Project:

- Few Guidelines:
  1. Group of 2 or 3 recommended. Group of 1 is not recommended and group of 4 is not allowed.
  2. Demo after exam i.e. 12 Dec, 13 and report due on 15 Dec, 2013
  3. The flow will be Coding, Evaluation, Demo (40%) and Report (60%)
  4. Report is expected to include demo discussion
  5. The students participating as tester for project 2 will be given extra credits i.e. 2 lecture notes. The test should not exceed 2 hours. Number of participants 70 max and min 12 participant per password type. People outside class could be taken

Three project options

1. User-driven disclosure for Android

It is an ambitious project and might lead to a publication. Out of the two major components, if one is done then it’s great

Inspired by:


TaintDroid: It is a modified version of Dalvik virtual machine as used on android and it does run time taint tracking in virtual machine.

It maintains for every byte of memory inside the Virtual Machine, it maintains a tag, like another byte and the value of the other byte tells us about source of the data

So if a program reads a stream of input from the network

Then taint joy will maintain a shadow array of bytes

Array of bytes:

| X | X | X | X | X |

Shadow array of bytes: It’s not accessible to the program

| N | N | N | N | N | N | N | N | N |
It can keep track of other things also like GPS location. GPS is stored in an object and object is stored in some contiguous location of memory.

<table>
<thead>
<tr>
<th>Location data</th>
<th>Location data</th>
<th>Location data</th>
<th>Location data</th>
<th>Location data</th>
<th>Location data</th>
<th>Location data</th>
</tr>
</thead>
</table>

GPS tainted bytes which will check if any unusual activity occurs

| G | G | G | G | G | G | G |

TaintDroid should take care of what information is sent over the network. So it will take the data and notify what is transferred exactly.

**What is missing as of now?**

→ Enforcement

So in this project basically turn taint joy into enforcement tool. So it will involve designing policy of what should be transferred and how much.

On Android when installation is performed, it looks at permission and if we can allow if we want but we cannot modify those permissions. Android permission don’t allow information flow.

In IOS it doesn’t ask during install time but get pop up during first time it access few data like GPS, Camera etc.

So there is no way to say that allow some functionalities and not others like allow network access and not GPS.

A project before was based on granularity of the access like allow GPS till some few miles and similarly like camera data or microphone data not accessed but allowed during a call etc.

- **2nd idea:** Usually the buttons are drawn by the application itself. In this paper “User-Driven Access Control: Rethinking Permission Granting in Modern Operating Systems,” Franziska Roesner, Tadayoshi Kohno, Alexander Moshchuk, Bryan Parno, Helen J. Wang, Crispin Cowan”, they have buttons which are managed by the trusted Operating System and when the button is clicked, the user wants that application to be accessed and hence the OS checks the permissions and allow it thereafter permanently or temporarily.

**Basically combine both ideas**

Suppose we click on the below button,

![Button](image)

It is clear that we need access to GPS. Click will generate following events.
1st: privilege to read GPS and 2nd grant permission to write to network and 3rd allow temporary exception where it can take GPS data and allow for the moment to send GPS data over the network. So notify the VM that temporarily it is allowed

TaintDroid is Open Source

It’s a big project, so u can do a part of it like
→ Modify the TaintDroid system
→ Infrastructure of OS managed button, OS securely receive user input and make dummy call to modified VM

2. Project 2: Memorable but incommunicable passwords

- Inspired by research paper “Neuroscience Meets Cryptography: Designing Crypto Primitives Secure Against Rubber Hose Attacks,” Hristo Bojinov, Daniel Sanchez, Paul Reber, Dan Boneh, and Patrick Lincoln.” like a method for authenticating users (like password) which will be secured against rubber host crypt analysis (it means beating up the person who knows the password and get it).
- The idea was the way the user would log in depends on the game playing skills and it is unique to everyone. So the uniqueness can’t be transferred to other user. But the user can be coerced though.

- The other inspiration is from the paper “PhorceField: A Phish-Proof Password Ceremony,” Michael Hart, Claude Castille, Manoj Harpalani, Jonathan Toohill, and Rob Johnson.”. The paper came up with a password system if you visited the right website then you will be able to enter password easily but if u visit a phishing website then it will be difficult to enter the password (it will be confusing). So the system was, when you visit the right website then it is given an access to small set of 12 images restricted to the user. So click on those images in certain manner. But when you visit the wrong website then the wrong website won’t have access to those 12 images. So it will show you random images which doesn’t belongs to your password.
- Concrete nouns is something which describe the image completely. A collection of images which are easy to detect and type, but difficult to describe it to others like Faces. So images related to faces is a nice option. Other example are landscape

So project would be to collect images like faces, landscape and then write a small website and students in class can be recruited

Two test performed:

1. Can people remember the images (easy to do)
2. And can they describe it to others (not trivial)

So the subject will be given a password based on set of images like either faces or landscape. The aim is to keep track of how correctly the subject can remember it by testing it in duration of 2 days, or a week. And also test how the user can describe the password to phisher
Give phisher good power, i.e. as much as possible. So of course user can’t see the images and user should describe it to phisher.

Phisher can do the following activities to decode the password:

1. Text box to add description
2. Skype and discuss
3. Search in database of images and show it to user

The aim is to calculate the time the victim takes to communicate it to the phisher

The tests will be like give same password to a user and different password to different users and calculate this kind of methodology. Try different methodologies to compute the system which is largely immune to the attacks while authenticating users

Q How extensive the data should be and how to demonstrate statistics?
A 70 maximum per password and 12 minimum per password

Subjects of password image type 1 and type 2. So how long does it takes to type those passwords

So distributions might look like:

The first graph shows two totally distinct curves and easily separable

The second graph has overlapping curves, so a clear conclusion is difficult to be drawn
3. **Project 3: Android Fuzzing Tool**

Build a fuzzer (a tool to test other piece of software) for Android API. It gives input, generated randomly or semi randomly, to other piece of software and see the reaction

It is used in network by creating random virtual users and random inputs to test how often and when the server crashes

**Different types of Fuzzer:**

1. **Linux fuzzer (Trinity):** It makes random Linux kernel system and random arguments to check of the kernel crashes. It runs in a virtual machine and log all the data outside VM.
   Random fuzzing gives very poor code coverage
   SMTP or FTP protocol. They take commands like EHLO, RCPT(recipient)
   sekar@cs.stonybrook.edu. The probability of EHLO is very small. So it won’t even reach RCPT and so depth won’t be reached
   So we need intelligence. If SMTP knew about the command then it can have variation in it like 34 different domain names in the mail id
   The project is not to develop a fuzzer. It should not generate random fingers events and see the result. It’s should be like an app which runs on device and run all sorts of system calls and see the output

Q. Is it necessary the application should run with root privilege?
A. Most android phones are not rooted. Most application don’t run with root privilege. No root privilege needed. If privilege is there

Example of cases:
If can take only GPS and run fuzzer when we don’t have GPS permission and see if they get violated and vice versa

Java Fuzz could be useful

So specific application can be taken into consideration and fuzzer can be tested on it

**Goal:** 1. Run Fuzzer  2. Run it for a while and find new bugs
Q. Should the project 3 be limited to Java VM (because it won't be able to detect attacks like buffer overflow etc there) or Dalvik?
A. Java is good, Dalvik is awesome

Q. Why the 3rd project is not trivial?
A. Fuzz an entire subsystem or manager class like Bluetooth manager or WiFi manager. It should have some intelligence

Q. Intelligence can be learnt or coded?
A. Coded. Intelligence would be good.

Q. Are there any specific bugs we need to take care beforehand?
A. When you demo, a set of security which were supposed to be caught should be caught. It should not be customized to the specific bug. It could be used for evaluation

Q. What if don't find anything?
A. Evaluation will be based on quality fuzzing tool. The more intelligence it has about the API it is trying to fuzz, it will be good

❖ Demo Details:
  • Demo flowchart project 1:
    Design and Implementation of modified version of Dalvik VM and demo of working
  • Demo flowchart project 3:
    1. Design and Implementation
    2. Website with password, interaction protocol. How the study was designed. Results. Statistical tests
  • Demo flowchart project 3:
    What subsystem we decided to fuzz. What intelligence encoded

Q. Do we need to work on latest Android Sources?
A. For fuzzing it will be nice to take latest version but not required
For first version work with TaintDroid. It might not work in KitKat version
Q. Should the application draw the buttons?
A. New application needs to be designed. Like a small application.

Q. How the subsystem know what is the functionality?
A. OS services should be given command about the button and its application. The secure subsystem has a small collection of icons which are cleared by the user.

Q. Is the application permission still needed?
A. In the long run, it will be redundant. In this project, let the permission be there. While demo, just make modifications as per the need arise.

Q. Could it be limited to buttons or links and text should be included?
A. It could be limited to buttons.

---

**Clarification of Deputy:**

In deputy we can write annotations like:

```c
myfunc(int *COUNT(n) p, int n){
  assert(o<=i && i<n);
  p[i]==z;
}
```

All the extra code the deputy inserts is simply an assertion like this and they don’t change the state. Either it terminates or leave the program in the same state. State of all the data is unchanged.

**Two phases of Deputy:**

1. Insert the assertion to make program safe
2. Try to optimize, just eliminate things and make it faster.

In Jones and Kelly when linked to non-Jones & Kelly it, it had extra metadata and we need to update the metadata.

**Updating of variables:**

Imagine we have a parallel assignment:

```c
myfunc(int *COUNT(n) p, int n){
  p=p+1;
}
```
n=n-1: }

To prove the parallel assignment, we will have to prove the following:

\[ p: \text{ptr}(n) \text{ int} \]  // (it should be valid with both new values \( p, n \) also)

\[ p+1 : \text{ptr}(n-1) \text{ int} \]

It remembers it like this

\[ p: \text{ptr}(p, p+n) \text{ int} \]

\[ p+1: \text{ptr}(p, p+n) \text{ int} \]

\[ p+1: \text{ptr}(p+1, p+n) \text{ int} \]

\[ p+1: \text{ptr}(p+1, p+1+n-1) \text{ int} \]

**Subsumption Rule:**

\[ p: \text{ptr}(a, b) \text{ int} \quad a \leq c \leq d \leq b \]

\[ p: \text{ptr}(c, d) \text{ int} \]

\[ p: \text{ptr}(a, b) \text{ int} \quad k : \text{int} \]

\[ p+k: \text{ptr}(a, b) \text{ int} \]

This is how mutation is handled

---------------------------------------------------------X----X--------------------------------------------------------------------------

---------------------------------------------------------X----X--------------------------------------------------------------------------