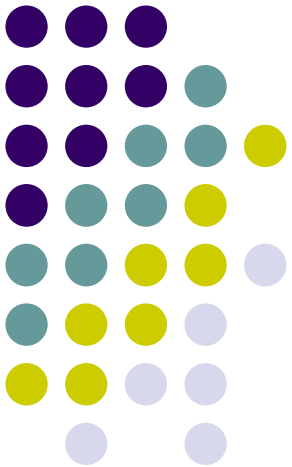
Ubiquitous and Mobile Computing CS 528: Curbing Mobile Malware based on User-Transparent Hand Movements

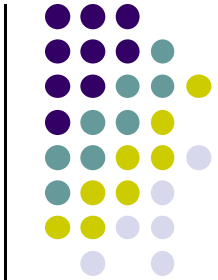
Zhaochen Ding

Computer Science Dept.

Worcester Polytechnic Institute (WPI)

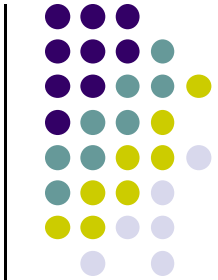


Society Background



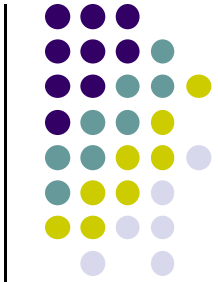
- Present Situation
  - Mobile devices becoming ubiquitous
  - Various functionality in addition to traditional capabilities
  - Consumers willing to use, developers willing to design
- Result
  - MALWARE

New Attempt



- Fact
  - Mobile malfare tends to attack in scenarios where the device user has no intention to access the underlying services
  - If the user's intent to access the services can be captured in some way, these attacks could be prevented
- Solution
  - Elicit a user's intent via gestures that are transparently and naturally performed by the user prior to accessing the services. In other words, whenever the user wants to access the service, she will naturally exhibit a particular gesture.
  - On the other hand, if the malware attempts to access the service, the gesture will be missing and the access request can be blocked

Project Background

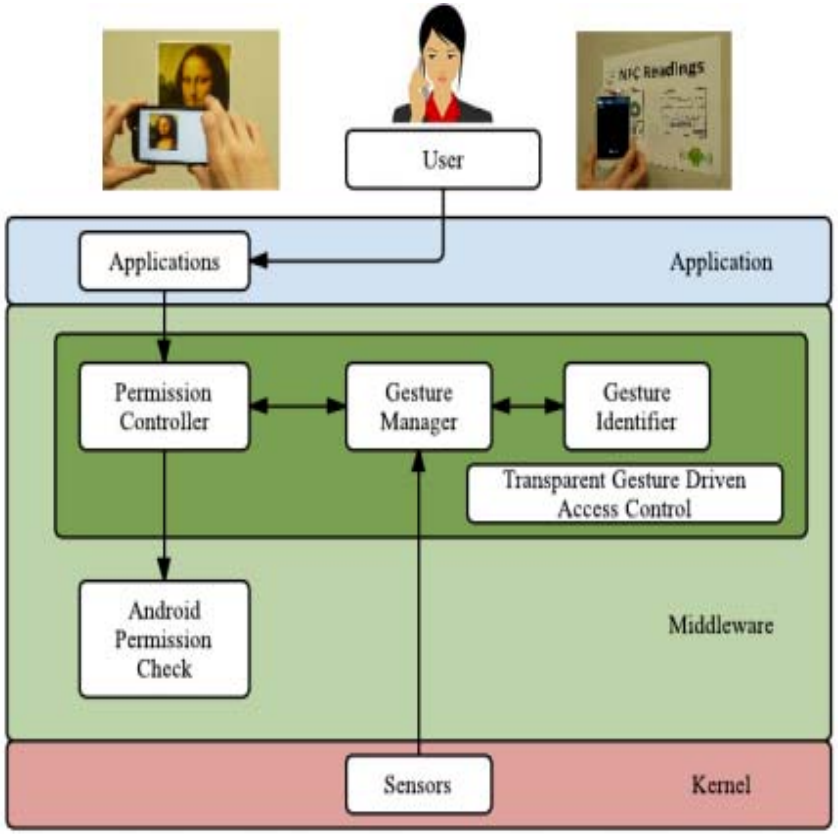


- Threat Model
  - Assume that the OS kernel is healthy and is immune to the malware infection
  - Assume that hardware is immune from the malware and the malware cannot manipulate the device's onboard sensors
  - The eventual goal of the malware is to access the device's sensitive resources/services to make premium phone calls, take pictures of user's surroundings or read nearby NFC enabled credit cards/tags .
- Design Goals and Metrics
  - The scheme should be lightweight in terms of memory, computation and power consumption.
  - The approach should be efficient, i.e., not incur a perceptible delay. If a user has to wait for a perceptibly long time while using the scheme, this will affect the overall usability.
  - The scheme should be robust to errors.
  - The approach should be transparent to the users.

Curbing Malware Using Transparent Gestures

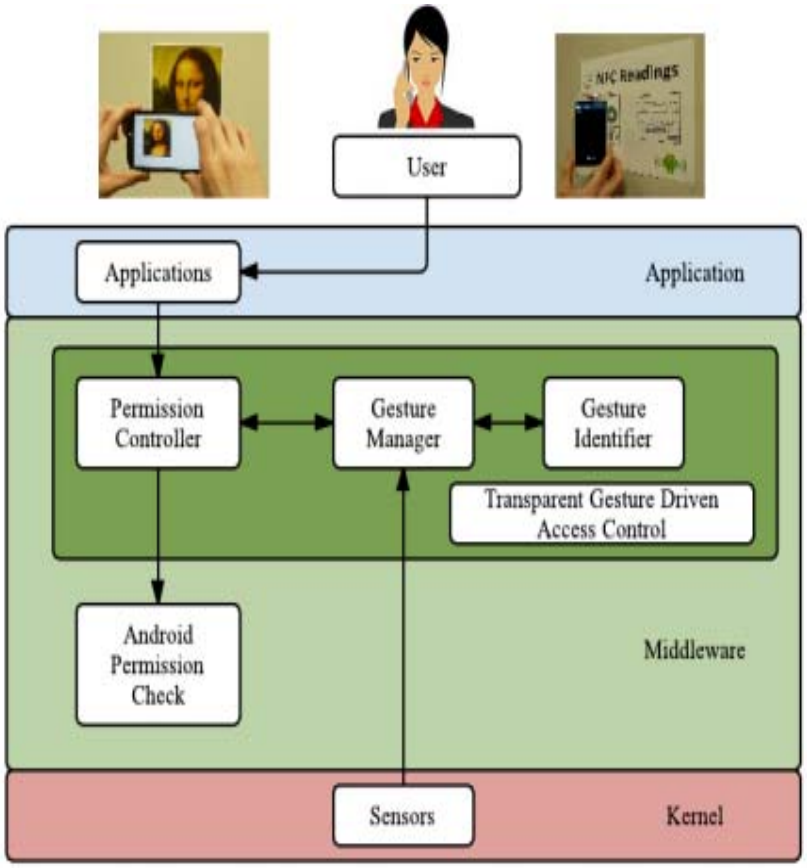
- Overview
  - Motion Sensors
    - Accelerometer and gyroscope
  - Position Sensors
    - Magnetometer and orientation sensors
  - Environmental Sensors
    - Temperature, pressure and illuminance
- These sensors are used to identify if the phone has been moved in a way that the user is trying to perform certain activities
- Hand Movement
  - Voice Dialing
  - Camera
  - NFC

Curbing Malware Using Transparent Gestures



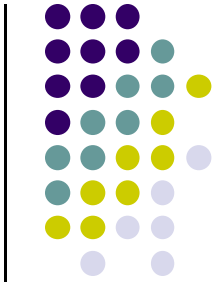
- System Model
- Add another layer of permission control on top of the original Android permission granting system.
- When an application requests a resource access, such as to make phone call or use camera hardware or use NFC, android checks the resource access permission associated with the app before allowing any access to the resource.
- If the user has provided the app with the permission it needs during the installation time, the app is allowed to access these resources.

Curbing Malware Using Transparent Gestures



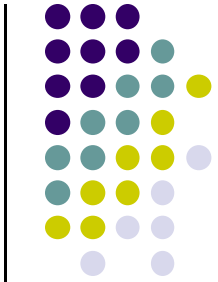
- Three Entities
  - **Gesture Identifier** which is a trained classifier that can identify a gesture
  - **Permission Controller** that checks for the permission token
  - **Gesture Manager** that communicates with the hardware, gets the phone movement data from the sensors and provides the analysis results to the Permission Controller

Data Collection: App Design



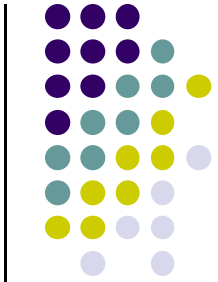
- Call App
  - The phone calling intent is triggered when the call is initiated/answered and the state becomes “OFFHOOK”. With this intent, the app starts recording the sensor data. This means that the user has made/answered the phone and the associated Call gesture, i.e., the motion to bring the phone to the ear, has been initiated.
- Snap App
  - In the Android system, the camera hardware can be used by any application that is registered with the permission to use it in its manifest file, i.e., there is no system-wide intent that can be intercepted by another application any time the camera is in use. Because of this restriction, a custom application which is to be used as an alternative to other camera applications to test our hand movement gestures is developed.
- Tap App
  - For reading NFC, our app waits for the NFC intent and starts recording the sensor data as soon as this intent is captured.
- Snoop and Control App
  - Along with different gestures, the app system also records data from various sensors at random point of times in order to compare these gestures with other activities. We refer to this data as “Snoop”. It provides a text box for the tester to specify the activity being performed and once the button is pressed for recording, it records all the sensor data for ten seconds.

Data Collection: Procedures



- Data Collection Procedures
  - App distributed to the users in the team's respective Universities in US and Finland.
  - Before distributing the app to these users, we explained what sensor data was being recorded, for how long it was recorded, at what occasions it was recorded and for what purpose the data was being used
  - There were a total of 23 users recruited for our study. They were students of the Computer Science departments of the two Universities
  - Experiment for the Call gesture was performed in realworld settings, i.e., the data was recorded when the volunteers made or received calls under normal use. The app notifies the user whenever the sensor data is being recorded by displaying an icon with a message in the notification bar
  - In a similar manner, we provided the Snap App to the users to collect data in real-world settings. They were asked to take pictures using our Snap App instead of the original camera app conforming to real-world settings.
  - For the Tap experiment, we provided our Tap App to the volunteers. All of our participants possessed NFC enabled smartphones.

Gesture Detection： Design and Evaluation



- Call Detection
  - User-Specific Model
  - Device-Specific Model
  - Generalized Model
- Snap Detection
  - User-Specific Model
  - Device-Specific Model
  - Generalized Model
- Tap Detection
  - User-Specific Model
  - Device-Specific Model
  - Generalized Model

$$precision = \frac{TP}{TP + FP}; \quad recall = \frac{TP}{TP + FN};$$

$$F\text{-}measure = 2 * \frac{precision * recall}{precision + recall}$$

The best features and the measurement values are calculated from running a 10-fold cross validation

Gesture Detection : Call Detection (14 users)

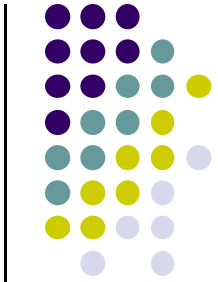


TABLE II. RESULTS OF USING THE OPTIMAL FEATURE SUBSET IN THE CLASSIFICATION OF CALL AND OTHERS

Classification Model	User ID /Device	Classifier	Features Subset	Precision	Recall	F-Measure
User-Specific	1	L	A,G,LA,P	0.91	0.97	0.94
	2	SL	A,G,LA,O,P	0.97	0.93	0.95
	4	RT	GR,Gy,LA	0.87	0.90	0.89
	5	RT	GR,LA,O,P,R	0.87	0.83	0.85
	6	RF	G,LA,P,R	0.83	1.00	0.91
	7	SL	G,LA,M,P	0.88	0.97	0.92
	9	RF	A,Gy,M,P,R	0.94	0.97	0.95
	11	RT	A,G,Gy,M,P,R	0.85	0.97	0.91
	12	RF	G,Gy,LA,M,P	0.79	0.90	0.84
	13	RF	G,GR,M,P	1.00	0.97	0.98
	14	RF	GR,LA,M,R	0.97	1.00	0.98
	15	RF	LA,M,P,R	0.88	0.93	0.90
	16	L	G,Gy,M,P,R	0.97	1.00	0.98
	20	RF	G,Gy,P	0.88	0.93	0.90
Device-Specific	Google Nexus	RF	G,Gy,LA,R	0.90	0.83	0.86
	Samsung Galaxy	RF	A,GR,G,Gy, P,R	0.83	0.88	0.86
	HTC	RF	GR,LA,M,R	0.97	1.00	0.98
Generalized	ALL	RF	A,GR,LA,M,R	0.92	0.83	0.87

Gesture Detection : Snap Detection (19 users)

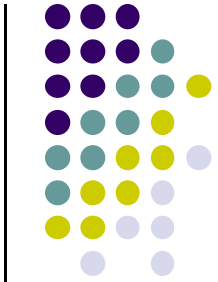


TABLE III. RESULTS OF USING THE OPTIMAL FEATURE SUBSET FOR THE CLASSIFICATION OF SNAP AND OTHERS

Classification Model	User ID /Device	Classifier	Features Subset	Precision	Recall	F-Measure
User-Specific	1	RF	P,Gy,A,R,LA	0.97	1.00	0.98
	2	SL	Gy,O,LA	0.97	1.00	0.98
	3	NB	M,P,GR	0.97	1.00	0.98
	4	L	G,P,M,GR	1.00	1.00	1.00
	5	NB	A,M,GR	1.00	1.00	1.00
	7	RF	A,P,M	0.97	1.00	0.98
	8	NB	O,A,M	1.00	1.00	1.00
	9	L	A,M,LA,GR	0.97	1.00	0.98
	10	RF	GR,M,O,P	0.97	1.00	0.99
	11	RF	G,Gy,P	1.00	1.00	1.00
	12	RF	A,G,Gy,P,R	1.00	1.00	1.00
	13	RF	GR,P	0.97	0.93	0.95
	14	RT	A,GR,G,R	0.91	1.00	0.95
	16	SMO	M,P,R	1.00	0.97	0.98
	18	L	A,LA,P	0.97	0.97	0.97
	19	RF	GR,Gy,P	0.97	1.00	0.99
	21	RT	G,LA,O,P,R	1.00	1.00	1.00
	22	RF	GR,Gy,P	1.00	1.00	1.00
	23	NB	A,GR,M,R	1.00	0.97	0.98
Device-Specific	Google Nexus	RF	A,GR,G,M, O,P	0.89	0.95	0.92
	Samsung Galaxy	RF	G,Gy,M,O,P	0.96	0.98	0.97
	HTC	RT	A,GR,G,R	0.91	1.00	0.95
	LG	RF	LA,M,P,R	1.00	0.98	0.99
Generalized	ALL	RF	A,LA,M,O,P	0.89	0.93	0.91

Gesture Detection : Tap Detection (20 users)

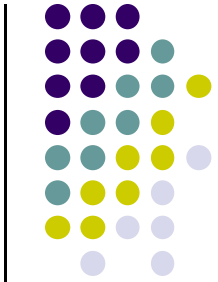
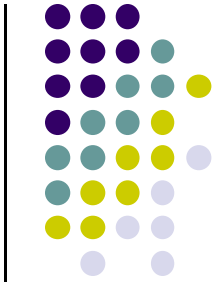


TABLE IV. RESULTS OF USING THE OPTIMAL FEATURE SUBSET FOR THE CLASSIFICATION OF TAP AND OTHERS

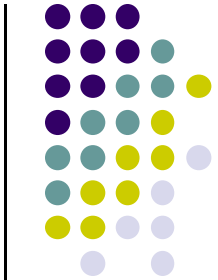
Classification Model	User ID /Device	Classifier	Features Subset	Precision	Recall	F-Measure
User-Specific	1	RT	G,M,GR	1.00	0.97	0.98
	2	NB	P,M	1.00	0.97	0.98
	3	RT	P,A,GR	1.00	0.97	0.98
	4	NB	GY,M,A	1.00	0.97	0.98
	5	NB	P,M	1.00	1.00	1.00
	6	RF	G,R,,GR	1.00	1.00	1.00
	7	RF	P,O,M	0.97	1.00	0.98
	8	NB	P,M	1.00	1.00	1.00
	9	NB	GR,M,P	0.94	1.00	0.97
	11	NB	A,Gy,M	0.97	1.00	0.98
	12	NB	A,G,O,P,R	1.00	1.00	1.00
	13	RF	O,P,R	0.92	0.96	0.94
	14	RF	A,LA,M	0.82	0.93	0.88
	16	RF	GR,M,O,P,R	1.00	0.97	0.98
	17	RF	A,GR,G,M,O	0.91	0.97	0.94
	18	RF	A,GR,G,M,O	0.97	1.00	0.99
	19	L	GR,G,O,P	1.00	1.00	1.00
	21	RT	M,P	0.97	1.00	0.98
	22	RF	Gy,O,P	1.00	1.00	1.00
	23	RT	GR,Gy,P	0.86	0.97	0.91
Device-Specific	Google Nexus	RF	GR,G,Gy,M, O,P,R	0.93	0.92	0.92
	Samsung Galaxy	RF	LA,M,O,P,R	0.96	0.97	0.96
	HTC	RF	A,G,LA,M	0.82	0.93	0.88
	LG	RF	A,G,M,O,P	0.96	1.00	0.98
Generalized	ALL	RF	GR,G,M,O,P	0.89	0.90	0.89

Discussion



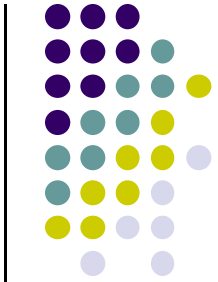
- Local vs Remote Classification
  - Local Classification may require more resources for the testing task
  - Remote Classification does not require extra resources for testing the gesture and running the classifier but needs data collection and trust to the remote service
- Power Efficiency
  - As the gesture detection procedures in our approach lasts for no more than a few seconds, the approach is indeed quite power-efficient.
- Quarantine State(NFC)
  - To address this situation, we rely upon lazy authentication, i.e., the tag information is read by the OS but kept in a quarantine state until the full tapping gesture is recorded and analyzed
- Fall-Back
  - In these scenarios, when the gesture detection mechanism fails, there is a need to fall back to allow the user to access the desired resource. This can be solved either by prompting the user to press a “Yes/No” button, or by asking the user for the explicit gestures such as hand-waving. In situations where users are not able to make the gestures, for example under extreme emergency, a voice command could be used.

Discussion



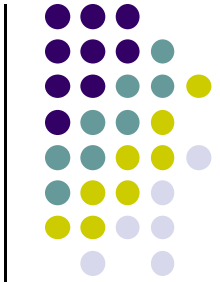
- Benign Automated Access
  - For these kinds of services which need automation, the permission token can be provided to the app at the time when the user provides the gesture associated with it, though the related activity needs to be performed at some later point of time.
- Social Engineering Attacks
  - However, it is possible for a malware developer to perform social engineering attack to fool the user into providing a valid gesture
    - For example, malware developer can design a game which asks the user to move the phone in ways that emulate either Call, Snap or Tap.

Related Work



- The majority of related prior work focuses on either preventing a device from getting infected or detecting a malware as soon as it has been installed on the device. The two most common defense approaches to detect malware are static analysis and dynamic analysis Both detection techniques have their own drawbacks. Malware authors can avoid static analysis detection through simple obfuscation, polymorphism and packing techniques. Dynamic analysis, although more resilient, is still a posteriori approach which is quite risky to adopt since malicious parties would have already obtained the valuable information.

Future Work



- In the future, gestures associated with other smartphone services, such as sending SMS or email, or web browsing, can also be integrated with this system. As new sensors become available on these smart devices, subsequent work may use the different sensors to identify other transparent gestures and further improve the accuracy for the calling, snapping and tapping gestures

References

- [1] W. Augustinowicz. Trojan horse electronic pickpocket demo by identity stronghold. Available online at http://www.youtube.com/watch?v=eEcz0XszEic, June 2011. [2] M. Ballano. Android threats getting steamy, 2011. http://www.symantec.com/connect/blogs/android-threats-getting-steamy. [3] A. Bose, X. Hu, K. G. Shin, and T. Park. Behavioral detection of malware on mobile handsets. In Mobile Systems, Applications, and Services (MobiSys), 2008. [4] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani. Crowdroid: Behaviorbased malware detection system for android. In Security and Privacy in Smartphones and Mobile Devices (SPSM), 2011. [5] A. Chaugule, Z. Xu, and S. Zhu. A specification based intrusion detection framework for mobile phones. In Applied Cryptography and Network Security (ACNS), 2011. [6] M. Conti, I. Zachia-Zlatea, and B. Crispo. Mind how you answer me!: transparently authenticating the user of a smartphone when answering or placing a call. In ACM Symposium on Information, Computer and Communications Security (ASIACCS), 2011. [7] F-Secure. Bluetooth-worm:symbos/cabir. Available online at http://www.f-secure.com/v-descs/cabir.shtml. [8] F-Secure. Worm:symbos/commwarrior. Available online at http://www.fsecure.com/v-descs/commwarrior.shtml. [9] J. Figura. Machine learning for google android. Available online at http://www.cestina.cz/obo/vyuka/projekty/figura-ml-for-android.pdf. [10] J. Han, E. Owusu, L. Nguyen, A. Perrig, and J. Zhang. Accomplice: Location inference using accelerometers on smartphones. In Communication Systems and Networks (COMSNETS), 2012. [11] S. Kolesnikov-Jessop. Hackers go after the smartphone, 2011. www.nytimes.com/2011/02/14/technology/14iht-srprivacy14.html. [12] H. Li, D. Ma, N. Saxena, B. Shrestha, and Y. Zhu. Tap-wave-rub: Lightweight malware prevention for smartphones using intuitive human gestures. In Conference on Security and Privacy in Wireless and Mobile Networks (WiSec), 2013. [13] P. Marquardt, A. Verma, H. Carter, and P. Traynor. (sp)iphone: Decoding vibrations from nearby keyboards using mobile phone accelerometers. In Conference on Computer and Communications Security, CCS, 2011. [14] Microsoft. What is user account control?, 2011. http://windows.microsoft.com/en-US/windows-vista/What-is-UserAccount-Control. [15] J. Oberheide, E. Cooke, and F. Jahanian. Cloudav: N-version antivirus in the network cloud. In USENIX Security Symposium, 2008. [16] J. Oberheide, K. Veeraraghavan, E. Cooke, J. Flinn, and F. Jahanian. Virtualized in-cloud security services for mobile devices. In In Virtualization in Mobile Computing, MobiVirt, 2008. [17] E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang. Accessory: Password inference using accelerometers on smartphones. In Mobile Computing Systems & Applications, HotMobile, 2012. [18] N. L. Petroni, Jr. and M. Hicks. Automated detection of persistent kernel control-flow attacks. In Conference on Computer and Communications Security (CCS), 2007. [19] F. Roesner, T. Kohno, A. Moshchuk, B. Parno, H. J. Wang, and C. Cowan. User-driven access control: Rethinking permission granting in modern operating systems. In Symposium on Security and Privacy, 2012. [20] R. Schlegel, K. Zhang, X. yong Zhou, M. Intwala, A. Kapadia, and X. Wang. Soundcomber: A stealthy and context-aware sound trojan for smartphones. In Network & Distributed System Security Symposium, 2011. [21] A. Seshadri, M. Luk, N. Qu, and A. Perrig. Secvisor: A tiny hypervisor to provide lifetime kernel code integrity for commodity oses. In Symposium on Operating Systems Principles (SOSP), 2007. [22] A. Shabtai, R. Moskovitch, Y. Elovici, and C. Glezer. Detection of malicious code by applying machine learning classifiers on static features: A state-of-the-art survey. Information Security Technical Report, 2009. [23] B. Shrestha, N. Saxena, and J. Harrison. Wave-to-access: Protecting sensitive mobile device services via a hand waving gesture. In Cryptology and Network Security (CANS). 2013. [24] R. Templeman, Z. Rahman, D. J. Crandall, and A. Kapadia. Placeraider: Virtual theft in physical spaces with smartphones. In Network & Distributed System Security Symposium, 2013. [25] D. Venugopal. An efficient signature representation and matching method for mobile devices. In Wireless Internet (WICON), 2006. [26] M. Ward. Smartphone security put on test, 2010. Available online at http://www.bbc.com/news/technology-10912376.