Personal Information

- **Name:** Pavan Kalyan Thota
- **Email:** pavanyadav.thota@gmail.com
- **Github:** Pavan-kalyan-thota
- **Location:** Bloomington, Indiana, USA
- **Time Zone:** UTC-5 (EST)
- **University:** Indiana University
- **Degree:** Master in Sciences
- **Major:** Computer Science
- **LinkedIn:** https://www.linkedin.com/in/pavan-kalyan-thota/

Project Description

- **Name:** Bringing Password-less Authentication to Plone 6
- **Mentor:** Andreas Jung, info@zopyx.com
- **Goal:** Make Authentication in Plone more secure and user-friendly using Password-less Authentication standard WebAuthn by W3C.

About Me

I have been programming for the last 6 years using various programming languages, tools, frameworks, and libraries as part of my academic work and personal interest. I am in my master's degree now, majoring in computer science at Indiana University Bloomington. I have experience of teaching a graduate-level course using python as an Associate Instructor at my university. I also worked as an Integration consultant at an IT firm after my undergraduate that involves working with multiple CRM and Cloud service providers like Salesforce, Mulesoft, and Dell Boomi, etc. As part of my coursework in undergraduate and graduate degrees, I built multiple full-stack projects using Django, React.js, Node.js, Kotlin, etc.

Plone

Plone is an open-source content management system built on top of the Zope application server. While Plone identifies itself as an “Enterprise Level CMS” the existing security features offered are not up to the industry standards. Changes can be made to existing password policies to make authentication more secure, however, many organizations have started using WebAuthn a password-less web standard by W3C to make their authentication process more secure and user-friendly.

Since it has already been decided to use WebAuthn in Plone the next step is to design and plan how to add this to existing applications(Plone 6, Classic UI, Volte, etc). As mentioned in the details of the project here, we can use WebAuthn in two ways as 2FA or as a Primary

Authenticator. I propose that we need an implementation that allows the User(Admin) to decide which way to use WenAuthn in their Plone instance i.e, either as primary Authenticator, required, or optional second-factor Authenticator.

WebAuthn and Public Key Cryptography

WebAuthn is a passwordless Authenticating standard by W3C using public key cryptography. Instead of storing user credentials containing passwords in servers with WebAuthn we store credentials with a public key which means nothing without their respective private key. So when the security of the server has been compromised and an unauthorized user gains access to stored encrypted passwords it would not be a threat to users' authenticity as the credentials with the public key alone cannot be used to gain access. Since these are unique to individual sites failure of one server will not cause any security concerns for users' authenticity on other sites more.

Not Available or Lost Authenticator Device

One of the major concerns regarding Webauthn is the scenario where the authenticator device is not available or lost many of the solutions discussed here need the use of an external device or service which are not available within the open-source community. So they might not be useful for this project. (open for discussion)

When the user is registered with more than one authenticating device, they can use the other available device to log in and make changes to existing authenticators but in the case of only one registered device, the proposed ideas from doc are Back-Up Codes, Recovery Codes, and Administrator Reset.

- **Back-Up Codes:**
    - These are One-Time use Login codes in case of an emergency with limited privileges.
    - Useful when the authenticator device is not available only at the moment.
    - Users have to generate these codes when registering and note them or print them for use in an emergency.
    - Users will be able to view or edit content but will not be able to change existing security settings.
    - When the user generates backup codes they will stored on the server securely in encrypted form and later used to validate when required.
    - However, this is not a secure way since it is prone to SQL Injection, phishing, malware, and other security attacks.
- **Recovery Codes:**
    - These are codes generated by users when they had access to the site and saved or printed securely.

    - Users can use these codes to register a new device and remove access from old devices.
    - These are helpful when the primary authenticator device is lost or needs to be replaced with a new one.
    - New public and private key pair will be created for the new device and the new public key will be added to the server.
    - However, recovery codes like backup codes need to be saved on the server and they are prone to security attacks.
- **Administrator Reset:**
    - Manual Reset of an account by an administrator based on some user verification.
    - It is confirmed that the private keys cannot be copied and shared from here so the admin cannot register a public key on the server and share the corresponding private key with the user that he can add to his device.
    - Instead of registering the device himself(admin) which might not be possible unless the administrator has access to the authenticating device. I propose that the admin should be able to generate recovery codes and share them with the user and the user can recover his account himself.
    - Implementation will be the same as mentioned in Recovery codes but these recovery codes will be generated by Admin.
    - Again storing any kind of secret on the server is a security concern.
    - There might be some other way to do this using Plones admin privileges which I am not completely aware of so it can be discussed further.

Two-Devices Approach

It can be observed that somehow all the above methods need us to store some kind of secret on the server which we should avoid. Since we cannot provide or recover a stronger credential with a credential weaker than it. Hence I propose if possible instead of using the above methods it is more secure to implement a rule that to complete a user registration using WebAuthn it is mandatory to have at least two authenticators from two different devices at least for users at the admin level privileges (more).

Deliverables

- Detail Design of implementation to integrate WebAuthn into Plone.
- Integration of WebAuthn into Plone as an additional option for registration and login that is configurable as a 2FA or Primary Authenticator.
- UI functionalities in Classic Plone with forms and dialogues to allow users to Register or Login using WebAuthn.
- Plone and UI functionalities to Manage keys.
- Secure Implementation of Account Recovery more details will be decided on the first deliverable.

Project Timeline

- Present - May 28
    - Get familiar with Plone stack, Documentation, and community.
    - Get to know mentors and begin to contribute to Plone.
    - Research more about ongoing developments regarding WebAuthn Account Recovery.
- May 29 - June 30
    - Final Decision on how to implement WebAuthn Account Recovery.
    - Setting up the initial Project repositories and git workflows.
    - Confirming Midterm evaluation goals.
- July 1 - July 10
    - Implementing the Plone and UI parts of the project.
    - Working on delivering midterm evaluation goals.
    - Submitting the midterm deliverables.
- July 11 - Aug 10
    - Implementing the changes after midterm evaluation if any.
    - Confirming final evaluation goals for the project.
- Aug10 - Aug 21
    - Working on completing the final project deliverables.
    - Submitting the final project with the necessary documentation.
- Aug 21 - Sep 4
    - Submitting the complete project deliverables to GSoC.
- Post GSoC
    - Continuing to be part of the community, learning about other ongoing projects, or starting new projects to improve Plone.

Why Me

I am a very persistent programmer that never quits until I complete what I have to. I am very thankful to the open-source community that has provided us with many applications on which modern technology is built and I am thrilled to start contributing to it. I believe in Hand-On learning and tried to add a lost device page to the prototype project here. The prototype uses the proof-of-concept project from here but also creates a backup code (“1234”) while adding the public-key credentials to backend auth_databse that will be used to validate user later, doing this helped me to understand the working of WebAuthn Registration and authentication processes more. I get acclimated to the programs I build or use, So even after GSoC 2023 I am more likely to be part of the community or might even start my own projects.

Note: This is the initial proposal open for any comments and suggestions.