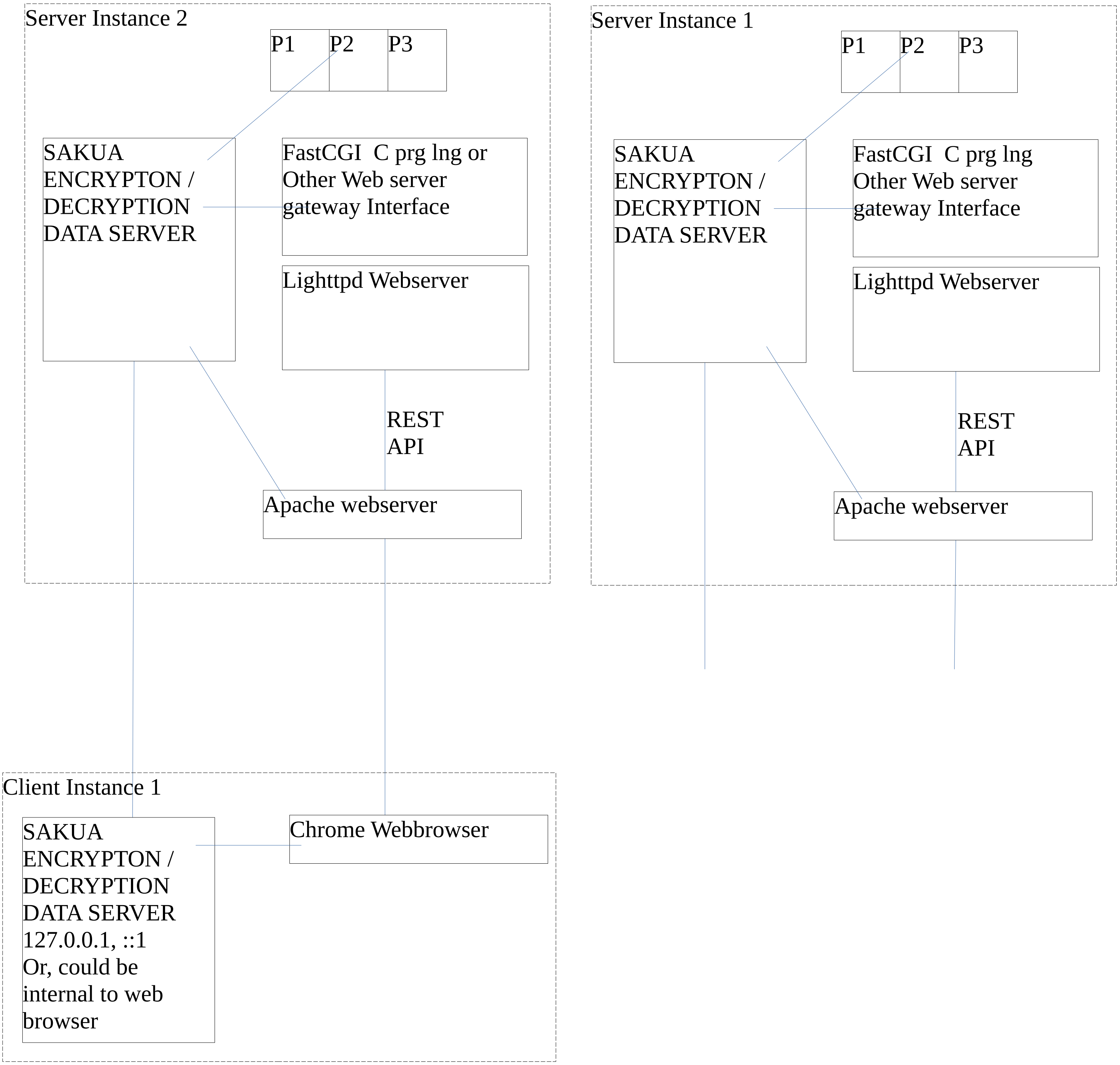
A Basic Design of SAKUA – Shared Access Key User Authentication

Copyright - Anoop Kumar Narayanan - 2023
anoop.kumar.narayanan@gmail.com


Server Instance 2
P1
P2
P3
SAKUA ENCRYPTON / DECRYPTION DATA SERVER
FastCGI C prg lng or Other Web server gateway Interface
Lighttpd Webserver
REST API
Apache webserver
Server Instance 1
P1
P2
P3
SAKUA ENCRYPTON / DECRYPTION DATA SERVER
FastCGI C prg lng Other Web server gateway Interface
Lighttpd Webserver
REST API
Apache webserver
Client Instance 1
SAKUA ENCRYPTON / DECRYPTION DATA SERVER 127.0.0.1, ::1 Or, could be internal to web browser
Chrome Webbrowser

SAKUA - How it works :

Registration to a website is done via a Normal HTTPS session.
Where in you supply some of the information present in KeyCreate file.
Including the DominoPassword Pattern and Pin.
https://github.com/anpnrynn/DominoPassword/
A C implementation is provided on this website itself :
https://github.com/anpnrynn/SAKUA/

NOTE:

1. GPS could be rounded off to a larger area.
2. SS could be something like DOB;Date for first transaction;Security phrase; or a combination.

The SAKUA session VERSION 1:

1. User accesses a website, this could be HTTP (I know right ?) or HTTPS (use this to be safer).
2. User is asked to enter the username, this information is transferred to the server.
3. The client/server calculates the the seedpin from username along with other information (including the password) on the website which was given during the time of registration. The information for generation of seedpin is displayed so that the user selects the same set of information on his side as well for the creation of encryption key.
4. The seedpin is used to calculate a seedstring.
5. The seedstring along with the registered pin is used to calculate securestring.
6. The securestring along with domino password pattern is used to calculate the shared encryption key.
7. The domino password pattern can be entered manually or retrieved from a secure saved file.
8. This encryption key can be generated on both the server and the client web browser / applications.
9. The encryption key is then used by the client to send the encrypted data over TCP/UDP sockets to the server with the username SHA512 or other message digest methods.
10. The server uses the message digest to locate the username and there by obtain the shared encryption key which is then used to decrypt the message.
11. Algorithms such as blowfish or something similar could be used as a shared key encryption algorithms.
12. From here on in there is a separate HTTP/1.0 or HTTP/1.1 or HTTP/2.0 session over secure shared key SAKUA connection happens, and data can be transferred over it. The interesting part is that the user is already authenticated and validated when the secure string is created as it is based on the username & password , registered pin and the registered pattern at the very least.
13. Optional Step: Now the nonce & CHAP/PAP (or newer Auth protocols) & OTP can be used over SAKUA to Authenticate the user over this secure data connection.

NOTE: The user needs to know the Pin, Password and Pattern, but the idea is that the pin and pattern could be saved in the browser/app and the password doesn't have to be a complex one, infact it could be very very simple one.

Also, for multiple websites you could have same pattern but a different 6 digit pin, now all that you need to remember is a single pattern, and pin and password pairs. This makes the end users' life very very easy while providing that person with a secure communication channel to the websites providing the service.

DominoPassword how it works:

1. User enters a pin from 2 to 999999. (HTML standalone webpage on the DominoPassword Github site only allows till 99999 as it is based on HTML and JS and its slow.)
2. A cryptographically secure string is calculated using basic shuffle algorithm which makes use of swaps and rotates.
3. The cryptographically secure string has all the characters present in the seed string.
4. The seed string could be website dependant.
5. The DominoPassword Pattern is entered whose length could be at max 108 characters long.
6. This pattern and sequence is mapped to the characters in the secure string generated via the pin.
7. The ouput is a cryptographically secure password which could be used as an encryption algorithm key.

A standalone HTTP file with JS version can be downloaded from dominopassword github site (dominopasswordgen2.html) should give you an Idea on how the DominoPasswordPattern is created. You can go back and forth between the pattern screens to create a more complex password. This was initially created as a Password Generation Tool.

Version 2:

Will create multiple single seed string and multiple seedpin with multiple securestring.
The pattern will be used to generate a very long cryptographically secure encrypion key. The length encrytion key could be increase dramatically.

Seedpin1 could be username + password
Seedpin2 could be username + password + dob
Seedpin3 could be username + password + dob + doj + ss + city ..etc

The algorithms could be different for each seed pin generation there by giving multiple secure strings when used with the same registered pin. This information about what combo to be used could be communicated via the initial access to the website.