DC Sec

Leibniz Universität Hannover

Hey, You, Get Off Of My Clipboard

On How Usability Trumps Security in Android Password Managers

Sascha Fahl

Marian Harbach

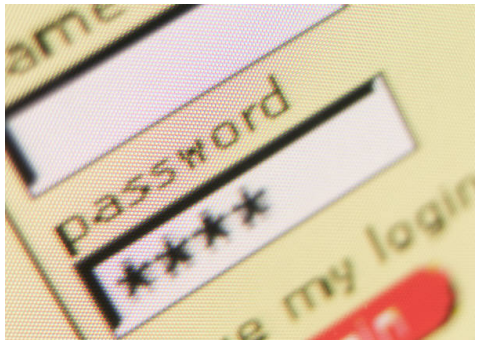
Marten Oltrogge

Thomas Muders

Matthew Smith

Passwords Are Everywhere

- Average user has more than 25 online accounts
- Managing passwords for so many accounts is challenging
- Password Managers are a way out of the dilemma and help users to handle many passwords



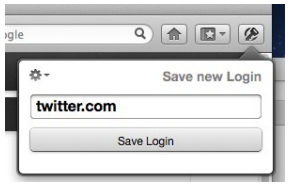
Password Managers for Desktop Browsers

- Users can choose between many different tools
  - Some come with the Browser

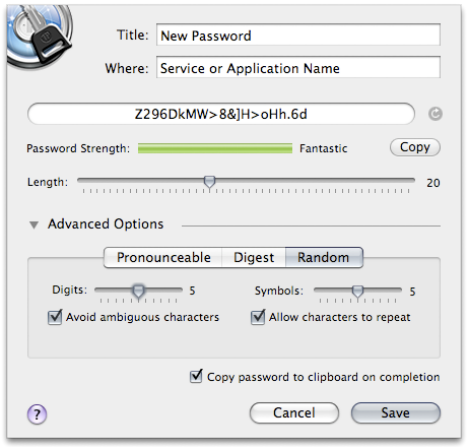
  - Some are third party plugins

Password Managers for Desktop Browsers

- Programming interfaces allow advanced features which support the users' normal workflows
  - Auto Safe



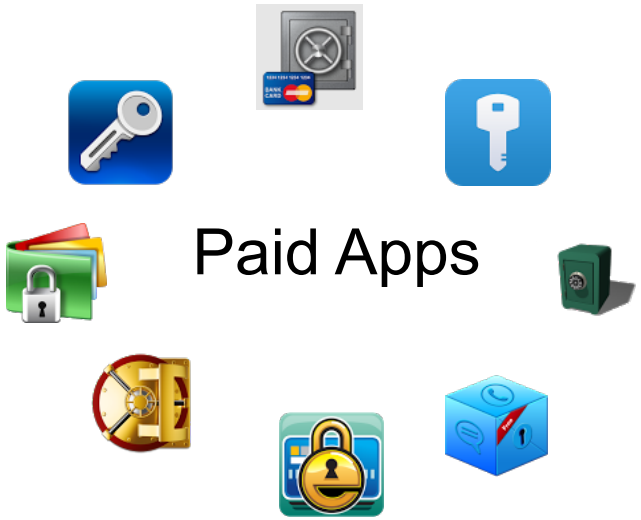
  - Password Generation



  - Auto Fill-in

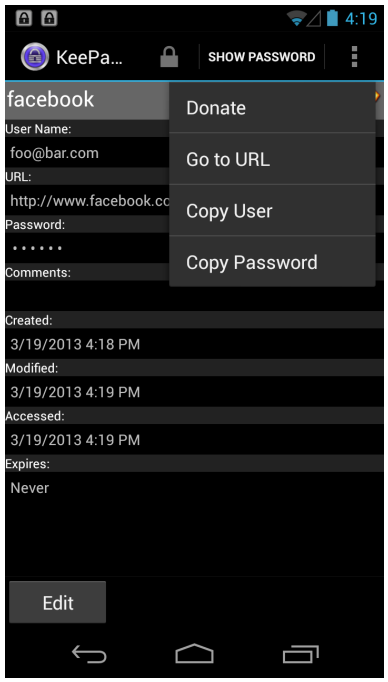
Password Managers on Android

- Android users can choose between many different Apps

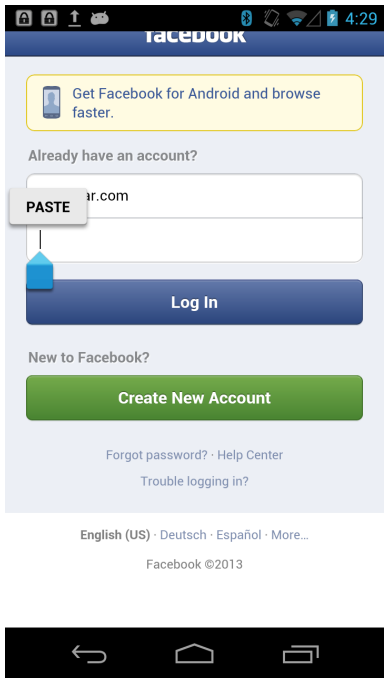


Password Managers on Android

- Android browsers lack plugin API
  - Password Manager Apps do not support auto fill-in
  - Force the user into a dangerous workflow:



switch to target App



copy username and/or password

paste username or password

Attacking the Copy-Paste-Workflow

- Arbitrary Apps can attack the Copy-Paste-Workflow and sniff credentials…
  - … and the attacking App requires zero permissions

- Overview of the attack:
  - Register global listener for clipboard changes
  - On clipboard change, check which App triggered copy operation
    - Is it a PM App?
  - Monitor for foreground App switch
    - New foreground App is assumed to be target of paste operation
  - Move collected information off the device

Attacking the Copy-Paste-Workflow

- Android provides a very “handy” clipboard API:

```
public class PWSpyClipChangedListener implements OnPrimaryClipChangedListener {
   @Override
    public void onPrimaryClipChanged() {
            ClipboardManager cm = (ClipboardManager)
                        getSystemService(CLIPBOARD_SERVICE);
            ClipData cd = cm.getPrimaryClip(); //tada
   }
}
```

- Android allows every App to register such a listener
  - No permission needed

Purpose of Sniffed Credentials

- If the target App is single purpose (e.g. Facebook App), guessing the credentials' purpose is trivial

- If target App is not single purpose (e.g. Browser) guessing the credentials' purpose is *almost* trivial
  - the world-readable */proc/net/tcp* file lists all active network connections
  - checking all active network connections just after the paste operation supports the attacker's guesswork
  - again, no permissions are required

```
18: EA104B82:01BB 2F5C154D:C2EF 06 (
19: EA104B82:01BB 84344B82:D953 06 (
20: 0100007F:E5BA 0100007F:01BB 06 (
21: EA104B82:01BB 30824C5C:02DB 06 (
22: EA104B82:01BB D7F84B82:C617 06 (
23: EA104B82:01BB 632E4B82:CA89 06 (
24: EA104B82:01BB D7F84B82:C616 06 (
25: EA104B82:01BB D7F84B82:C60B 06 (
26: EA104B82:01BB 5891B659:0459 06 (
```

Sending Out Credentials

- For now we have collected credentials and their purpose

- Another Android “feature” allows the attacker to send out the sniffed information even without requesting the INTERNET_PERMISSION
  - wait until the phone switches to stand-by mode
  - invisibly open Android’s stock browser
  - transport the sniffed information in a HTTP GET request
  - close the browser window using the server’s response and a custom protocol

Scale of The Attack

- We analyzed 13 free and 8 paid password manager apps on Android

- Installed all apps on an Android 4.0 device
  - All apps provide the Copy-Paste-Workflow for credentials
  - All apps are vulnerable to our attack!

- We wanted to know what the developers think about this issue.

Interviews With Developers

Informed all developers about the security threats and asked them to participate in an interview

- 15 of 21 developers agreed

Central questions:

- Why was the C&P feature added to their PM apps?
- Were developers aware of the security threats and, if so, why did they add the C&P feature nonetheless?
- Which features, if any, do developers miss in Android's SDK for developing PM apps?

Interviews With Developers - Results

Why was the C&P feature added to their PM apps?

- Identified three reasons, user demand was most important:

  "The feature was highly requested by users. The most common example: users want to login to a website on their mobile device, so he/she copies credentials from [our PM] to the clipboard and then pastes them into the browser."; P15

Were developers aware of the security threats, and, if so, why did they add the C&P feature nonetheless?

- All but one developer were aware of the threats:

  "It's a balance between ease of use and security. Of course it would be much more secure to not use the clipboard, however people accept the risk of doing so; the alternative of not using a password manager is worse."; P3

Which features, if any, do developers miss in Android's SDK for developing PM apps?

- All developers complained about an appropriate plugin API for mobile browsers:

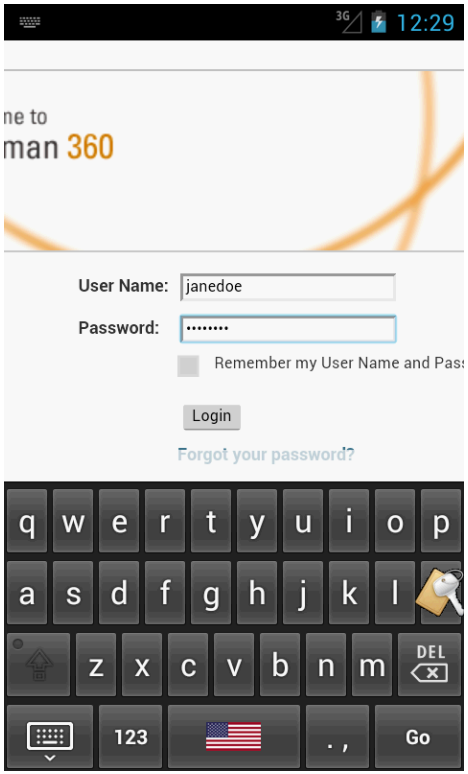
  "Android doesn't offer hooks into the native default browser [. . .] and does not allow our app to access input fields of other apps [. . .] which makes it necessary that password managers make heavy use of the clipboard."; P3

Possible Solution

To avoid heavy usage of the insecure C&P API on Android, use a customized software keyboard instead.

USecPassBoard

- Replaces the default keyboard
- secure and usable
- is available in every app
- has access to an app's input fields only on the user's discretion

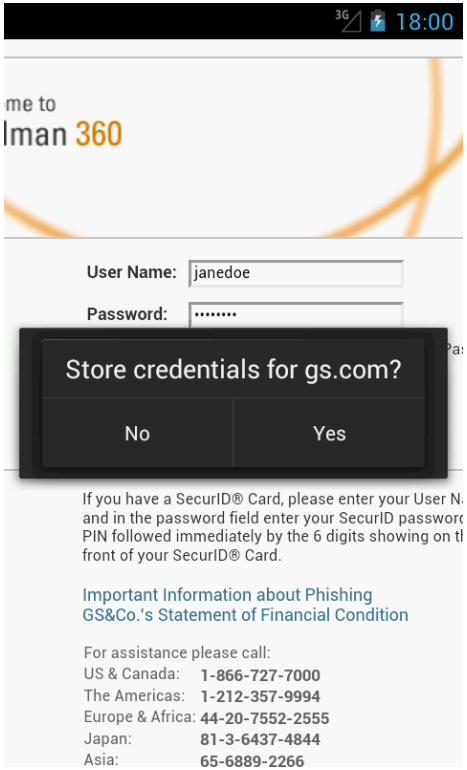


USecPassBoard – Store Credentials

USecPassBoard asks to save credentials for a new context

Context

- A context is either an app or a website for which credentials are valid
- App-Contexts are identified by the App's unique package name
- Website-Contexts are identified by the browser's package name and the currently active website

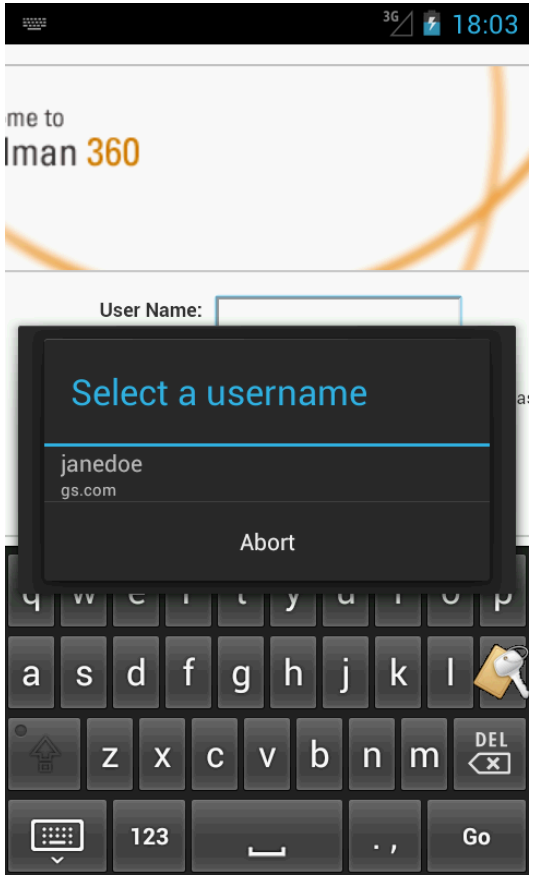


USecPassBoard – Fillin Credentials

USecPassBoard asks to fill-in credentials for a stored context

Fill-in

- After selecting an input field, stored credentials can be selected
- Only valid credentials are listed for a context



USecPassBoard – Security

- The keyboard is available for every app
  - However, credentials are bound to a context
  - Contexts are strongly connected to unique package names/websites
  - The channel between the keyboard and the target app is not accessible by other apps
- The credential database is AES encrypted and requires the user to enter a master-key for unlocking

Summary

- The lack of plugin APIs causes PM apps to make heavy use of the system's clipboard.
- Current PM app implementations are vulnerable to credential sniffing attacks trough the copy-paste-workflow.
- Most developers were aware of possible security threats, but argue that abandoning the feature will harm their users' security.
- USecPassBoard is a proof-of-concept solution that provides security and usability and avoids using the insecure clipboard.

Outlook

- USecPassBoard is a possible solution which should be extended and improved in future work.
  - USecPassBoard does not need API changes.

- With the support of Google, a comfortable plugin API for password manager apps in browsers and other apps similar to the desktop would be feasible and preferable.