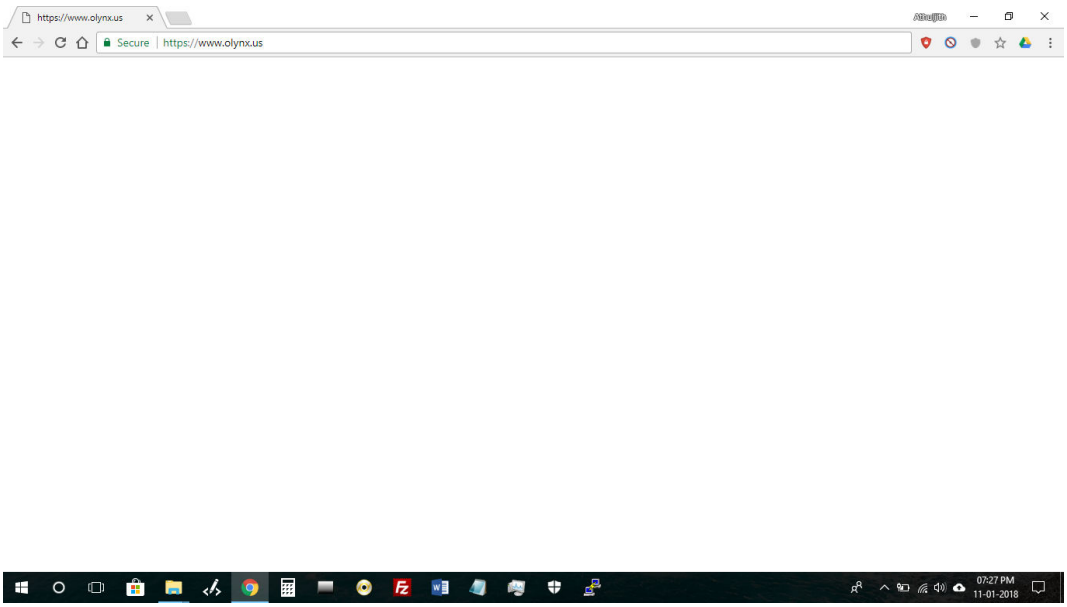

Fig 3.3 SSL issued website on web browser.

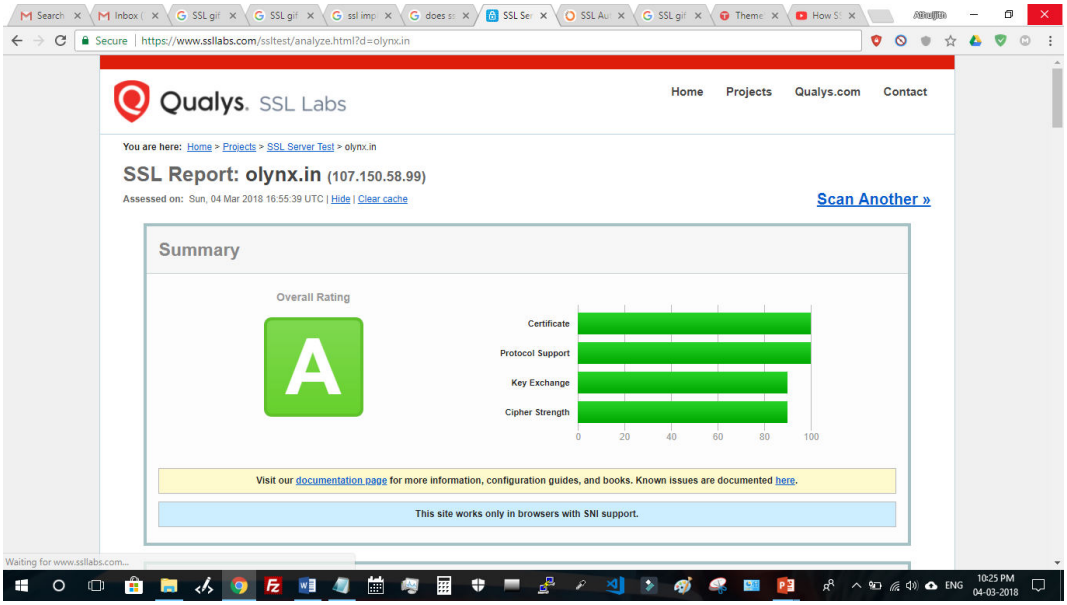

Fig 3.4 Quality check of issued certificate .

COMPARATIVE STUDY

Making a website like this will help some common customers to ensure quality SSL certificate for their small business website . and there can provide more security over http protocol.

EXISTING SYSTEM :

Existing system type , having some sort of amount to issue common ssl for their website , so there is some difficulty to install that ssl certificate to the root server

PROPOSED WORK :

We have proposed an system which can provide SSL certificate from the same CA authority with an all browsers trusted root certificate .

By using this website users can simply generate free of cost SSL certificates for their website . ID : https://automatic.athuljith.com

We have put our effort to reduce the cost of freely available SSL certificates to customers , so there by we can use this website to generate SSL certificates . Users just want to complete only 5 steps to generate the SSL, so there can make an easy type of generation . Clients will be more comfortable with this website .

8. ADVANTAGES AND DISADVANTAGES

8.1 ADVANTAGES

- Easy Generate SSL certificates
- The certificate is worldwide trusted
- Linux server implementation is available
- Automatic renewal is available
- Cost free CA certificate providing
- Secure all websites on a server with one single certificate

8.2 DISADVANTAGES

- Due to free of cost our servers may be get DDoS ATTACK

9. CONCLUSION AND FUTURE ENHANCEMENT

An SSL certificate is necessary to create SSL connection. The People can use Olynx SSL for generating SSL certificate with a simple client portal to the Linux Encryption Keys or Linux Encryption certificates, so there need to give all details about the identity of your website and your company as and when you choose to activate SSL on your web server. Following this, two cryptographic keys are created - a Private Key and a Public Key. The next step is the submission of the CSR (Certificate Signing Request), which is a data file that contains your details as well as your Public Key. The CA (Certification Authority) would then validate your details. Following successful authentication of all details, you will be issued SSL certificate. The newly-issued SSL would be matched to your Private Key. From this point onwards, an encrypted link is established by your web server between your website and the customer's web browser.

From technical point of view, there's no risk involved using any of the SSL/TLS Certificates whether it’s paid or free of cost. As every SSL/TLS Certificate protocol confirms that the handshake made between any client or server should generate secure and robust session keys to stop spoofing of data and cyber-attacks like man in the middle. What everyone should look out for is that whether the free SSL/TLS Certificate is capable of providing real time certificate status using any of OCSP (Online Certificate Status Protocol) or CRL (Certificate Revocation List) or not. This project will be an open through for providing SSL certificates on free of cost basis and also in a manner of easy SSL certificate generation. Main thing is that the website owner should be capable of conveying a message to their website visitors that their website integrated with SSL/TLS Certificate is trustworthy.

10.REFERENCES

[1]. Al-Janabi, SufyanFaraj, and Amer Kais Obaid. "Development of Certificate Authority services for web applications." Future Communication Networks (ICFCN).pp. 126-136,2012

[2]. Prohic, Natasa. "Public key infrastructures–pgp vs. x.509." INFOTECH Seminar Advanced Communication Services (ACS). pp. 134-152, 2004

[3]. Jancic, A., and Matthew J. Warren. "PKI Advantages and Obstacles." AISM.pp. 132-148,2004

[4]. Ten Have, Steven, Marcel vander. Elst, and Wouter ten Have. Key management models. Financial Times Prentice Hall,.pp. 128-152,2003

[5]. Hunt, Ray. "PKI and digital certification infrastructure." Networks, 2001. Proceedings, pp. 130-154 2001.

[6]. Nykänen, Toni. "Attribute certificates in x. 509." Techn. Ber., Helsinki University of Technology , pp. 135-157, 2000.

[7]. Curry, Ian. “Version 3 X.509 Cerificate.” Entrust Technologies, pp. 125-138,1996.

[8]. Gerck, Ed. "Overview of Certification Systems: X. 509, PKIX, CA, PGP & SKIP." The Bell 1.3, pp. 142-164, 2001.

[9]. CUDA-SSL: SSL/TLS accelerated by GPU , Published in: Security Technology (ICCST), International Carnahan Conference on , pp. 144-155,2017

[10]. Analysis secure socket layer protocol with heartbleed bug and distributed denial-of-service Sign In or Purchase , Control, Electronics, Renewable Energy and Communications (ICCEREC), International Conference on , pp. 142-170,2016

[11]. Performance analysis of enhanced secure socket layer protocol ,Communication and Network Technologies (ICCNT), International Conference on , pp. 136-154,2016

[12]. Devising Secure Sockets Layer-based distributed systems , pp. 133-156, 2012