FusionAuth

Cloudelligent

The State of Homegrown Authentication Report: 2025

TECH PAPER

Table of Contents

Dive into the tech stack and composition of the teams building the login experience on their own.

Introduction

This inaugural State of Homegrown Authentication report, sponsored by FusionAuth and Cloudelligent, is the first to dive into the tech stack of those building authentication themselves.

The Customer Identity and Access Market (CIAM) has historically ignored the practices of the teams building authentication on their own, leaving a scarcity of direct insight into peers' challenges and successes.

In this report, teams building authentication can get a solid sense of where others doing the same thing have found success.

Executive Summary

The State of Homegrown Authentication report paints a clear picture of the teams building their own login systems—from the mix of third-party libraries and auth servers they use (or don't) to the people dedicated to the task. Here are the key takeaways:

Productivity

1. Only 30% lean on external help like third-party auth servers or libraries, while the rest either run their own auth server or build everything from the ground up.
2. The parts of homegrown auth that take the most time are maintenance and microservices architecture design.
3. 60% of teams say their best and brightest are leading the charge.
4. When it comes to authentication library initial time to value and long-term maintenance, Spring Security (Java) and Passport.js are on top, followed by ASP.net and Django.
5. Compared to the overall usage of authentication libraries, Express Session took the place of Devise (Ruby) and Laravel Sanctum as a close 5th place.

The Tech Stack

1. 72% were running auth in a containerized or Kubernetes environment.
2. Spring Security (Java) and Passport.js were the most commonly used authentication libraries.
3. The most common CI/CD tool in use is Github actions.
4. 44% prefer to develop locally, but a full 50% of those that are doing any kind of testing prefer to test locally, versus in SaaS.

Features & Challenges

1. Passkeys are a polarizing feature; the same number of respondents felt like experts as those who felt completely unfamiliar with them.

2. The biggest advantage and challenge of running authentication in-house is security.

3. While teams value the security benefits of rolling their own auth, a full 20% reported experiencing a security breach in their homegrown auth system.

The Detailed Results

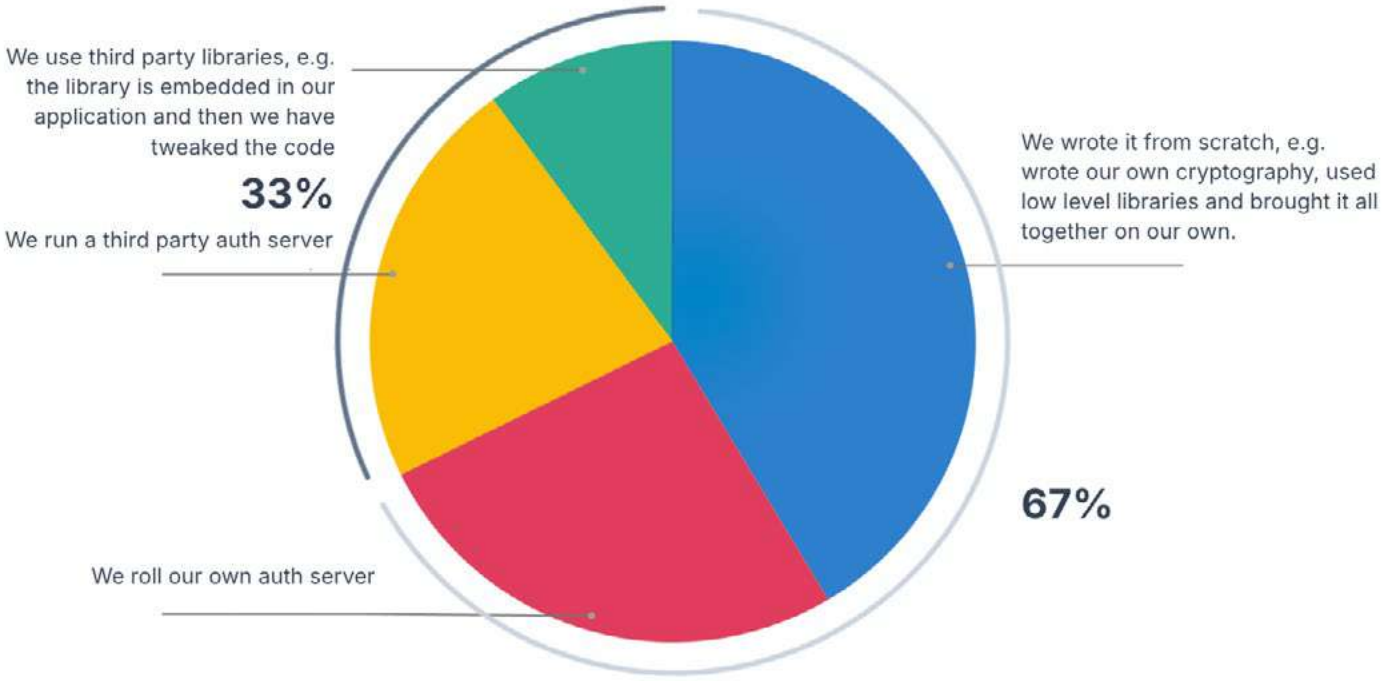
The experience of the team building authentication themselves can be roughly divided into people and elements impacting productivity, key features and challenges of homegrown auth, and the tech stack. Here you can understand your peers' experiences, to inform your own plans for homegrown auth moving forward.

Productivity

What kind of homegrown auth?

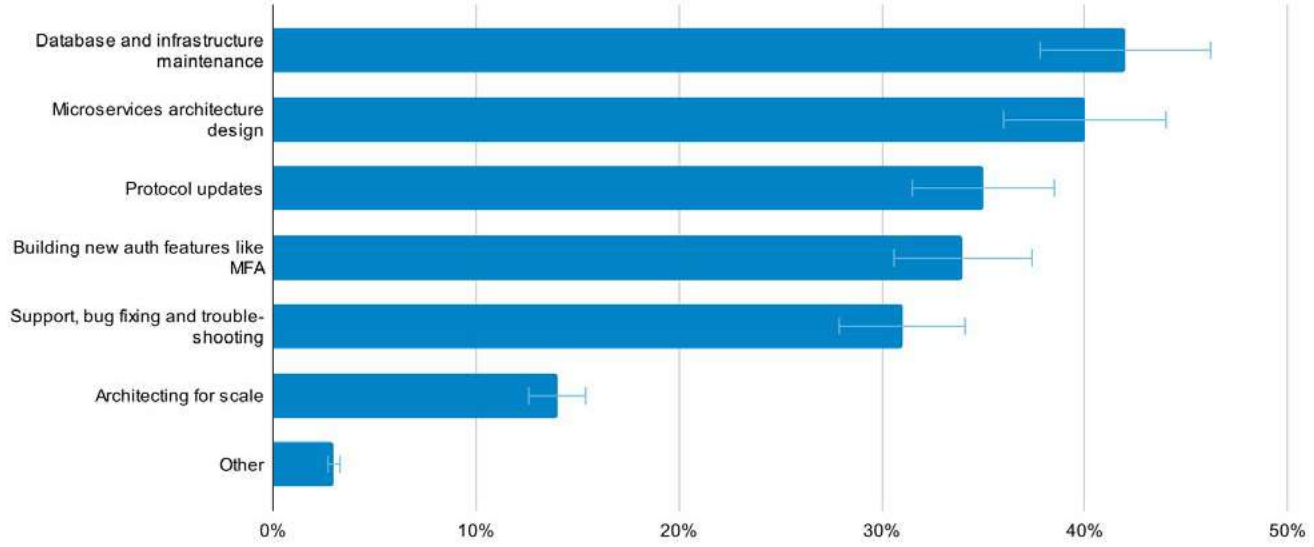
A full 67% of respondents either wrote their auth from scratch or rolled their own auth server. In contrast, only 33% ran a third party auth server or used third party libraries.

Chart 1. What kind of homegrown auth did you implement?



In any of the set-ups listed above, the parts of homegrown auth that take the most time are maintenance and microservices architecture design.

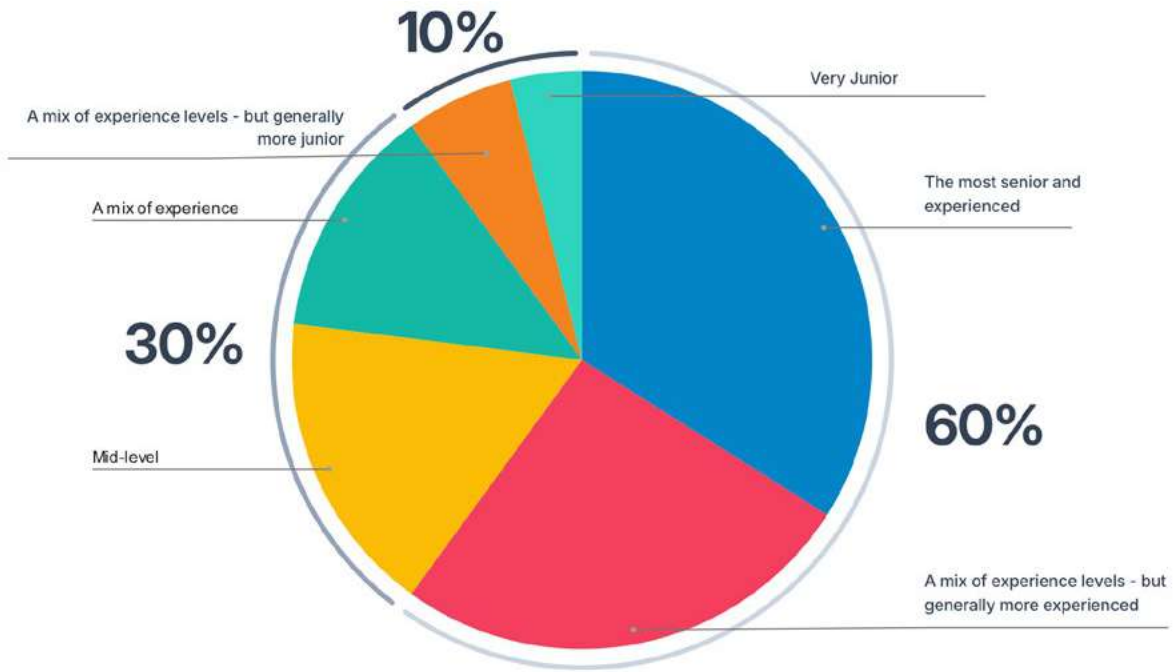
Chart 2. What are the parts of your identity infrastructure that take the most time?



People Requirements

This study found, unsurprisingly, that the folks working on homegrown identity are generally the more senior people in the org, with 60% saying that the most senior, or generally more experienced people were involved, and only 10% saying junior people were responsible.

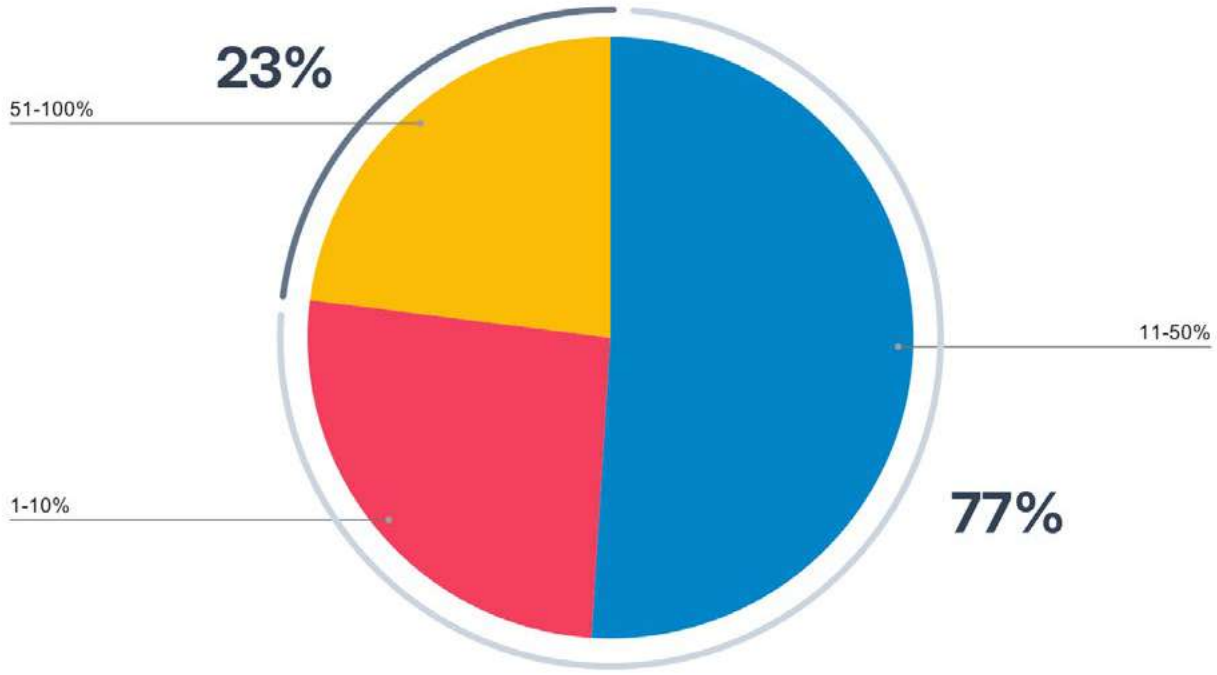
Chart 3. Overall, how senior are the people on your team working on auth?



It appears that the 60% of senior people might not necessarily have been specialists in auth, as only 23% (instead of 60%) stated that 50-100% of those working on auth were actually experts in it.

77% reported that the number of experts in auth, out of those working on it, were 50% or less.

Chart 4. How many of the people working on your authentication are specialists in authentication/identity?



The Homegrown Auth Stack

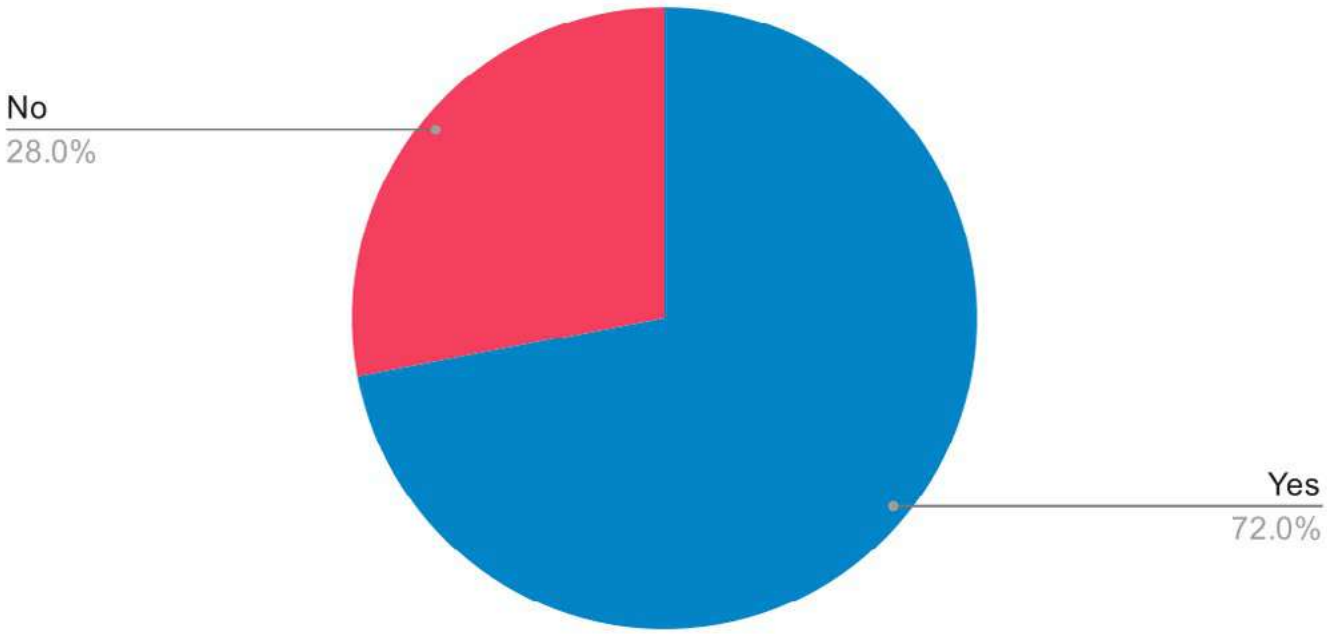
In this section, we look at a typical stack for homegrown auth, including these factors:

1. **Containers and Kubernetes**
2. **Auth library**
3. **Programming language**
4. **Protocol**
5. **CI/CD tool**

Containers & Kubernetes

A clear majority, at 72%, are running auth in a containerized or Kubernetes environment.

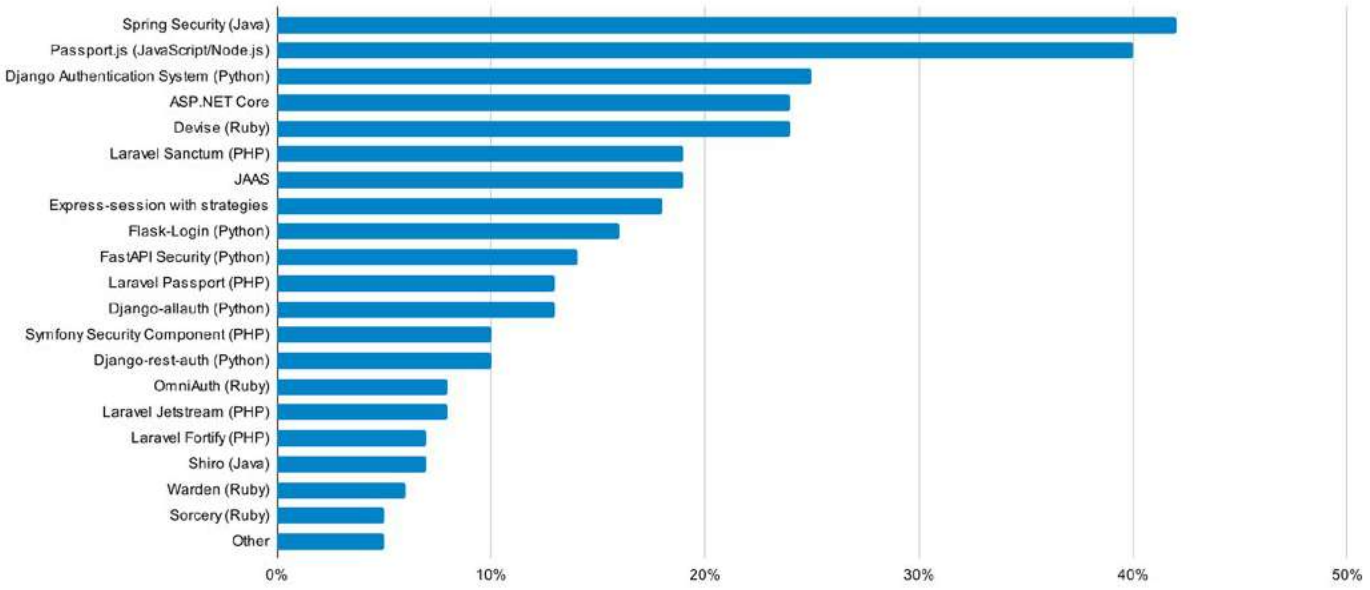
Chart 5. Is your homegrown auth running in a containerized or Kubernetes environment?



Authentication Library

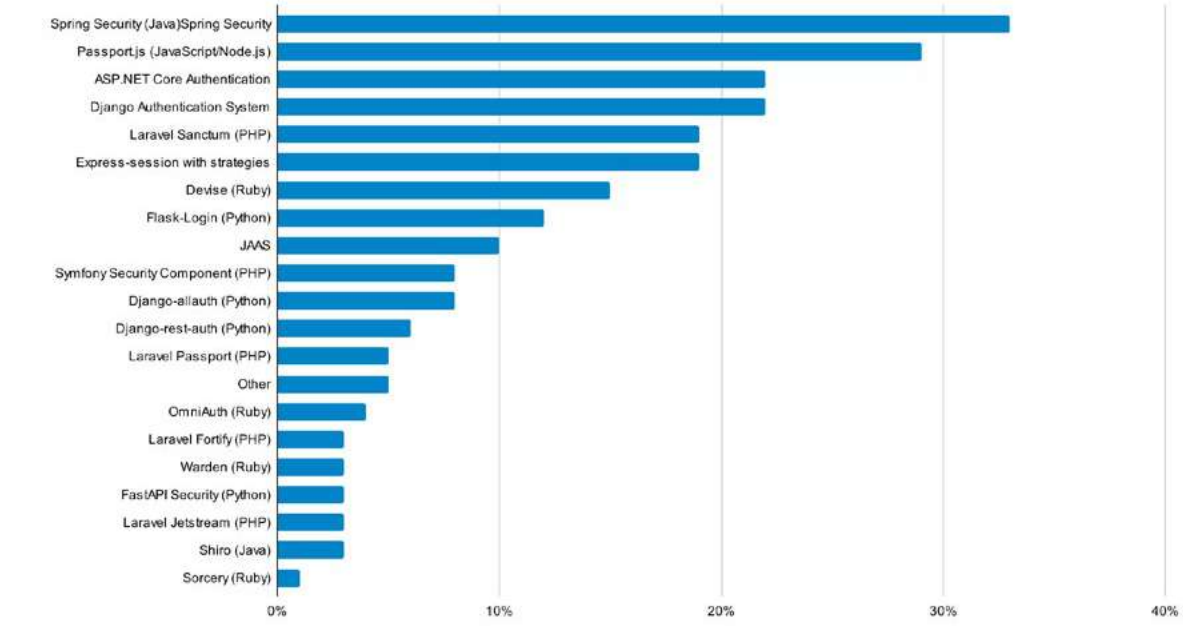
The top authentication libraries in use are Spring Security (Java) and Passport.js, followed by Django, ASP.NET and Devise (Ruby). Some of the least reported in-use included other Ruby options like Warden and Sorcery.

Chart 6. What authentication library do you use?



When it comes to the best libraries for long-term maintenance, we don't see a whole lot of difference from the top used libraries.

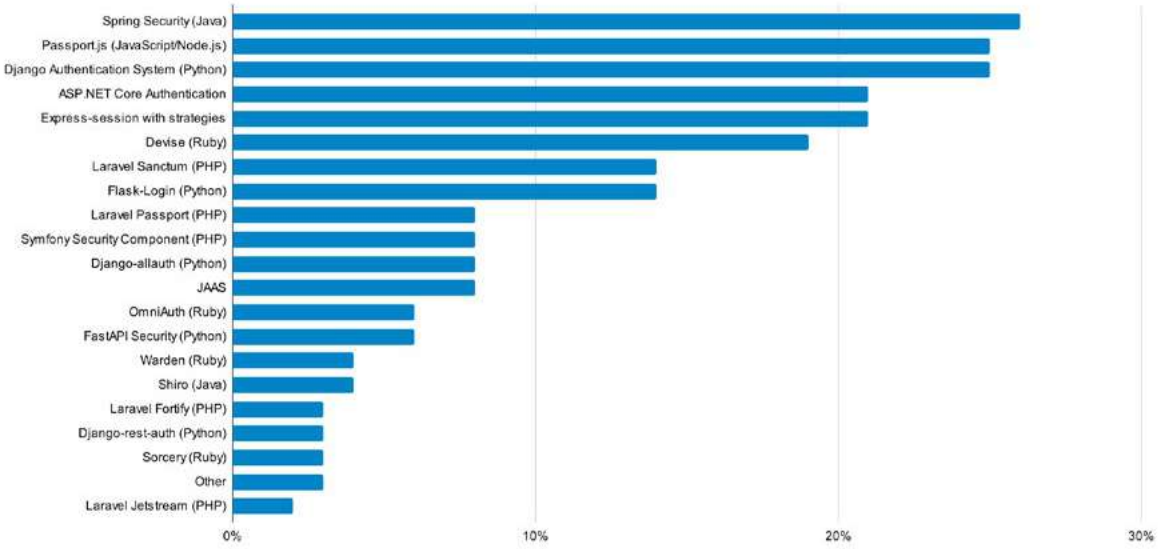
Chart 7. Which authentication library is best in terms of long-term maintenance?



When it comes to time to value, Passport.js and Spring Security still came out on top, but this time with Django as a close third.

Express-session with strategies also moved up above Devise (Ruby) and Laravel Sanctum (PHP), compared to the chart of the libraries that are used most often.

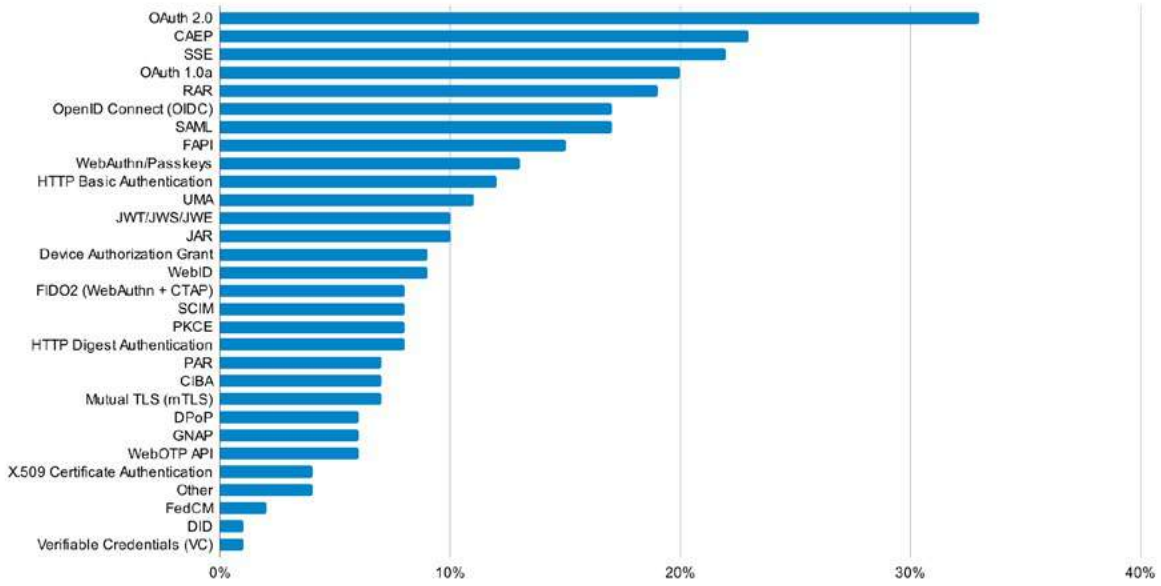
Chart 8. Which authentication library is best in terms of initial time to value?



Protocol

There was a large amount of variety in the protocols used for homegrown auth across the board. Surprisingly, OIDC was listed as often as SAML, which is generally viewed as a relatively difficult protocol to implement (depending on the version of course).

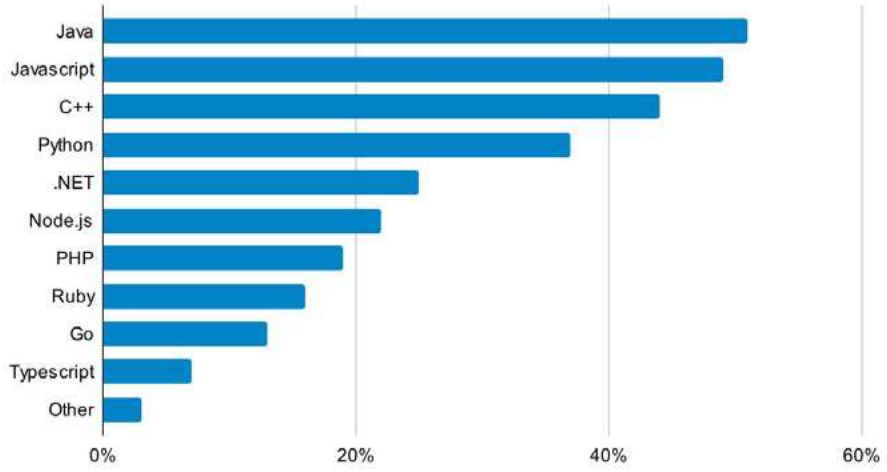
Chart 9. What protocol(s) is your auth based on



Programming Languages

The most common programming languages in use were Java, Javascript and C++ . Similar to the authentication libraries in use, Ruby showed up near the bottom here as well.

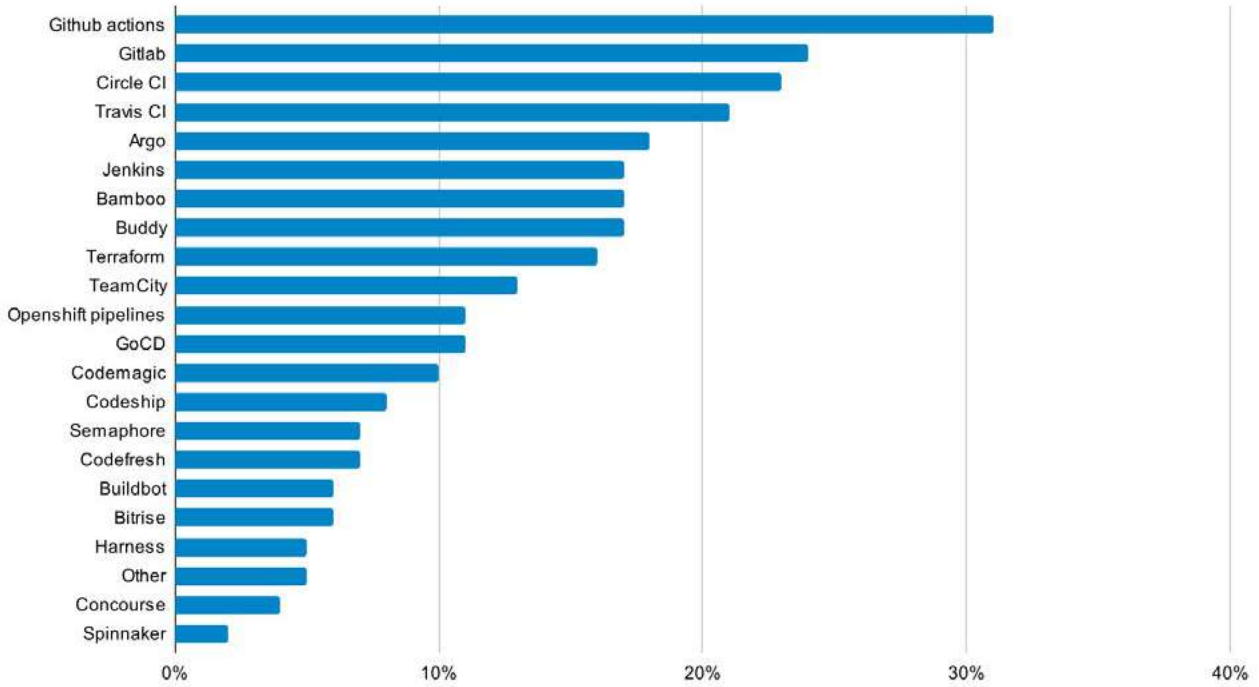
Chart 10. What programming language(s) do you use?



CI/CD Tool

The most common CI/CD tool used by teams building homegrown auth was GitHub actions, followed by Gitlab, CircleCI and TravisCI. The least commonly used was Spinnaker.

Chart 11. Which CI/CD tool are you using?



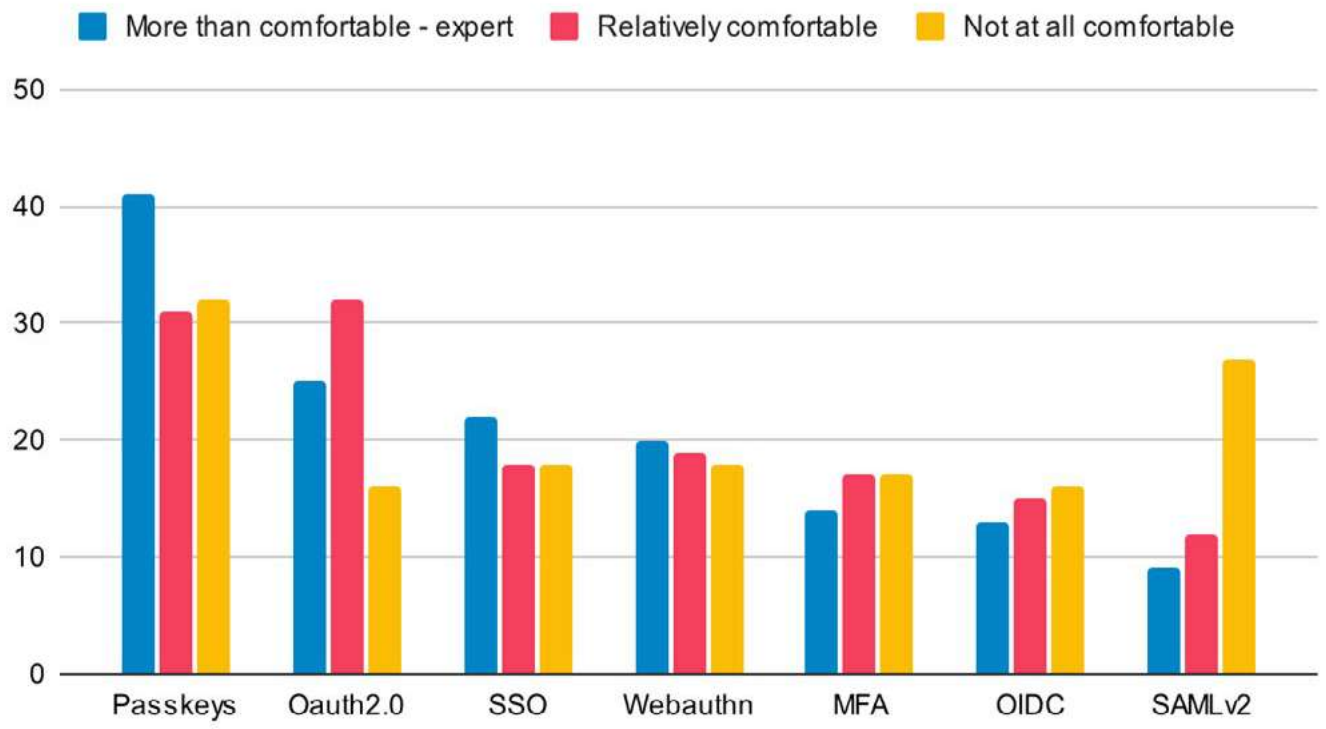
Features & Challenges

Passkeys are polarizing across the board

Passkeys seemed to have a love/hate relationship with those rolling their own auth:

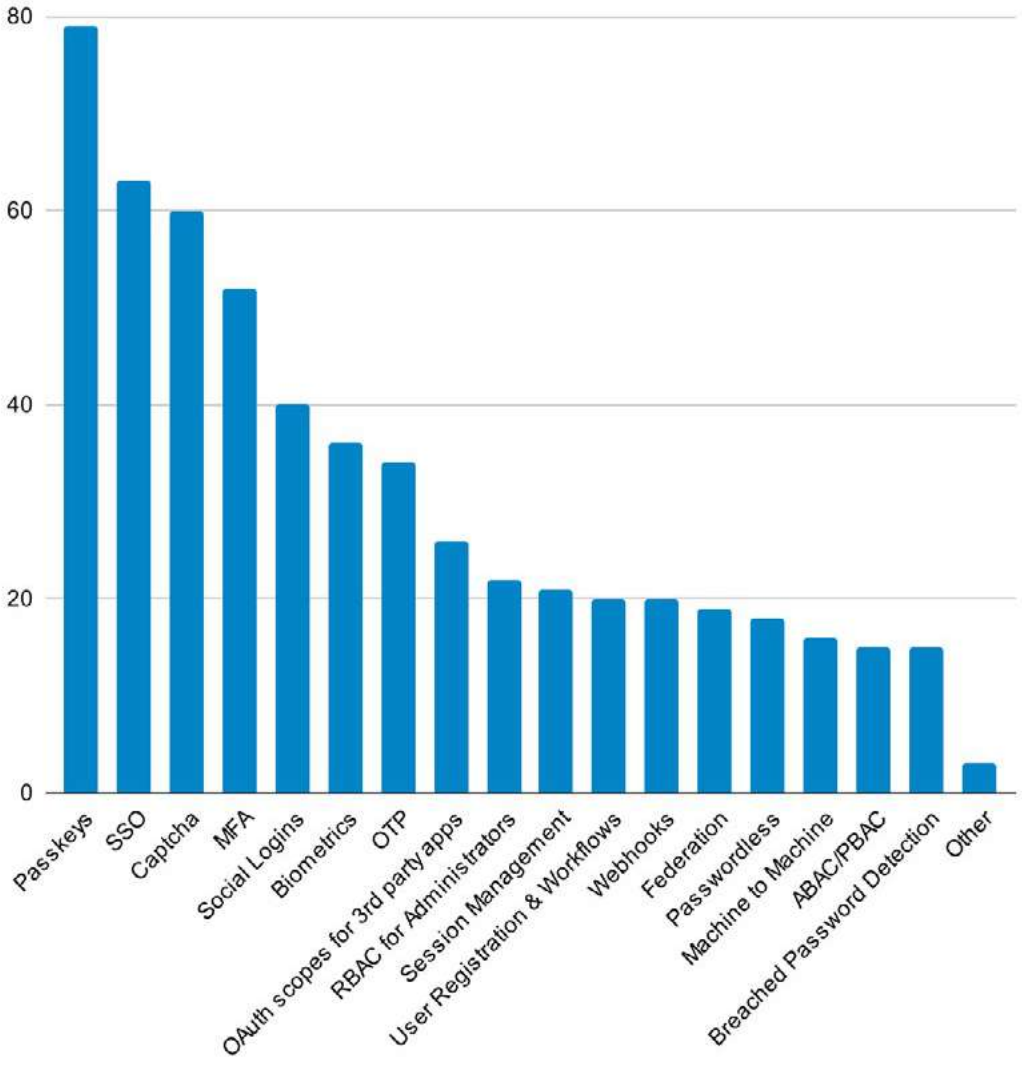
Respondents were both most and least comfortable with passkeys - followed by SAML.

Chart 12. How experienced are you with the following?



Passkeys were among the most implemented features, followed by SSO, Captcha and MFA (the least implemented were breached password protection, passwordless, ABAC/RBAC).

Chart 13. Which of the following has you or your team already implemented?

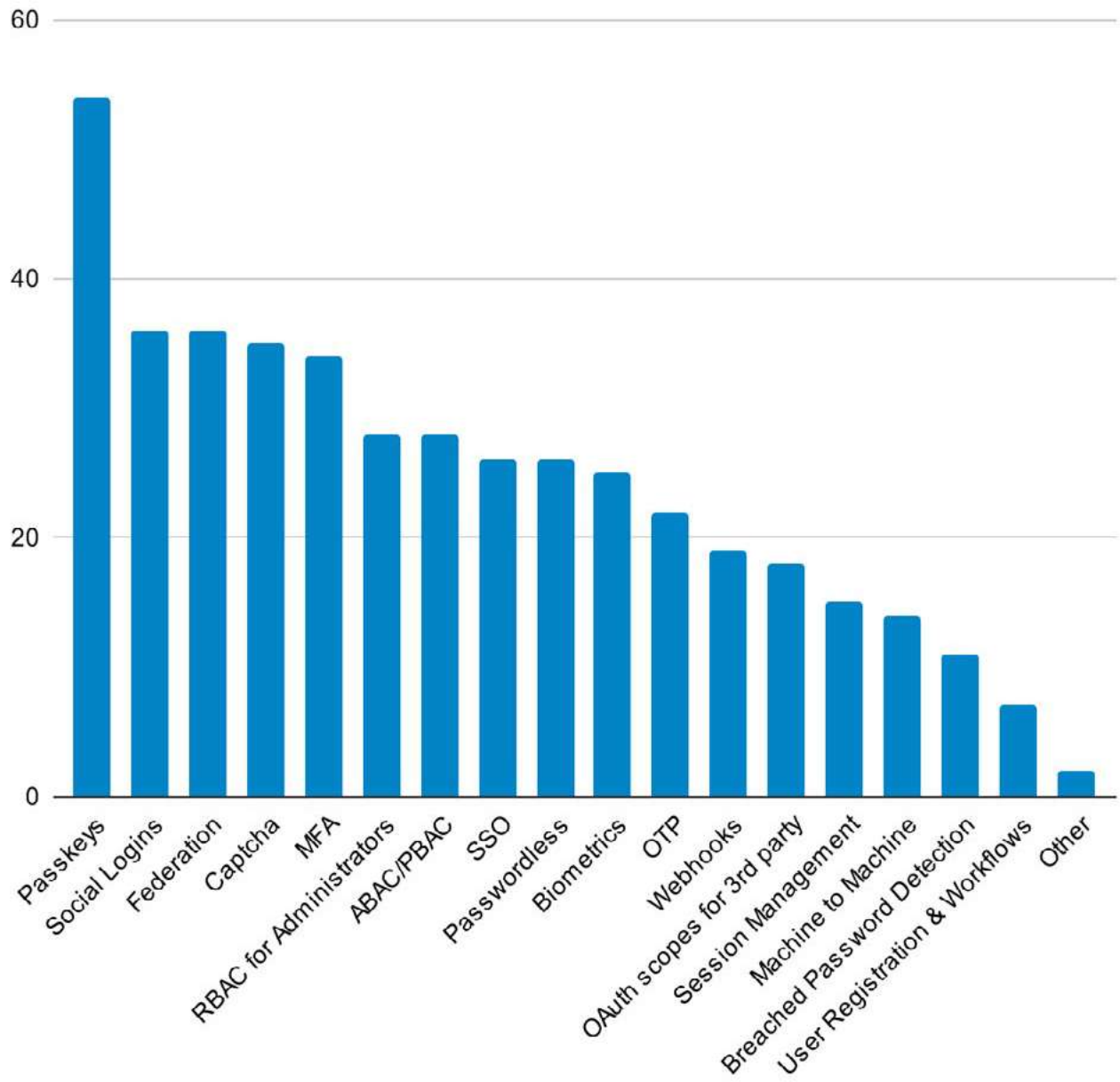


Not only were passkeys the most implemented feature, they were also the most desired feature that had not yet been implemented, followed by social logins, federation, Captcha and MFA.

The features that teams were the least interested in implementing, even if they didn't have them, were user registration, breached password protection and Machine to Machine, or service accounts.

Surprisingly, SSO is only in the middle of the pack here. Based on the fact that it is already highly implemented, one might infer that it is a highly prioritized feature to develop, and teams are not waiting on implementing it, or unable to.

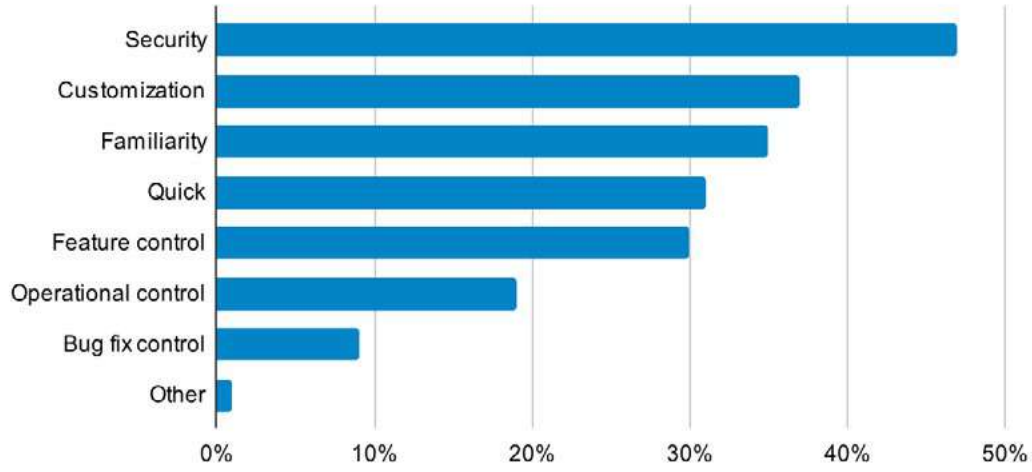
Chart 14. Which auth features do you want to have, but don't have currently?



Challenges & Benefits

Security and customization are listed as the best part of rolling your own auth.

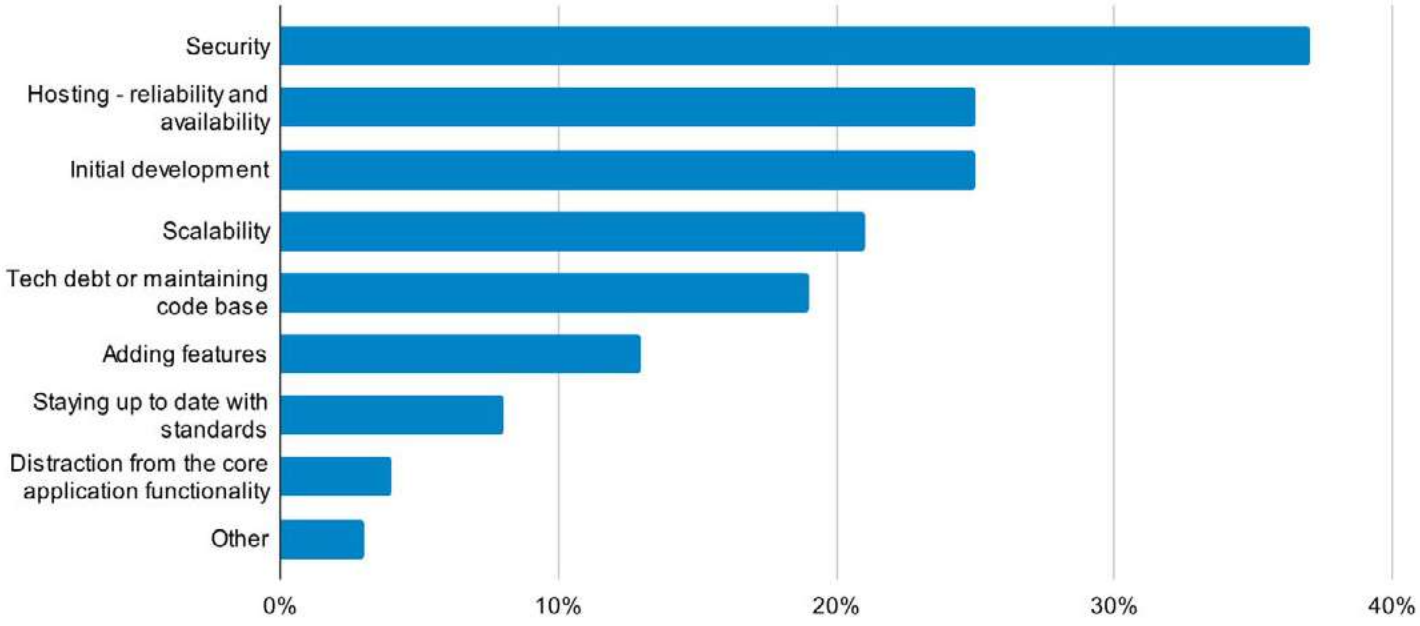
Chart 15. What are the best parts of rolling your own auth?



Security is also the biggest pain point of rolling your own auth, followed by architecting for reliability and availability, and the initial development.

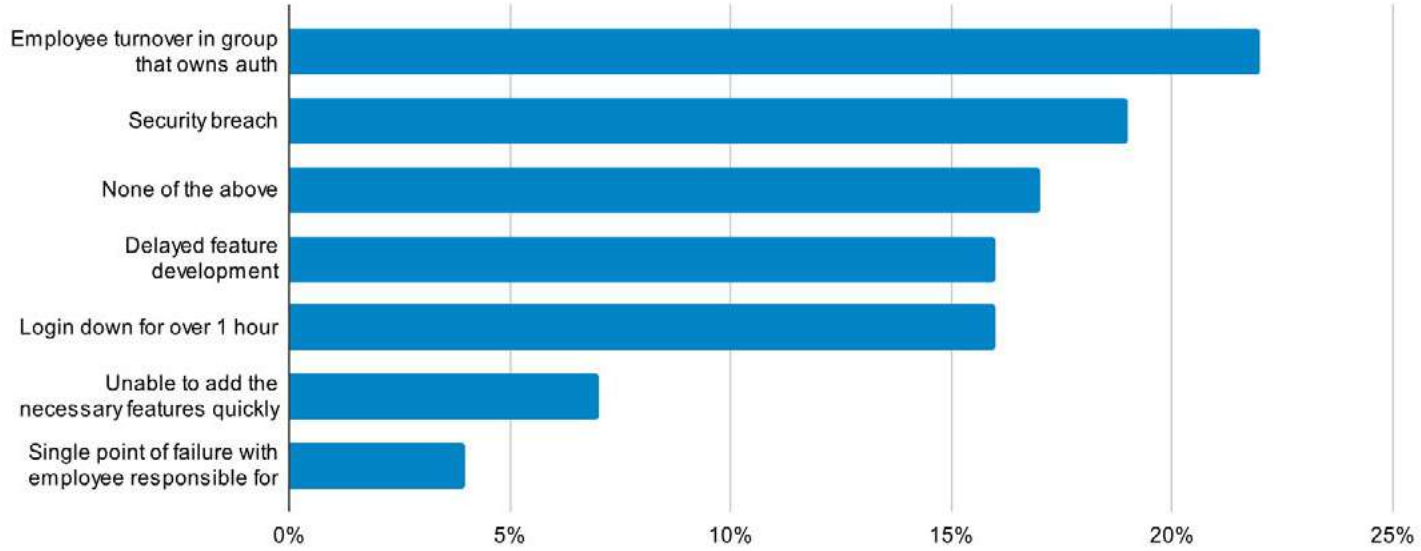
Notably, staying up to date with standards and distraction from the core application functionality were not the biggest pains.

Chart 16. What are the biggest pain points for rolling your own auth?



Given the importance of security as both a benefit and a challenge, it was interesting that almost 20% reported a security breach of their auth. That being said, it was not common to experience delays in feature development or login being down for over an hour.

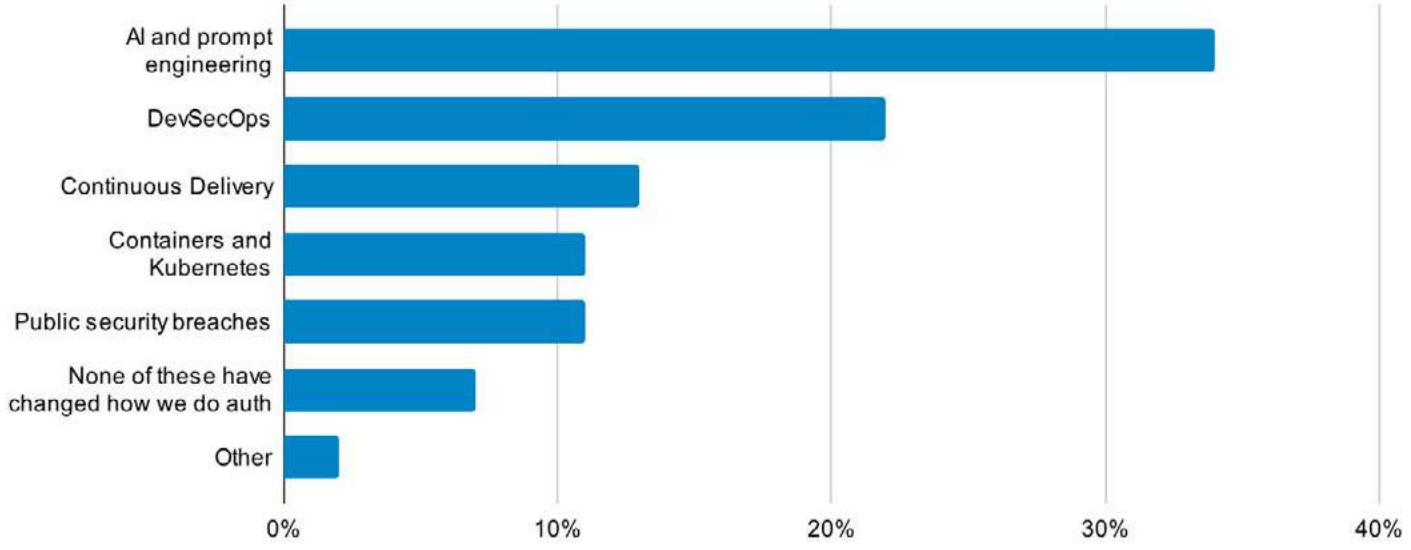
Chart 17. Have you experienced any of the following with your in-house authentication?



Team Preferences & Dynamics

Interestingly, the engineering trend that has had the most impact on how these teams build auth is AI and prompt engineering, followed by DevSecOps and Continuous Delivery. This is not surprising, given the significant difference in testing that these trends have sparked.

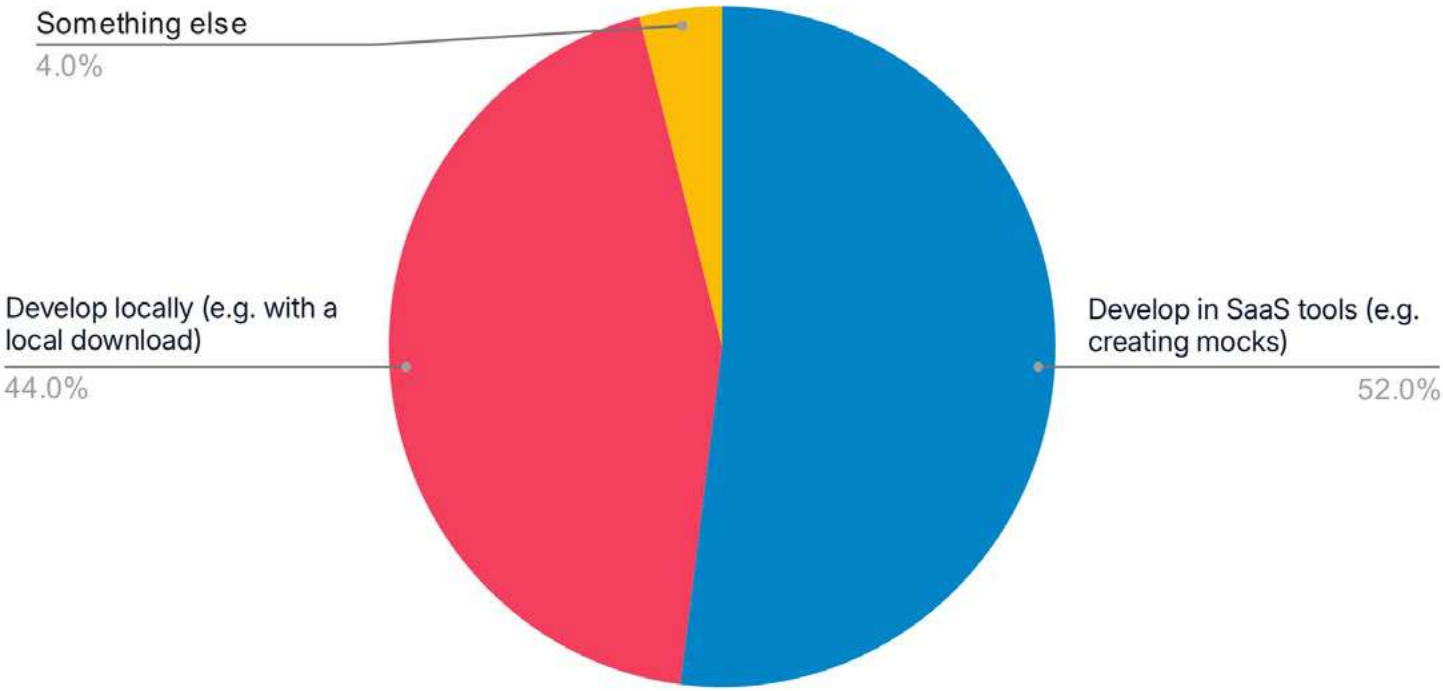
Chart 18. Which of these innovations have changed how your team builds and designs authentication the most?



Interestingly, almost half of teams prefer to develop locally when compared with developing in SaaS.

This is interesting given the overwhelming majority of options to outsource authentication only allow for multi-tenant SaaS.

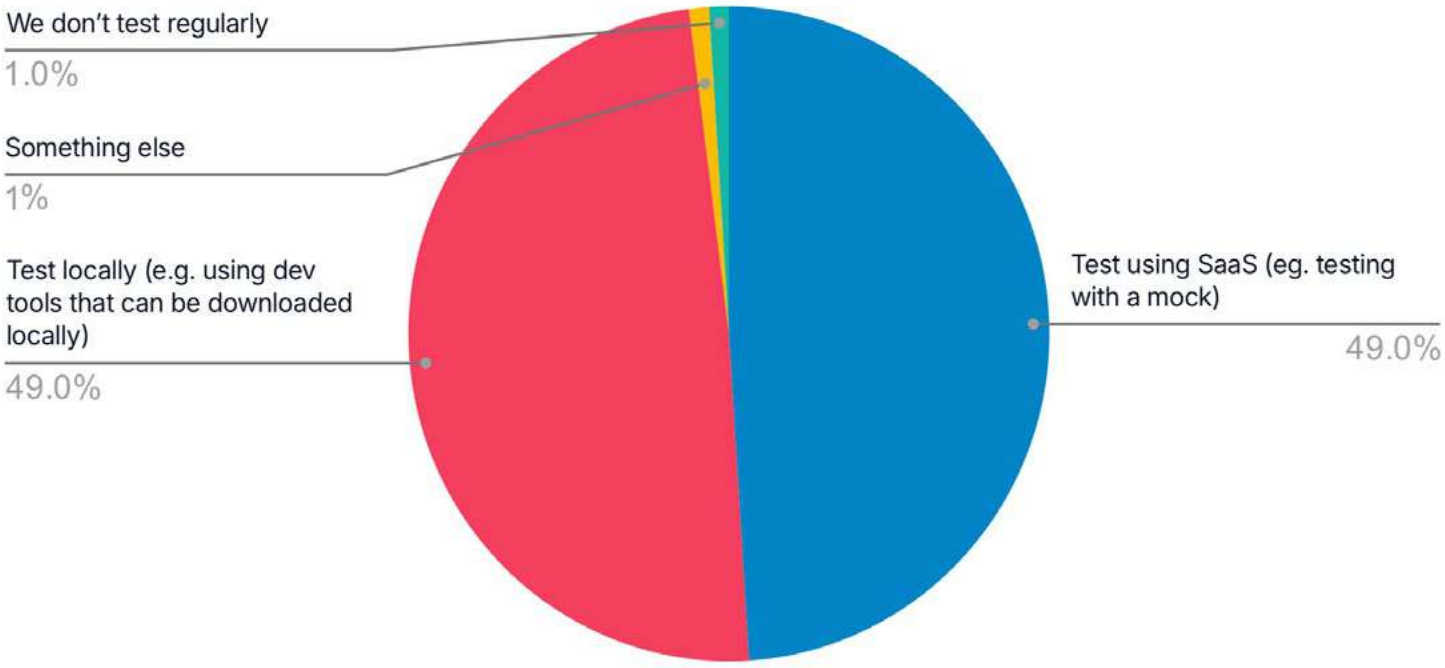
Chart 19. Which of the following types of software do you prefer for your dev environments?



This is a similar story when it comes to testing, except that the preference for local testing, versus testing in SaaS tools, is 50/50 (out of those that do any kind of testing).

This again points to a strong desire for many teams doing homegrown auth to build locally.

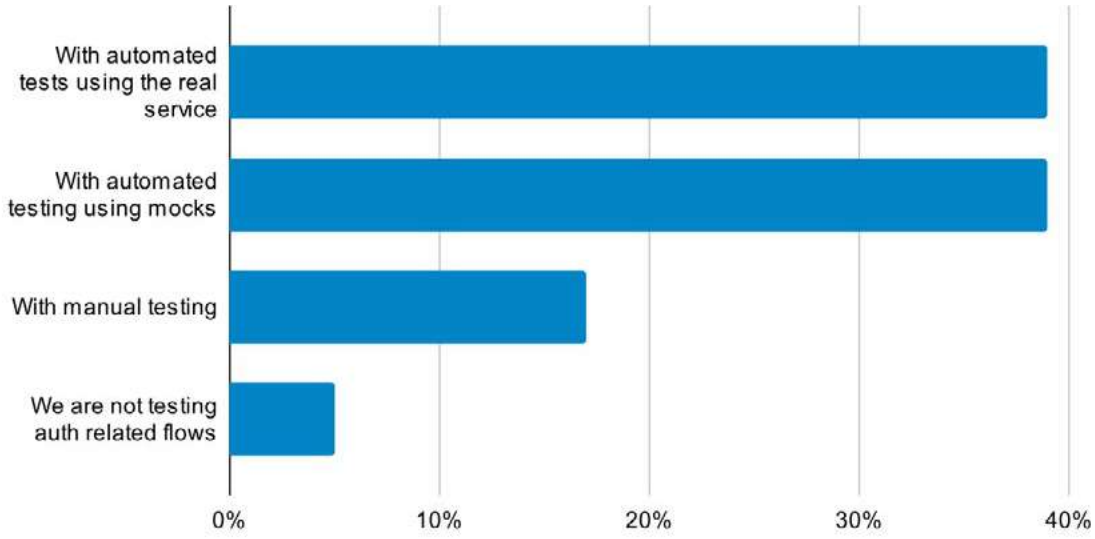
Chart 20. Which of the following types of software do you prefer for your testing environments?



The preference of around half of the respondents for local testing is explained by the fact that many of them are employing automated testing using the real service, versus using mocks which would be required with a SaaS tool.

A small number, 5%, are not testing at all, and 17% are employing manual testing.

Chart 21. How are you testing login and other auth related flows for your application



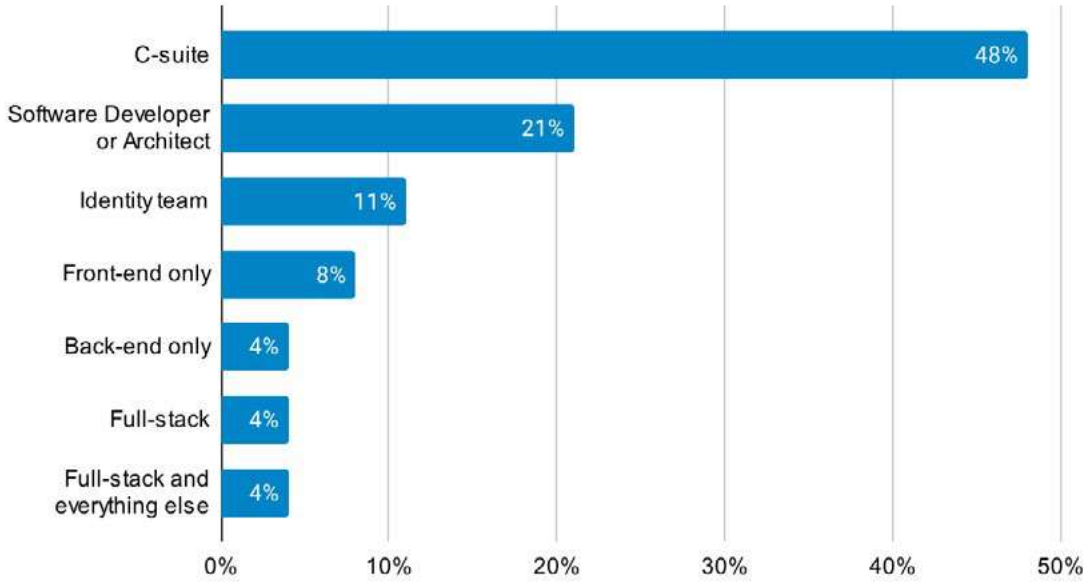
Demographics

For the inaugural State of Homegrown Authentication report, sponsored by FusionAuth and Cloudelligent, a sample of 144 practitioners was asked about their practices with building homegrown authentication.

89% were directly involved in building authentication, and the other 11% worked closely with the responsible team.

Interestingly, only 11% were on a formal identity team, and a full 48% identified as the CTO.

Chart 22. Which best describes your position:



Conclusion

This report suggests the profile of a team building homegrown auth, composed of senior engineers and the C-suite, but not necessarily fully expert in identity. The authentication libraries chosen generally match with the lowest maintenance requirements and the easiest integration, though only for the most commonly used libraries.

The libraries in the middle of the pack switched around when it came to determining which were most efficient and easy.

The report also reveals the profile of a group with needs that remain unmet by the majority of CIAM providers, which force teams down the route of multi-tenant SaaS. Future areas of research would dive deeper into the preferred production deployment models, as well as possible geographic differences.

About FusionAuth

FusionAuth is the only downloadable Customer Identity and Access Management (CIAM) platform with an enterprise-grade, hybrid deployment model for diverse development pipelines.

Trusted by over 450 global organizations, FusionAuth provides customers of any size with a single-tenant VIP suite, the option to download and run anywhere, world-class support, and no hidden costs regardless of scale.

About Cloudelligent

Cloudelligent offers smart solutions to modernize and manage your infrastructure, applications, data, and day-to-day operations using AWS Cloud - while making sure everything is secure.

Empower your business to generate new revenue streams and deliver exceptional experiences by leveraging our cloud expertise.

Thank You