Calling Native "Apps" from the Web

[Combining the Best of Two Worlds]

Although Web technology is constantly improving, native applications remain important. As proof of that, Google and Apple have not retired their native payment applications but rather added a Web interface, which recently also has become a W3C standard known as **PaymentRequest** [https://www.w3.org/TR/payment-request/]. One reason why native payment applications have more or less become the norm is that they are equally usable in non-Web contexts such as *paying in a shop* or *sending money to a friend*. The ability interacting with OS level components like TEEs (Trusted Execution Environments), is also an area where native applications have a clear edge over Web applications.

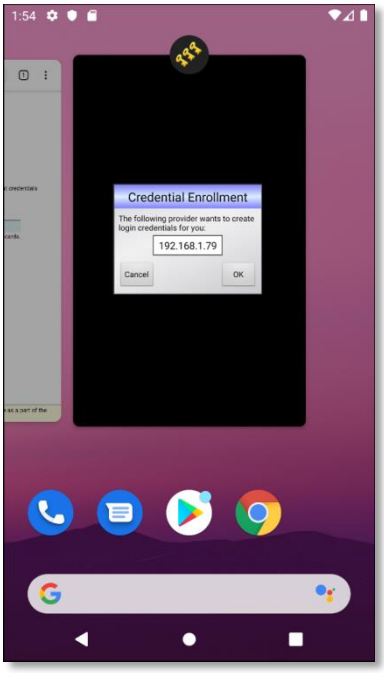
However, payment applications that do not rely on *central providers using preconfigured URLs*, typically need to enroll payment credentials from *any number of independent parties using their respective Web sites*, making purely "App" based enrollment solutions rather impractical. Native Apps also tend to constraint the way users can sign-up and authenticate. That is, there is obviously a need for a counterpart to **PaymentRequest** for other applications than payments as well.

The current, widely deployed "standard" for invoking native Apps from the Web is using custom **URL handlers**. Although working, this solution has (at least as implemented in Android), quite severe limitations including:

- Provides no security context of the calling Web page
- "Fire-and-forget" scheme offering no return value to the calling Web page
- Awkward task management since the App and the calling Web page are (*by design*) disconnected
- A result page invoked by the App may launch *another* browser than you started with, creating session state issues

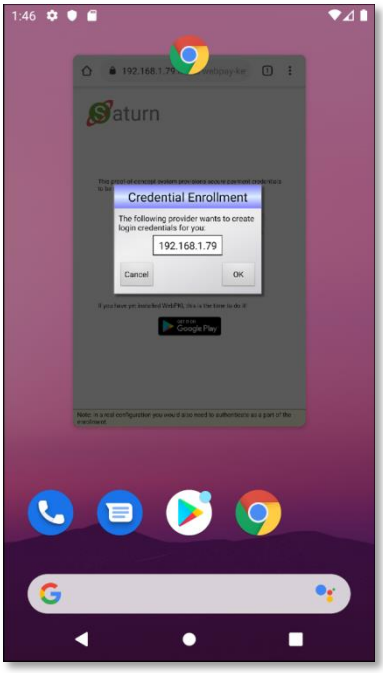
> Since all of these issues (and more) have been fully addressed in **PaymentRequest**, they are presumable relevant.
> *As far as I can tell,* **PaymentRequest** *does not introduce vulnerabilities beyond what* **URL handlers** *already do*.

"Just for fun" (well, not really) I have been exploring using **PaymentRequest** for other applications than it was designed for:



Task view using **URL Handler**

Please ignore the ugliness of the App. It was developed 2013(!) and will UI-wise be rewritten from scratch…



Task view using **PaymentRequest**

As can been seen in the left screenshot, the App (*incorrectly*) shows up as a separate task. Using **PaymentRequest** the called App becomes in all respects (except for DOM access), a part of the calling browser page.

Note: Although this whitepaper talks about payments, *identity related applications are probably an even bigger target*.

V0.13, 2019-09-04, Anders Rundgren, anders.rundgren.net@gmail.com