

Secure Cloud Computing with Brokered Trusted Sensor Networks

Apu Kapadia, Steven Myers, XiaoFeng Wang and Geoffrey Fox
School of Informatics and Computing
Indiana University, Bloomington
{kapadia, samyers, xw7, gcf}@indiana.edu

ABSTRACT

We propose a model for large-scale smartphone based sensor networks, with sensor information processed by clouds and grids, with a mediation layer for processing, filtering and other mashups done via a brokering network. Final aggregate results are assumed to be sent to users through traditional cloud interfaces such as browsers. We conjecture that such a network configuration will have significant sensing applications, and perform some preliminary work in both defining the system, and considering threats to the system as a whole from different perspectives. We then discuss our current, initial approaches to solving three portions of the overall security architecture: i) Risk Analysis relating to the possession and environment of the smartphone sensors, ii) New malware threats and defenses installed on the sensor network proper, and iii) An analysis of covert channels being used to circumvent encryption in the user/cloud interface.

KEYWORDS: Sensor Network, Brokered Network, Security, Wireless.

1. INTRODUCTION

We consider systems in which there are large groupings of sensors reporting exorbitant quantities of potentially sensitive data, and the need to perform large amounts of processing or computation on this data with multiple large grid and cloud computing installations. The processing may need to be done in real or near-real time. Further, we consider that there are adversaries that have a vested interest in either learning information from the system, modifying the results finally output from the system (be it through modification of the sensor input, filtering or processing of data), or denying access to the system. Therefore maintaining

data provenance, secrecy and trust is of paramount importance throughout the data life-cycle (i.e, from the point of data collection by the sensors, to its final consumption by an individual or process). All data-transformation and filtering, networking and sensor aspects of these systems are assumed to be susceptible to attack. Similarly the environment in which some parts of the system operate is assumed to be potentially under adversarial control. In our modeling we assume the actual cloud-computing facility to be secure. Our goal is to be able to provide reliable results computed from sensor data in a manner that enables one (be it the user or the system) to make educated decisions on the reliability of that data based on trust metrics, while simultaneously preventing the loss of data-secrecy or integrity. Further, maintenance of system integrity and security is considered a core requirement. Issues such as anonymity are beyond the scope of our current research. Herein we provide a formal description of the networking architecture we anticipate and the security threats. We delineate between threats and security holes for which conventional security technology suffices to solve the problem, those threats for which modifications to conventional technology are required, and those which are new and somewhat specific to the problem at hand. We next outline a largescale feasible research program to solve the many associated problems. We conclude by highlighting several of the aspects of this program for which we are actively engaged in producing solutions, and the architectures for our solutions.

1.1. Roadmap

In Section 2. we provide a high-level specification of the type of systems we are considering. This is followed, in Section 3., by a high-level threat model that depicts ways adversaries can manipulate such systems and their malleable environments. In Section 4., we provide more in depth discussions on three specific subsets of security problems from Section 3. for which we are currently developing solutions. In Section 5. we provide related work for these

problems. Section 6. finishes off with discussion and conclusions.

2. COMPUTATION, NETWORKING & SENSING MODEL

We consider a model in which there are potentially millions of deployed sensors. The sensors may be (but are not necessarily) organized by some principle into different hierarchical layers or partitions. These sensors may be continuously publishing their observations, or supply their observations on request. In either event the observations are relayed through a brokering and filtering network, where sensor data is eventually consumed by a cloud or grid-computing infrastructure; alternatively the data can be filtered or processed, and stored. Importantly, we do **not** consider traditional low-power sensors such as motes, RFIDs and smartdust, where a great preponderance of wireless sensor-network research has been done. Rather, we consider potentially high throughput sensors attached to a — in comparison — large amount of computational and networking power, e.g., in the cloud. Specifically, we consider smartphone-class devices with reliable cellular network connectivity (with hundreds of Kbps throughput as opposed to tens of Kbps available on motes) and frequent recharging (e.g., nightly) that supports more computationally intensive applications than motes. Yet, this model still leaves open a large number of security issues that must be solved.

In Fig. 2. we visualize the different components of the networked system. Android smartphones denote the sensors in the system, and are in the possession of individuals. The smartphones have some computational capacity, and transmit through WiFi or cellular services to a brokering network, running over traditional TCP/IP services. The brokering service can itself have computers performing filtering, processing and/or creating other mashups of sensor data.

2.1. The Sensor

Herein, we consider the sensors to be modern smartphones. These devices are diversely deployed in the field, contain a large number of sensors, and have moderate computational ability. Further, they are fully networked, and with modern 3G networks have reasonable bandwidth (e.g., 100–1000kbps). Additionally, most sensors have 802.11 WiFi radios, and may have sporadic or continuous WiFi connections in urban environments, with bandwidth of 1–50Mbps. These phones may be in the control of trusted (or semi-trusted) individuals, or be located in some potentially untrusted environment. Further, they have a reasonable processing capability on modern low-power proces-

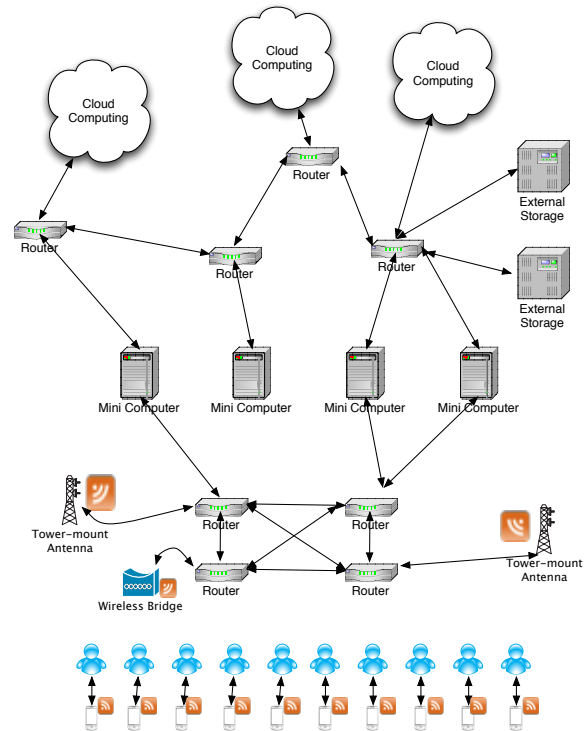


Figure 1. A Depiction of the Different Components of the Sensor and Cloud-Computing Network.

sors, such as an ARM architecture processor running at 500–800MHZ. It is assumed that the phones have standard sensors including, eGPS, 802.11x, Bluetooth v2 (Class 1, 2 or 3), temperature, orientation, acceleration, audio microphone, and camera (stills or video). In particular, our project focuses on the use of HTC G1 Android (v1.6) development phones, due to the ease of programming and their ability to multi-task (unlike the iPhone). Such platforms can perform a full host of cryptographic operations, but also have security issues relating to the fact that they are multi-purpose computing platforms. Thus OS security issues are larger, and it is difficult to construct a small OS, such as TinyOS [19] designed for motes, which can be more easily hardened to withstand attack. While the smartphones are capable of more standard cryptographic protocols, a large number of such sensors in a region that are broadcast could overwhelm communications channels, and battery life is still a concern — if not as pressing. Therefore, low bandwidth and energy usage requirements are still a concern. However, one can easily port low-energy and bandwidth secure networking stacks, such as those provided by TinySec [17] or MiniSec [21].

2.2. The Brokering Network

With potentially millions of smartphone sensors producing data at any given time, the need for a high performance networking infrastructure that is capable of self-filtering unimportant data feeds before they are transmitted for processing becomes apparent. Further, the need to funnel potentially very large amounts of bandwidth to a few collection points for processing is also evident. The communication between the sensors and the computing infrastructure is mediated by a brokering network that uses a publish/subscribe model. In such a model, each sensor can publish the data it is collecting on a continuous basis, along with appropriate meta-data that depict the content, provenance and trustworthiness of the data. Requests for specific information at the cloud or grid computing interface will drive the request for specific types and trustworthinesses of data from the sensors. Such requests will further invoke the subscription to different forms of data both real-time and stored. Typical forms of data cleaning and processing can, of course, be performed by dedicated servers who independently subscribe to sensor feeds, and then publish their own mashed data feeds for consumption by others. In such cases provenance and trustworthiness must be maintained. Ultimately, there will be many different parallel consumers of data, and thus the network must be as responsible as is possible to prevent duplication of effort, redundant routing, streaming and processing of data.

For this project, the Narada Brokering network¹ is being used. The network can provide basic secrecy and integrity requirements, but does not by default provide any information regarding provenance or trustworthiness. While other suitable brokering networks can be used (e.g., Solar [?]) we chose Narada because of local expertise and support available to our project.

2.3. Computing Model

We assume that the final consumers of data will be cloud or grid computations, as will many of the filtering and processing modules. While each cloud or grid may see its output as the final consumable, the desire to recycle computation means that the data may itself become simply another input to an alternate computation upstream. The study of securing cloud and grid computation are separate research fields in their own right, and so our model simply assumes that these computations do not leak information, break integrity of the data nor provide covert channels to the data. Computational power and storage is considered to be more or less limitless to within reasonable bounds.

¹See www.Naradabrokering.org

3. SECURITY, PRIVACY & TRUST ISSUES

The computing environments of a sensor grid are fraught with different kinds of threats, which endanger the security and privacy assurance the system can provide. Mitigation of these threats relies on establishing trust on individual system layers through proper security control. In this section, we survey the security and privacy risks on each layer of sensor-grid computing and the technical challenges for controlling them.

A sensor grid interacts with its operating environment through a set of sensors. Those sensors work either autonomously or collaboratively to gather data and dispatch them to the grid. Within the grid, a brokering system filters and routes the data to their subscribers, the clients of the sensor grid. We now describe the security and privacy issues on each layer of such an operation. This includes the environment the sensors are working in; the sensors; the grid; the clients; and the communications between the sensor and grid, and the grid and clients.

The Environment. An adversary could compromise the sensors' working environments to contaminate the data they collect. For example, one can add ice around individual sensors to manipulate the temperatures they measure; alternatively, one could imagine that GPS signals were being spoofed in an area. Detection of such a compromise can be hard, when the adversary has full control of the environment. A possible approach is to check the consistency of the data collected from multiple sensors and identify anomalous environmental changes as indicated by the data.

Sensors. Sensors can be tampered with by the adversary who can steal or modify the data they collect. Mitigation of this threat needs the techniques that detect improper operations on the sensors and protect its sensitive data. Since we assume sensors are smartphones, they also are susceptible to a large number of security concerns of traditional PCs, which includes viruses and malware.

Cloud or Grid. Information flows within the grid can be intercepted and eavesdropped on by malicious code that is injected into the system through its vulnerabilities. Authentication and information-flow control need to be built into the brokering system to defend against such a threat.

Client. The adversary can also manage to evade the security and privacy protection of the system through exploiting the weaknesses of the clients' browsers. The current design of browsers is well known to be insufficient for fending off attacks such as cross-site scripting (XSS) and

cross-site request forgery (XSRF). Such weaknesses can be used by the adversary to acquire an end user's privileges to wreak havoc on the grid. Defense against the threat relies on design and enforcement of a new security policy model that improves on the limitations of the same origin policy adopted in all of the mainstream browsers.

Communication Channels. The communications between the sensors and the brokering network, the brokering network and the cloud or grid, and the cloud or grid and the client, are subject to both passive (e.g., eavesdropping) and active (e.g., man-in-the-middle) attacks. Countering this threat depends on proper cryptographic protocols that achieve both data secrecy and integrity. In each case, different engineering requirements based on differing scarce resources require different solutions. In the case of the wireless connection between the sensor network and the brokering network, bandwidth and power-usage are key requirements. Once on the brokering network, data provenance becomes a key challenge. Traditional cryptographic protocols would seemingly suffice from the cloud to the user. However, a tricky issue here is the information leaks through side channels. For example, packet sizes and sequences. Our preliminary research shows that such information reveals the state of web applications, which can be further utilized to infer sensitive data within the application. Understanding and mitigating the problem needs further investigation.

4. PROBLEMS TO BE ADDRESSED

While there are a large number of potential security issues to be addressed, as partially scoped and enumerated in the previous section, the investigators are working on the following specific problems.

4.1. Detection of anomalous use of sensors

A key issue involved in trusting data from the sensors in the described network is to ensure that the sensors themselves can be trusted. That is, either they are in the possession of individuals who are trustworthy, or they have not been tampered with in their environment if not possessed by an individual.

In our model if the sensor is in the possession of a trusted individual, it is more likely that its sensors are reporting an honest or legitimate environment, and not one that has been manipulated with the goal of producing faulty results that get incorporated in to final computation. Smartphones, however, can be easily stolen, misplaced or temporarily intercepted and reprogrammed by adversaries. If stolen or misplaced, the environment that the sensors report may be

altered, and thus the data collected may be untrustworthy. The use of traditional authentication technologies to ensure a legitimate user is in control of the smartphone sensor is not practical, as said users cannot be queried to authenticate every time the sensor-net needs to report readings.

We propose a system in which a phone attempts to determine if it is or is not in the possession of a legitimate user. In cases where the phone determines it is in questionable hands it deauthenticates itself. Deauthentication either removes it from the sensor network, or forces its sensor readings to be tagged as untrustworthy, with risk measurements being included in provenance data to ensure that the risk of improper readings is communicated down stream and taken into account on further processing. In order for the phone to determine whether it is under legitimate possession, we are developing a risk assessment system based on the inputs from the sensors of the phone itself. Thus the sensors are used directly to determine if the sensors' readings should be trusted. We are implementing a prototype of this system on the HTC/Google G1 Android (v1.6) Phone.

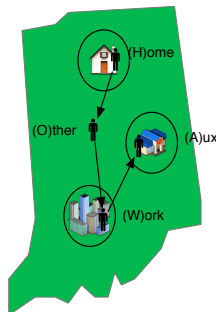
We are taking different approaches with different sensors on the phones. Note we are using these sensors to determine risk of improper possession independent of which sensors are of interest to the sensor network. Further, we make two broad classifications of the use of sensor input for risk determination. First, *environmental sensors* attempt to measure properties of the environment around the phone, or of the user. Second, *social-networking sensors* measure "friendly" or "unfriendly" people that surround the phone.

4.1.1. Environmental Sensors

Positioning Information. Android smartphones can determine their position using a combination of several different information sources, which includes cellular transmissions (in particular, tower location), GPS positioning and WiFi positioning. The combination of all of these pieces of information is often called eGPS, and frequently provides position far more accurately than any of the technologies alone. Our high-level goal is for the phone to learn certain geographic locations and routines that correspond to either a safe or dangerous state.

We extend the work of Farrahi and Gatica-Perez [14]. We are using a third-order Hidden Markov Model (HMM) to determine the risk of misuse of a phone based on current positional information. Farrahi and Gatica-Perez considered the problem of determining location for contextual application purposes, but without specific interest in authentication and security mechanisms. A day is divided into blocks of 30 minutes. In any given period the phone is con-

sidered to be in one of four specified places (e.g., Home, Work, Aux 1, No Location Reading) or in a generic unlabeled place (Other). Thus the location of an individual through a time period is being converted into a string, as is depicted in Fig. 2. Currently, we are considering a supervised learning case where a user specifically defines these five locations, with the goal of using clustering algorithms to eventually learn popular locations. Traces of individuals' positions are then collected, and the HMM iterative Viterbi training and Forward algorithm are used for training on this past annotated data sequences and predicting risk. Based on a trained HMM, and a recent history of the phones' positions, the forward algorithm is used to determine the likelihood of the recent history, and this estimate is used to determine the risk associated with the phone's current position. Of clear importance is the efficiency with which both training and evaluation can be performed. Due to the need to only occasionally perform training (say daily or weekly to update the movement model with the most recent trends), its efficiency is of lesser importance than that of real-time risk evaluation which needs to be performed on demand in real-time in order to prevent users from becoming frustrated with risk-calculation delays.



Location recorded every 30-Min. for 24 Hrs. producing the string

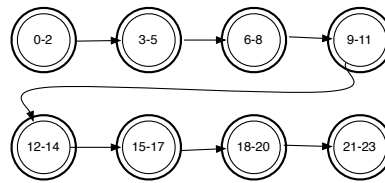
HOWAAA.....

String is parses starting on each letter into triplets for 3rd order HMM

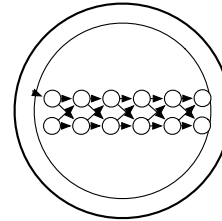
H	O	W	W	A				
O	W	A	A	A				
W	A	A	A	A				

Figure 2. A Depiction of How Positional Data Through the Day is Converted in to a String Over a Small Alphabet.

As previously mentioned, risk evaluation is based on the use of the forward algorithm. The forward algorithm runs in $\mathcal{O}(n^2 \cdot t)$ where n is the number of states and t is the number of time-blocks being analyzed; given an HMM M the forward algorithm returns the probability that a given sequence of positions x_1, \dots, x_t is output by an HMM, given that it terminates in state σ_t . More formally, $\Pr[M \rightarrow x_1, \dots, x_t | \sigma_t]$, for a given x_1, \dots, x_t , and σ_t . However, for



A hierarchical HMM model is used to learn users schedules. At the outer layer we in essence have a node for each 3 hour block of time in the day.



Each node contains within it a 3rd order multi-state HMM to learn the schedule over the corresponding hours.

Figure 3. A Depiction of the Constructed HMM for Predicting Position.

risk analysis we have no preference for any specific terminal state, and so we are interested in $\Pr[M \rightarrow x_1, \dots, x_t]$. A simple modification that sums the probabilities over all final states runs in $\mathcal{O}(n^3 \cdot t)$, and returns the value of interest. Given the running time is cubic in the number of states and we need near real-time evaluations of the algorithm, we need to minimize the state space. To minimize the state space we actually construct 8 individual HMMs to learn patterns of behavior during different 3-hour periods of the day, and link them together through a simple state-machine.² The model is depicted in Fig. 3.

We justify this construction as a reasonable model because the risk of one's current geographic position is a function of both one's current position and recent historical position relative to the current time, as opposed to one's longterm schedule. We are currently in the process of experimentally determining the correct recent history window that will deliver the best ability to detect abnormal behavior.

Temperature Temperature of the phone can be used to determine information relating to whether the phone is currently in someone's physical possession. If the phone reads approximately body temperature ($37^\circ C$) then it is reasonable to assume that is in a person's possession.³ Similarly, if the phone is at approximately room temperature or the outdoor ambient temperature, then the phone is likely either not directly on the person and is likely to have either

²This construction could be viewed as a Hierarchical HMM in which the transition distribution in the high-level HMM are all Kronecker δ -functions.

³There may need to be some invalidation of this metric at times when the ambient temperature is the same as body temperature.

been put down or remain in a bag.

While we believe there is strong potential to help use the phone's current temperature to monitor risks, our initial test of the Android phone is that the delay in converging to new temperatures by the phone's sensor makes this data unusable for our intended applications. We found that when moving the phone in a pocket at body temperature and moving it onto a desk, it took on the order of tens of minutes to converge to anywhere near the ambient room temperature. Further, in the same scenario it took several minutes to decisively report non-body temperature readings.

Acceleration Acceleration measurements can be used in several manners to help determine risk. Techniques have been developed to measure a person's gait using the accelerometer in phones, assuming they are placed in an individual's pocket, or otherwise carried on the person [30, 15, 1]. While we do not intend to implement such a scheme ourselves, we are looking at the possibility of including the results of these works to deploy such a technique in our larger sensor scheme. Further, we plan to use techniques that include simpler measurements but are based on other contexts. For example, if a user does *explicitly* authenticate to the device, then at this point in time we know that the device is trusted. If the device stays in motion for the next several minutes, then one can assume that the correct user is still in possession of the device. In contrast if the phone becomes stationary for a prolonged period of time, the phone probably has been put down, and now alternative risk measurements must be used.

4.1.2. Social Networking Sensor Risk Measurement

One key aspect of our system is to use a form of social networking for authentication and risk measurement. Imagine a scenario where a phone finds itself in a previously unvisited location, and other sensors are providing questionable risk data. However, imagine that the device can find the presence of a number of other phones that it frequently observes when in known low-risk states. The presence of these phones should indicate that the risk that an individual does not have proper possession of the phone is low: the phones of colleagues, friends and family members are near, so either the entire group is at risk (unlikely or the phone is simply in a new environment). Our system will employ a combination of white and black listing of other phones, which will alter the risk assessments made by the system. Additionally, we will learn "friendly" phones by determining which other phones are frequently in the presence of the user in non-risky situations. This assessment will be done by considering both Bluetooth and 802.11 wireless networks.

Bluetooth. General Bluetooth frames are much more difficult to detect than corresponding 802.11x frames *with the standard radio hardware built in to phones*.⁴ There are two options to bypass this problem. The first is that the phones broadcast themselves in so called "Bluetooth discovery mode", this will make the phone visible to all, but can result in higher battery usage. The second is to pair specifically with those phones that are whitelisted to be considered friendly; pairing requires a one-time user intervention. In this case, the phones could attempt to pair when they are in close contact.

More problematically, our current implementation platform (Android v1.6) does not provide an API to interface with the Bluetooth infrastructure. Thus Bluetooth can only be accessed by the user, and not a risk-analysis program. Android (v2.0) does provide the implementation of such API, but there is currently no firmware upgrade for our reference platform (HTC G1 development).

WiFi (802.11x). Much of the widely deployed smartphones allow their WiFi radios to operate in *promiscuous mode*, which permits the radio to listen to and communicate the existence of frames that it can receive, even if the radio was not the target for the frame in question. This mode allows 802.11x radios to detect the presence of nearby devices. The only requirement to instantiate our social-networking risk measurement is to ensure that all the participating phones are broadcasting their position by sending beacons on regular intervals. It is yet to be determined if the development platform supports such modes of operation.

4.1.3. Combining Risk Measurements.

A more sensitive risk measurement can be constructed if one does not require each sensor to independently generate a risk metric in our risk model. However, in order to make our scheme flexible for different uses, and in devices with different subsets of sensors, we consider an architecture that treats the sensor measurements independently, and then produces a global risk measurement. Note that this separation does not prevent the global risk measurement from learning co-dependencies between risk profiles of different sensors, and making use of such dependencies. There is a fair amount of research on methods for aggregating risk measurements in a number of different scenarios (e.g., Financial, Credit, Insurance, Intrusion Detection). Currently we are determining which, if any, of the current models provides a similar or appropriate model on which to base an aggregation of our sensor work. In the mean time,

⁴Relatively inexpensive hardware is available to capture general Bluetooth packets, but it is not standard on known phones.

we use an expected value of the different risk metrics that is weighted with high-degrees to the positional and social networking schemes.

4.2. “Sensory Malware” threats and defenses

To fully understand the threat space of malware on smartphones, we are exploring various attack scenarios. While traditional malware defenses focus on protecting resources on the computer (or as we would expect, on the smartphone), we are specifically interested in the new class of attacks where *sensory malware* uses onboard sensors to steal information from the user’s physical environment [5]. For example, the user carries around a video and audio sensor (microphone) at all times, and thus immense amounts of information such as sensitive conversations, spoken passphrases or biometrics, keyboard acoustic emanations when placed next to a keyboard, and broader surveillance becomes possible. Video “sensors” can gather visual information about a user’s private environment such as pictures of colleagues [38], which may be sensitive with military and intelligence-gathering agencies. Accelerometers and GPS sensor information can be used to infer location and activity patterns of users such as soldiers, thus compromising military secrecy.

While generic architectures [10, 23] have been proposed to control access to the network, for example, after software has accessed certain sensor information, various vectors exist for leaking garnered information. Overt channels between components on the smartphone (Android provides very little security against communicating applications, for example), or covert channels between related malware applications (through a storage channel, for example) are currently viable vectors for leaking sensitive data to adversaries. It is even possible to leverage other “blessed” applications on the phone to act as a carrier for such information (by invoking a web-browser with an encoded URL, for example). Thus we are interested in building a unified architecture for controlling access to sensor data, and limiting what information can be gleaned from the user’s environment unless he or she is making use of legitimate applications. We are currently building a software prototype of one instance of sensory malware to demonstrate the reality of the threat, and to better understand defensive techniques to limit such malware.

We aim to study types of sensory malware that are stealthy and thus use few resources on the mobile device. For example, speech-based malware may use several heuristics to target analysis at only specific portions of the audio sample. Such targeted analysis can drastically reduce the amount of resources needed to analyze audio samples, thus decreasing

the observability of such malware. To conserve power, such malware can also target its offline processing to when the mobile device is connected to a power source for charging. Under such circumstances the malware uses few precious resources and does not detract from the user’s experience. Speech malware of this type may even operate using more general “profiles” that tune the malware to recognize several different situations, or contexts, such as a recognized phone number that is dialed. Based on the context, the speech malware can, for example, detect a credit card customer service line and target analysis to credit card number extraction. Calls to financial institutions such as banks often require portions of the user’s social security number, which could be extracted similarly. Such profiles can make use of other clues such as audio or video triggers to better target surveillance and transmit specific information.

To counter such threats, therefore, we need a framework that is better equipped to deal with sensory malware threats. Research is needed to understand the threat space of sensory malware, so that effective defenses can be deployed. As mentioned earlier, existing solutions are unable to deal with situations in which malware communicates through covert channels, and thus such work must also take into account anomalous resource usage to detect such covert channels. Being low-powered devices makes the job of defensive software much more challenging, and thus lightweight detection techniques are necessary. It is even possible that the mobile platform can leverage computation in the cloud for “outsourced intrusion detection,” which might strike a tradeoff between the time to detection and power consumption.

4.3. Side-channel detection and mitigation

It is well known that the contents of encrypted traffic can be disclosed by its attributes observable to a eavesdropper, for example, packet sizes, sequences, inter-packet timings. Such attributes, often referred to as side-channel information, often pose a grave threat to the confidentiality of the communication under the protection of cryptographic protocols. Side-channel leaks have been extensively studied for decades, in the context of secure shell (SSH) [27], video-streaming [26], voice-over-IP (VoIP) [37], web browsing and others. As an example, a line of research conducted by various research groups studied anonymity issues in encrypted web traffic. It has been shown that because each web page has a distinct size, and usually loads some resource objects (e.g., images) of different sizes, the attacker can fingerprint the page so that even when a user visits it through HTTPS, the page can still be re-identified [9, 29]. This vulnerability is known to be a serious concern for anonymity channels such as Tor [31],

which are expected to hide users' page-visits from eavesdroppers.

A sensor grid system can also be highly susceptible to the threat of side-channel leaks. As described before, such a system collects data through distributed sensors, processes it within a cloud, and delivers the data and related services to end clients. This highly distributed computing paradigm is fraught with the hazards of information leaks, when confidential data are transmitted between the sensors and the cloud, and between the cloud and the clients, despite the protection of the state-of-the-art cryptographic techniques. Such privacy risks are described as follows:

Wireless Sensor Communication. The wireless channel connecting the sensors to the cloud is extremely vulnerable to the eavesdropping attack. The sensitive data delivered through this channel can be easily intercepted and analyzed by the adversary. Though encryption can prevent a direct disclosure of the data, it does not cover the side-channel information, which, under some circumstances, can be used to infer the content of the sensitive data. As an example, collaborating with Microsoft Research (MSR), we recently discovered that even for the organization deploying up-to-date WPA/WPA2 Wi-Fi encryptions, it cannot prevent an unauthorized party from collecting the query words its employees enter into Google/Yahoo/Bing Search. This is because the suggestion-list features of these search engines makes the sizes of the packets generated in response to different query letters distinct. As a result, the adversary who observes these packets, despite not gaining access to their contents, can map their sizes to the different letters one types into the search engines.

Cloud-consumer Communication. The encrypted data exchanged between the cloud and its customers are equally subject to the side-channel threat. Cloud computing is built upon the infrastructure of *software as a service* (SaaS), through which *web applications* are delivered as services to web clients. Unlike its desktop counterpart, a web application is split into browser-side and server-side components. As a result, a subset of its internal information flows (i.e., data flows and control flows) are inevitably exposed on the network, which reveal application states and state transitions. Our collaborative research with MSR reveals that the side-channel weakness of SaaS is fundamental, which can be used to infer a large amount of information from many high-profile, extremely popular web applications. The sensor grid system also faces the same threat: it offers services and data to its customers through web applications, whose side-channel information could lead to the disclosure of the data, even when the communication has been protected by the cryptographic protocols like HTTPS.

The seriousness of the side-channel threat varies from case to case, depending on the features of the data and the way in which they are transmitted. An important research, therefore, becomes how to design a systematic way to detect the side-channel vulnerabilities within sensor/cloud interactions and the web applications that serve the sensor grid's customers. A possible solution is to use *information-flow analysis* [28], when the source code of related software is available. The software developer can first label taint sources within a program, e.g., variables that contain sensitive user data, and then run a detection tool to analyze its source code and track the propagation of taint data through both data flows and control flows. Whenever taint data are found to be transmitted across the network between the application's client and server components, an information-leak evaluation is performed to understand whether side-channel information, such as packet sizes, sequences and timings, can be linked back to the content of the data. When the source code is unavailable, we can use the techniques like fuzz testing to evaluate sensor-cloud interactions and cloud-client interactions on different data sets, to identify the correlation between the attributes of encrypted traffic and the content of the data.

Control of side-channel leaks can also be highly nontrivial, particularly when web applications are involved. Our collaborative research with MSR reveals that conventional defenses like packet padding and adding noise can be less effective and more costly than expected, without considering the specific properties of individual applications. This problem comes from the difficulty in hiding the side-channel information related to state transitions specific to each application, and the limited information an application has about the attributes of the web traffic it generates, due to the extension or compression made by the web server. This vulnerability calls for a change in the current way of developing web applications to include the collaborations among multiple related parties: as an example, we could let the software developer specify the policies for padding packets at different program states, and the web-server vendor enforce the policies within the web server that actually generates the packets.

5. RELATED WORK

Kapadia et al. [16] list several security challenges for similar smartphone based sensing environments. While their work focuses mainly on an opportunistic sensing model where sensors are *tasked* for readings sent back as *reports* to other users or applications in urban sensing environments, we focus on environments where sensors push massive amounts of data to a compute cloud. We now list related work for the three specific problems discussed in Section 3..

5.1. Mobile phone security and privacy

There has been some work in using sensors to establish context for different purposes on smartphones. The work of Peddemors et al. [24] uses past networking and sensor events to predict future network events. They give examples of predicting network availability. The ability to predict events is distinct from deviating from normal or prescribed behavior. Nonetheless they use the prediction of being at home or work, and for durations. Therefore, the system should be considered. Of particular problem is the complexity of computing predicted events, which would be too slow in our scenario.

The work of Tanviruzzaman et al. [30] is most similar to that discussed here. In their work, they suggest the use of a hierarchy of sensor information to establish authentication, and show some work on using accelerometer data on an iPhone to produce a biometric that can be used to authenticate to the phone.

Other work by Jong-Kwon and Hou [18] has predicted user behavior and movements from the perspective of a large WiFi network, for the purposes of assigning scarce resources appropriately. However, we do not rely on one overarching network for our positioning system. Yet, the possibility exists that such work could be used to have the network aid in performing risk analysis.

The field of smartphone security and the security of cellphone infrastructure is now being widely researched. Traynor [32] gives a short overview of infrastructure possibilities and problems. Traynor et al. [34] consider the potential effect of a malnet of smartphones on the cellular network's infrastructure. Enck et al. [13] discuss exploits in the SMS-network infrastructure, and Traynor et al. [33] discuss mitigation strategies for such exploits.

Relating to mobile phone security, there has been recent interest in maintaining their security. The potential to attack these devices, and that they would suffer similar security fates to personal computers, such as viruses and malware, has been long understood [8]. Specific approaches to considering defense against such software on smartphones has been considered by Cheng et al. [7]. The specific strengths and weaknesses of the Android security model are explored by Ongtang et al. [23]. The ability to securely determine if software downloads are trusted on such devices is explored by Enck et al. [11]. Enck et al. [12] give an introduction to understanding the Android security model specific to the smartphones we are using for implementation.

5.2. Sensory malware threats and defenses

As mentioned earlier, researchers are already investigating attacks and defenses related to sensory malware [5]. Xu et al. [38] provide a proof-of-concept implementation of video-capture malware. Their malware captures video and transmits this video after suitable compression to lessen the burden on the network. These malware do not appear to be stealthy enough because of the large amounts of video data transferred on the network. We thus seek to develop and evaluate solutions where malware is even more stealthy, by limiting the network communication. In fact, we would like to study situations where network access is limited completely using techniques such as Kirin, a lightweight security certification mechanism for applications on Android. Even in cases where a system such as Saints [23] is used to control the interaction between applications, we would like to study the use of covert channels to circumvent such mechanisms.

Detection techniques such as behavioral detection of malware by monitoring system calls [3], and power consumption [20] already attempt to detect malware on mobile platforms. We aim to study the limits of such detection techniques since resources are limited, and how malware can circumvent detection because of the inherent limitations on the detection techniques.

5.3. Side-channel information leaks

Side-channel leaks have been known for decades: a documented attack has been dated back to 1943 [22]. The threat has been extensively studied in different contexts: information is found to be exposed through electromagnetic signals (e.g., keystroke emanation [35]), shared memory/registers/files between processes (e.g., the recent discovery of the side-channel weakness in Linux process file systems [39]), CPU usage metrics, etc. Recently, such information leaks are found to threaten cloud computing platforms like Amazon EC2 [25].

Encrypted communications are often subject to the side-channel attacks, which leverage such information as packet timings and sizes to infer the contents of encrypted data. Prominent examples include Brumley et al.'s attack on the RSA secret keys used in OpenSSL [4], Song et al.'s work on keystroke inference from SSH [27], Wright et al. and others' analysis of phrases and sentences from the variable-bit-rate encoding in VoIP [37], and Saponas et al.'s detection of movie titles in an encrypted video-streaming system (Slingbox Pro) [26]. Encrypted web communication has also been found to be vulnerable to the side-channel attack. Prior research shows that a network eavesdropper can often

fingerprint web pages using their side-channel characteristics to identify the pages the victim visits. This idea first appeared in the personal communication among Wagner, Schneier and Yee in 1996 [36], and was later demonstrated in a course project report in 1998 by Cheng et al. [6]. Sun et al. [29] and Danezis [9] both indicated the impacts of the attack on anonymity channels like Tor, MixMaster and WebMixes. It was also discussed by Bissias et al. [2], who studied WPA and IPsec, instead of SSL/TLS in other research.

6. SUMMARY

We have outlined a high-level architecture that should both be realizable, and provide for the ability to perform on-demand analysis and processing of data from a large number of heterogeneous and globally placed sensors. The network is structured so that it is feasible to consider real or near-real time processing and interpretation of the data with appropriate resources. However, challenges remain in determining how to assure privacy, integrity and provenance of the data from its collection, through its life-cycle of processing to final consumption. The authors' belief is that the largest research questions based on our model lie at the tail ends of the data life-cycle; namely, there are open research questions at data-collection by smartphone sensors and in the final delivery of a processed data-consumable. Specific directions aimed at solving these problems have been discussed, along with initial development of solutions. We summarize this in Table 1

Table 1. Summary of the Three Threats, Associated Dangers and Mitigation Strategies We Actively Address.

Threat	Danger	Mitigation
Sensor Abduction	Malicious Sensor Data	Detection of non-regular usage
Side-channel information leakage	Communication encryption is circumvented by analysis of packet sizes& spacing	Flow-analysis and padding
Sensor malware	Sensor data theft	Sensor access control models

REFERENCES

- [1] *Identifying users of portable devices from gait pattern with accelerometers*, volume 2, 2005.
- [2] G.D. Bissias, M. Liberatore, D. Jensen, and B.N. Levine. "Privacy vulnerabilities in encrypted http streams," In proceedings of Privacy Enhancing Technologies Workshop (PET 2005), pages 1–11, 2005.
- [3] A. Bose, X. Hu, K.G. Shin, and T. Park. "Behavioral detection of malware on mobile handsets," In *MobiSys '08: Proceeding of the 6th international conference on Mobile systems, applications, and services*, pages 225–238, New York, NY, USA, 2008. ACM.
- [4] D. Brumley and D. Boneh. "Remote timing attacks are practical," In proceedings of the 12th USENIX Security Symposium, pages 1–14, 2003.
- [5] L. Cai, S. Machiraju, and H. Chen. "Defending against sensor-sniffing attacks on mobile phones," In *MobiHeld '09: proceedings of the 1st ACM workshop on Networking, systems, and applications for mobile handhelds*, pages 31–36, New York, NY, USA, 2009. ACM.
- [6] H. Cheng and R. Avnur. "Traffic analysis of SSL encrypted web browsing," <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.3.1201\&rep=rep1\&type=url\&i=0>, 1998.
- [7] J. Cheng, S.H.Y. Wong, H. Yang, and S. Lu. "Smartsiren: virus detection and alert for smartphones," In *MobiSys '07: proceedings of the 5th international conference on Mobile systems, applications and services*, pages 258–271, New York, NY, USA, 2007. ACM.
- [8] D. Dagon, T. Martin, and T. Starner. "Mobile phones as computing devices: The viruses are coming!" *IEEE Pervasive Computing*, 3(4):11–15, 2004.
- [9] G. Danezis. "Traffic analysis of the http protocol over TLS," <http://homes.esat.kuleuven.be/~gdanezis/TLSanon.pdf>, as of Dec 2009.
- [10] W. Enck, M. Ongtang, and P. McDaniel. "On lightweight mobile phone application certification," In *CCS '09: proceedings of the 16th ACM conference on Computer and communications security*, pages 235–245, New York, NY, USA, 2009. ACM.
- [11] W. Enck, M. Ongtang, and P. McDaniel. "On lightweight mobile phone application certification," In *CCS '09: proceedings of the 16th ACM conference on Computer and communications security*, pages 235–245, New York, NY, USA, 2009. ACM.
- [12] W. Enck, M. Ongtang, and P.D. McDaniel. "Understanding android security," *IEEE Security & Privacy*, 7(1):50–57, 2009.
- [13] W. Enck, P. Traynor, P. McDaniel, and T. La Porta. "Exploiting open functionality in sms-capable cellular networks," In *CCS '05: proceedings of the 12th ACM conference on Computer and communications security*, pages 393–404, New York, NY, USA, 2005. ACM.
- [14] K. Farrahi and D.G. Perez. "Learning and predicting multimodal daily life patterns from cell phones," In J.L. Crowley, Y. Ivanov, C.R. Wren, D. Gatica-Perez, M. Johnston, and R. Stiefelhagen, editors, *ICMI*, pages 277–280. ACM, 2009.

- [15] T. Iso and K. Yamazaki. "Gait analyzer based on a cell phone with a single three-axis accelerometer." In *MobileHCI '06: proceedings of the 8th conference on Human-computer interaction with mobile devices and services*, pages 141–144, New York, NY, USA, 2006. ACM.
- [16] A. Kapadia, D. Kotz, and N. Triandopoulos. "Opportunistic Sensing: Security Challenges for the New Paradigm." In *The First International Conference on Communication Systems and Networks (COMSNETS)*, January 2009.
- [17] C. Karlof, N. Sastry, and D. Wagner. "Tinysec: a link layer security architecture for wireless sensor networks." In *Sensys '04: proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 162–175, New York, NY, USA, 2004. ACM.
- [18] J.K. Lee and J.C. Hou. "Modeling steady-state and transient behaviors of user mobility: formulation, analysis, and application." In *MobiHoc '06: proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, pages 85–96, New York, NY, USA, 2006. ACM.
- [19] P. Levis, S. Madden, J. Polastre, R. Szewczyk, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler. "TinyOS: An operating system for sensor networks." In *Ambient Intelligence*. Springer Verlag, 2004.
- [20] L. Liu, G. Yan, X. Zhang, and S. Chen. "Virusmeter: Preventing your cellphone from spies." In E. Kirda, S. Jha, and D. Balzarotti, editors, *RAID*, volume 5758 of *Lecture Notes in Computer Science*, pages 244–264. Springer, 2009.
- [21] M. Luk, G. Mezzour, A. Perrig, and V. Gligor. "Minisec: a secure sensor network communication architecture." In *IPSN '07: proceedings of the 6th international conference on Information processing in sensor networks*, pages 479–488, New York, NY, USA, 2007. ACM.
- [22] Wired News. "Declassified NSA document reveals the secret history of tempest," <http://www.wired.com/threatlevel/2008/04/nsa-releases-se>.
- [23] M. Ongtang, S. E. McLaughlin, W. Enck, and P. D. McDaniel. "Semantically rich application-centric security in Android." In *ACSAC*, pages 340–349. IEEE Computer Society, 2009.
- [24] A. Peddemors, H. Eertink, and I. Niemegeers. "Predicting mobility events on personal devices", *Pervasive and Mobile Computing, Special issue on Human Behaviour in Ubiquitous Environments*, To Appear.
- [25] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage. "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds." In *CCS '09: proceedings of the 16th ACM conference on Computer and communications security*, pages 199–212, New York, NY, USA, 2009. ACM.
- [26] T.S. Saponas, J. Lester, C. Hartung, S. Agarwal, and T. Kohno. "Devices that tell on you: privacy trends in consumer ubiquitous computing." In *SS'07: proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, pages 1–16, Berkeley, CA, USA, 2007. USENIX Association.
- [27] D.X. Song, D. Wagner, and X. Tian. "Timing analysis of keystrokes and timing attacks on SSH." In *SSYM'01: proceedings of the 10th conference on USENIX Security Symposium*, pages 25–25, Berkeley, CA, USA, 2001. USENIX Association.
- [28] G.E. Suh, J.W. Lee, D. Zhang, and S. Devadas. "Secure program execution via dynamic information flow tracking." In *ASPLOS-XI: proceedings of the 11th international conference on Architectural support for programming languages and operating systems*, pages 85–96, 2004.
- [29] Q. Sun, D.R. Simon, Y.M. Wang, W. Russell, V.N. Padmanabhan, and L. Qiu. "Statistical identification of encrypted web browsing traffic." In *SP '02: proceedings of the 2002 IEEE Symposium on Security and Privacy*, page 19, Washington, DC, USA, 2002. IEEE Computer Society.
- [30] M. Tamviruzzaman, S.I. Ahamed, C.S. Hasan, and C. O'Brien. "ePet: when cellular phone learns to recognize its owner." In *SafeConfig '09: proceedings of the 2nd ACM workshop on Assurable and usable security configuration*, pages 13–18, New York, NY, USA, 2009. ACM.
- [31] The Tor Project. *Tor: anonymity online*. <http://www.torproject.org/>, 2009.
- [32] P. Traynor. "Securing cellular infrastructure: Challenges and opportunities." *IEEE Security & Privacy*, 7(4):77–79, 2009.
- [33] P. Traynor, W. Enck, P. McDaniel, and T.L. Porta. "Mitigating attacks on open functionality in sms-capable cellular networks." In *MobiCom '06: proceedings of the 12th annual international conference on Mobile computing and networking*, pages 182–193, New York, NY, USA, 2006. ACM.
- [34] P. Traynor, M. Lin, M. Ongtang, V. Rao, T. Jaeger, P.D. McDaniel, and T.F. La Porta. "On cellular botnets: measuring the impact of malicious devices on a cellular network core." In E. Al-Shaer, S. Jha, and A.D. Keromytis, editors, *ACM Conference on Computer and Communications Security*, pages 223–234. ACM, 2009.
- [35] M. Vuagnoux and S. Pasini. "Compromising electromagnetic emanations of wired and wireless keyboards." In *proceedings of the 18th USENIX Security Symposium*, pages 1–16, Montreal, Canada, 2009. USENIX Association.
- [36] D. Wagner and B. Schneier. "Analysis of the SSL 3.0 protocol." In *WOEC'96: proceedings of the Second USENIX Workshop on Electronic Commerce*, pages 4–4, Berkeley, CA, USA, 1996. USENIX Association.
- [37] C.V. Wright, L. Ballard, S.E. Coull, F. Monrose, and G.M. Masson. "Spot me if you can: Uncovering spoken phrases in encrypted voip conversations." In *SP '08: proceedings of the 2008 IEEE Symposium on Security and Privacy*, pages 35–49, Washington, DC, USA, 2008. IEEE Computer Society.
- [38] N. Xu, F. Zhang, Y. Luo, W. Jia, D. Xuan, and J. Teng. "Stealthy video capturer: a new video-based spyware in 3g smartphones." In *WiSec '09: proceedings of the second ACM conference on Wireless network security*, pages 69–78, New York, NY, USA, 2009. ACM.
- [39] K. Zhang and X. Wang. "Peeping Tom in the Neighborhood: Keystroke Eavesdropping on Multi-user Systems." In *USENIX Security '09: proceedings of the 18th USENIX Security Symposium*, Montreal, Canada, 2008. USENIX Association.