Research Statement

Muhammad Naveed

My friend and I went to a convenience store in Denver for soda and the salesman told us the store's computer was down, so we could not buy anything. Computers are crucial to our lives; they are used by billions of individuals and millions of organizations. Due to such universal adoption, they are being attacked for fun, profit, and chaos. Computers are far from being secure against such attacks. The Ashley Madison breach led to suicides and shattered families. The Apple iCloud breach led to the public release of nude photos of several celebrities. Code Spaces, a version control company, went out of business after hackers deleted its data from Amazon cloud. Therefore, I work to make the world a better place by building systems that prevent such attacks.

I analyze and build secure systems. I work in two complementary areas: *applied cryptography* and *systems security*. In applied cryptography, I analyze cryptographic schemes and build practical-yet-provably-secure systems. In systems security, I explore the fundamental security flaws in popular systems, such as Android, and build defense systems requiring no or minimal changes to the underlying systems. My work in these areas complement and reinforce each other. I developed a cryptographic system to prevent attacks on cloud messaging services [10] and used system techniques to develop cryptography for data outsourcing [1, 4]. My work is helping bridge the gap between cryptography and systems. During my PhD work, I fostered synergies between cryptographers, system experts, physicians, genomicists, and lawyers from 15 institutions worldwide. I am confident I can foster such synergies in the future as well.

My work has had significant impact on Android security and has helped companies such as Google, Samsung, Facebook, and Amazon secure their products and services, improving security for millions of Android users. The Google Android Security team acknowledged me on their website for my contributions that are incorporated in Android and positively impacted Android security.

Applied Cryptography

My Approach. I design practical cryptographic solutions for real applications, such as cloud storage, data breach prevention, electronic medical records, and genomic privacy, using formal modern cryptography: precise definitions, well-stated assumptions, and rigorous proofs. I approach cryptography from an application viewpoint: *application→requirements→solution→system*. I study applications in detail and collaborate with application domain experts to formulate requirements. I investigate whether a practical solution is possible; if not, I explore the reasons behind it to relax the requirements so as to find a useful solution for the application. Key elements of my research are novel communication, computation, and trust models that capture real applications faithfully and allow for practical cryptographic schemes. I build systems implementing my solutions and evaluate them in realistic settings. I also cryptanalyze cryptographic schemes and systems to understand what security they offer to real applications. I believe that such cryptanalysis is crucial for understanding and designing secure systems.

My cryptography work [1, 2, 3, 4, 5, 6] addresses *secure data outsourcing* and *data breach prevention* problems.

Controlled Functional Encryption: A Practical Model for Computing on Encrypted Data. I built a system that enables patients to securely outsource their genomic data to hospitals so that medical professionals can only compute personalized medicine tests allowed by the patients and hackers who break in learns nothing. A personalized medicine test determines the disease risk of an individual by computing the inner-product of genome variants and disease marker vectors. Along with a physician and genomicist, I formulated requirements for a practical solution. After studying existing approaches, I realized that new tools are required for the solution. I developed a new cryptographic model called Controlled Functional Encryption (CFE) [2], which is a relaxation of Functional Encryption (FE). Both CFE and FE are cryptographic models for computing on encrypted data. In FE, the party computing the function uses the same key to compute a function on multiple ciphertexts; in CFE, a fresh key is required for each new ciphertext. For many real applications, this change in model is acceptable and enables development of efficient schemes based on standard cryptographic assumptions, several magnitudes faster than state-of-the-art FE schemes. I developed a scheme that computes the inner-product of two vectors by cleverly using public key encryption and additive secret-sharing. I also developed a scheme for arbitrary functions based on a careful combination of public key encryption and Yao's garbled circuits. Using my Java implementation (open source), a laptop computes personalized medicine tests on full scale human genome data *in less than a second*. CFE is not limited to genomic applications and can be used in many other applications. This is representative of my work with domain experts where I designed a novel application informed cryptographic model satisfing the requirements of real applications.

Blind Storage: Searching Encrypted Data. Cloud storage is an economical, reliable, and universally accessible alternative to local storage. However, the lack of confidentiality from the cloud provider prevents its widespread adoption. Standard encryption provides confidentiality but renders data unsearchable. There has been a lot of work on

Symmetric Searchable Encryption (SSE) that enables efficient search on encrypted data. After studying SSE, I discovered that existing schemes were incompatible with cloud storage APIs. All prior schemes required both cloud storage and compute services, which increases cost, attack surface, and response time. More importantly, due to high latency and monetary cost of outbound data transfer, it limits customers to cloud providers offering both storage and compute services, such as Amazon, and excludes storage-only services, such as Dropbox. I developed the first SSE system that is compatible with cloud storage APIs [1]. I designed a novel secure storage primitive, called Blind Storage, and used it as a black box to construct an SSE scheme. My SSE scheme is one of the most efficient SSE schemes for single keyword queries; it has less than 10% overhead over plaintext search. It supports additions and deletions of documents. The system can also be used to protect local storage infrastructure to avoid data breaches. I built Blind Storage system and SSE on top of it in C++ and the code is open source.

This work also helped educate students about searchable encryption. In Spring 2014, one out of three course projects in the UIUC undergraduate security class was to implement my scheme. I helped design the project.

CURIOUS: A Novel Oblivious RAM System. Symmetric Searchable Encryption could leak information to an adversary who observes access patterns in addition to encrypted data. Oblivious RAM (ORAM) is a tool that hides access patterns. In the last few years, many ORAM schemes have been proposed and all of them use bandwidth as a performance metric. It was unclear whether bandwidth is a sound proxy metric for real performance metrics, such as response time, requests/second, and monetary cost. Moreover, no prior ORAM systems work with real cloud storage. To understand if state-of-the-art ORAM schemes are aligned with the constraints of cloud storage services, we implemented four representative ORAM schemes as full-fledged systems to work with Amazon S3 and evaluated them on real applications, such as web, file, database, and email servers [4]. The results were surprising; we found that bandwidth does *not* correlate with real performance metrics: response time, requests/second, and monetary cost; due to the misalignment between ORAM design and how cloud storage works. I used these findings to develop a novel modular ORAM framework, called CURIOUS, that is aligned with how cloud storage works. It *is the most economical and efficient ORAM scheme for cloud storage to date*. CURIOUS enjoys other crucial properties needed by real applications which prior ORAM schemes lack, such as ability to expand size, access data without waiting for previous requests to complete, and recover from a client crash. CURIOUS and the four representative ORAM systems are open source.

Blind Storage and CURIOUS are examples of my work where I used system techniques to develop cryptography.

Cryptanalysis of Influential Systems. The most popular solution to preserve confidentiality against cloud storage providers and prevent data breaches is property-preserving encrypted database systems (PPE-EDBs) because they promise *(i)* security, *(ii)* no change to applications or database servers, *(iii)* minimal performance overhead, and *(iv)* support for a large class of SQL queries. Many such systems are available; for example, CryptDB, Microsoft Research Cipherbase, Google Encrypted BigQuery, and SAP SEEED. In fact, Microsoft SQL 2016 will be shipping with such a system called Always Encrypted, which is advertised as one of the flagship benefits[1]. It is well known that property-preserving encryption (PPE) leaks information, but it was unclear what this leakage meant for real applications. Because of the tremendous interest from industry and push for use in applications such as healthcare[2], it was crucial to understand what this leakage means for such applications. I analyzed PPE-EDB systems using electronic medical records (EMR) as a concrete application [3]. EMR is a real application where security and privacy are critical, is representative of many other applications, and is a prime market for such systems. I used well known and novel techniques to design inference attacks against these systems using real data from 200 U.S. hospitals. An inference attack uses ciphertexts along with auxiliary information to recover plaintexts. My attacks use real auxiliary data publicly available over the Internet. In short, our attack framework is the least any attacker can do and uses the weakest threat model such systems are designed for. Our attacks demonstrate that an alarming amount of information is revealed from these systems: *one attack recovered more than* 80% *of patient records for all* 200 *hospitals*. This work was covered by *Forbes, Ars Technica, The Register, Golem,* and *ComputerWorld*. Based on press reports, blog posts, and personal communication with companies, *the paper [3] sent a strong message to industry and researchers from other areas that PPE-EDBs are* not *secure enough for sensitive applications, such as electronic medical records.* The paper was invited to be presented at the 2016 Real World Crypto Conference.

Systems Security

My Approach. In the design of commodity computer systems, security competes with various constraints including price, development time, and usability. As a result, billions of computer systems are shipped with security flaws. Until we can build secure-by-design *commodity* computer systems, we have to live with vulnerable computer systems. Security flaws can be due to bugs, design choices, deployment ecosystem, or unanticipated use of the system. I study

[1] For example, see here or here (video demo: starts at 18:32).

[2] Microsoft SQL 2016 Always Encrypted is advertised using a healthcare application.

fundamental security flaws in mainstream systems, such as Android; design solutions to fix these flaws with no or minimal changes to the underlying system; and work with relevant companies to fix these flaws in actual products.

My systems security work [7, 8, 9, 10, 11, 12] answers an important question: *"What threats do an unprivileged malicious application pose to an Android user* and *how to thwart them?"*

Runtime Information Gathering (RIG). RIG attacks enable a malicious *unprivileged* app to steal sensitive information from other apps. Operating systems are designed to isolate different processes to prevent information leakage; however, processes share some resources. Some of these shared resources, called side channels, lead to serious privacy breaches. Side channels are a well known type of RIG attacks. I showed how harmless Android resources become dangerous side channels for privacy breaches due to explosive growth in apps and rich public background information. I have designed several previously unknown **side channel** attacks and built defense systems to prevent them [7, 12]. I have also studied **external channel** RIG attacks and designed defense systems to prevent them [8, 11, 12]. An external channel, such as Bluetooth, NFC, and audio jack, enables a smartphone to connect to external accessories. I was the first to discover threats posed by external channels [8].

Attacks. Android uses a plethora of unprotected public resources to share information across applications. These shared resources are important; for example, Android uses speaker status (on/off) to pause music for an incoming call. I designed four distinct attacks exploiting new, unexpected side channels to show how an *unprivileged* malicious app can steal sensitive information, such as the smartphone user's identity, location, diseases, financial information, and driving routes [7]. In one attack, I showed how an *unprivileged* malicious app observing just the speaker status can record the time the speaker is on when the Google Maps app is playing audio directions, such as "turn right on Green street." With a carefully designed attack, this seemingly benign information reveals the driving route of the user. I also demonstrated the threat side channels pose to Android-controlled Internet of Things (IoT) devices [12]. I designed attacks against two state-of-the-art IoT devices: Nest Protect (a fire alarm) and Belkin NetCam (an IP camera with night vision). I showed how an attacker, through an *unprivileged* malicious app, could stealthily cause the phone to *not* alert the user of a potential fire by muting sound and vibration exactly at the moment when Nest Protect sends a fire alarm. The remote sensing and actuation through smartphone are the major benefits of these IoTs and these attacks completely disable these features. The attack demos can be seen here and here.

External channels, such as Bluetooth, NFC, and audio jack, allow a smartphone to extend its capabilities. A large number of external devices are available, which are used in security critical applications such as healthcare and retail. I showed that existing commodity operating systems, like Android, do not provide sufficient security for external devices. The fundamental problem is that an external device pairs up with the phone and the phone allows any app with access to that external channel to access all devices on that channel [8]. This coarse-grained access control leads to serious attacks. Ideally, these devices should pair up with a set of user authorized apps. I showed that a malicious app with Bluetooth permission can steal data from any Bluetooth device connected to the phone on four top-of-the-line medical devices. I also demonstrated that a malicious app can help an adversary create a clone of the user's original Bluetooth device, which can then inject fake data into the original device's real app. I also conducted a measurement study to study apps of Bluetooth devices and found that all of them suffer from this problem. I extended such attacks to other channels, such as NFC, audio jack, SMS messages, and app-server sockets [11]. I conducted a preliminary study with an undergraduate student and discovered that Apple iOS devices (iPhone/iPad/iPod) may be vulnerable to similar attacks. The attack demos can be seen here and here. One of these demos won UIUC Coordinated Science Lab video of the month award.

Defense Systems. My collaborators and I built four defense systems for runtime information gathering (RIG) attacks: Mitigator [7], Dabinder [8], SEACAT [11], and App Guardian [12]. Mitigator prevents side channel attacks from per-app data-usage statistics, speaker status (on/off), and address resolution protocol (ARP) information. Dabinder thwarts Bluetooth external channel attacks by pairing Bluetooth accessories to specific applications as opposed to the common practice of pairing them to phone. SEACAT is an SE-Linux based mandatory and discretionary access control system to protect external channels, such as Bluetooth, NFC, and audio jack.

Mitigator, Dabinder, and SEACAT prevent RIG attacks for which they are designed for, but *how can we prevent unknown RIG attacks?* Side channels are notoriously hard to find. Who would think that speaker status (on/off) [7] or power usage [13] can reveal driving routes or that a gyroscope can be exploited as a microphone [14]? The known side channels are just the tip of the iceberg; there could be many more. Patching operating systems for each newly discovered side channel is cumbersome and *not* scalable. Similarly, patching operating systems for different external channels is also cumbersome. I built an application based system, called App Guardian, to prevent RIG attacks [12]. App Guardian does *not* require any change to the operating system or the applications and provides immediate protection against RIG attacks with minimal performance and power overhead. Surprisingly, App Guardian uses side channels to prevent RIG attacks. App Guardian can be downloaded from Google Play. This work was covered by *Computer World*.

Device Driver Customization. Android is the most popular smartphone platform with an about 83% market share. The key reason to such wide adoption is the open source nature of Android; however, manufacturers need to modify the open source Android to add drivers for components, such as camera and touchscreen. I showed that these modifications result in serious security flaws that allow *unprivileged* apps to take pictures and screenshots on popular phones [9]. My collaborators and I developed dynamic and static analysis tools, designed end-to-end attacks, and conducted a measurement study to show that millions of phones are vulnerable to such flaws. This is a fundamental problem with the fragmented Android ecosystem due to a large number of distinct models, release-time constraints, lack of security-savvy developers, and lack of guidance from Google. We shared these findings with Samsung and worked with them to fix these flaws. *As a direct result, Samsung has improved their customization process and over 50 million newer Samsung phones, such as Galaxy S6 and Note 4, do not have such flaws.* Samsung awarded us equipment for research as a token of appreciation. This work won the best paper award at the 2014 NYU CSAW Security Research Competition and the Feng Chen Memorial Award from CS@Illinois. The work was covered by *The Register* and *Tom's Hardware*. The attack demos can be seen here.

Mobile Push Clouds. I have also studied security problems in reputable mobile push cloud services, such as Google Cloud Messaging (GCM) and Amazon Device Messaging (ADM), and showed that an *unprivileged* malicious app can steal sensitive messages from an Android device, stealthily install or uninstall apps on it, remotely lock out its legitimate user, or even completely delete data. I designed a cryptographic defense system that seamlessly works on top of the existing infrastructure. *This work improved the security of Google, Amazon, Facebook, and Urban Airship cloud messaging services. Facebook awarded us $2,000 for this work.* The attack demos can be seen here. This is an example of my work where I used cryptography to solve a systems security problem.

Future Research

Application Informed Cryptographic Models for Computing on Encrypted Data. The purpose of scientific modeling is to capture the real world as realistically as possible. On the one hand, most cryptographic primitives model extremely hard scenarios of the problems, leading to inefficient schemes. On the other, most real applications can be modeled more simply, exploiting unique opportunities offered by these applications. This gap in modeling prevents cryptographic schemes from being used in real applications. I plan to bridge this gap by developing models that capture real applications faithfully and allow for efficient constructions. My work on Controlled Functional Encryption [2] is one example of a practical model for computing on encrypted data. I plan to investigate an adaptation of a fully homomorphic encryption model where the client can do a small amount of work and interact with the server but outsource the majority of the computation to the server. This model can support many applications, including secure cloud computation. I envision developing a complier that would take legacy programs and generate code for the client and server automatically, so the potential adopters do not have to port their applications. I am also developing a secure data outsourcing solution for sensitive applications, such as electronic medical records, where access patterns, data sizes, and timing information leaks sensitive patient information such as patients' diseases and hospital quality measurements, such as the number of patients died in the hospital. Based on existing techniques, such as Oblivious RAM, which only hide access patterns, an efficient solution that hides length and timing information as well seems formidable, but by developing novel techniques that exploit application domain information, I have quite promising preliminary results. I believe that application informed models for computing on encrypted data will lead to faster translation into practice.

Making Cryptography Efficient through Principled Security Relaxations. The traditional goal of cryptography is to develop perfectly secure schemes; however, recently there has been a lot of progress on cryptographic schemes with weaker security guarantees, such as property-preserving encryption and symmetric searchable encryption, which intentionally leak information for efficiency. Due to their efficiency, such schemes are gaining tremendous interest from industry, government, and research community. The fundamental question that arises is, *what are the security implications of such leakage for real applications?* Little has been done to explore this question. I plan to conduct an in-depth study of the implications of information leaked by such schemes. We are still far from developing practical leakage-free schemes for problems such as encrypted search; for example, I show that even the most efficient Oblivious RAM scheme cannot reduce leakage in symmetric searchable encryption with performance better than streaming all outsourced data [5]. Therefore, I plan to develop reasonable leakage notions and a framework to understand the implications of such leakage for real applications. Finally, I plan to develop efficient cryptographic schemes for such reasonable leakage notions. I am currently working on understanding leakage of property-preserving and searchable encryption. I am also developing more expressive, secure, and efficient searchable encryption schemes.

Safety, Security, and Privacy in the Internet of Things Age. Most Internet of Things (IoT) devices are ordinary devices but with an Internet connection; for example, Belkin WeMo Switch is an electricity outlet that connects to the Internet to enable remote control. The fundamental difference here is the Internet connection, which enables users to remotely control and receive information from the device, and is the weakest link in the security of these devices.

If attackers compromise an IoT device, they can *remotely* control it to learn private information or potentially cause hazards, such as fire and electric shock. I plan to study fundamental limitations of IoT devices that could lead to such vulnerabilities and design defense systems to prevent such attacks to provide better safety, security, and privacy. The goal of the project is to make IoT devices at least as safe, secure, and private as their non-connected counterparts. I have already done some work in this direction. Along with three undergraduate students, I have conducted preliminary research on the safety of IoT devices. I studied several popular IoT devices and found several vulnerabilities. I am also developing a centralized defense system that would prevent attacks, even if the IoT devices are vulnerable.

IoTs generate a large amount of data that are stored by IoT service providers. Such data are useful for analytics, recommendations, and personalization; for example, Nest Thermostat creates a personalized temperature schedule based on previous temperature settings and reports energy usage history of the heating system. However, such data reveal a lot of private information about one's life. I plan to develop novel cryptographic models and practical schemes to enable such computation on encrypted data in a privacy-preserving fashion.

Technology Policy. I am also interested in developing technology-informed policy solutions for security and privacy problems. Technological solutions often need appropriate laws and regulations in place to fully protect people. For instance, privacy enhancing technologies can protect digitized DNA but legal protection is required to safeguard chemical DNA biospecimens. I am working on policy solutions for IoT devices, bug bounty programs, and genomic privacy. I collaborate with a lawyer for bug bounty and genomic privacy policy work. *My IoT security policy proposal was one of the five finalists at the 2014 NYU CSAW Policy Competition*.

Both systems security and cryptography are essential for a more secure and privacy-preserving world. My systems security work has significantly improved security and privacy for millions of Android users, and shows the importance and impact of such work. I believe that my vision for cryptography research will enable faster translation of cryptographic research into real applications. I have also invested considerable time in studying crucial application areas, such as healthcare and genomics [15, 16]. I plan to foster synergies between cryptography, systems, and application areas, such as healthcare and genomics, and ultimately make the world a safer, more secure place.

References

[1] Muhammad Naveed, Manoj Prabhakaran, and Carl Gunter. Dynamic searchable encryption via blind storage. In *IEEE Symposium on Security and Privacy (SP)*, pages 639–654, 2014.

[2] Muhammad Naveed, Shashank Agrawal, Manoj Prabhakaran, Xiaofeng Wang, Erman Ayday, Jean-Pierre Hubaux, and Carl Gunter. Controlled functional encryption. In *ACM Conference on Computer and Communications Security (CCS)*, pages 1280–1291, 2014.

[3] Muhammad Naveed, Seny Kamara, and Charles V Wright. Inference attacks on property-preserving encrypted databases. In *ACM Conference on Computer and Communications Security (CCS)*, pages 644–655, 2015.

[4] Vincent Bindschaedler, Muhammad Naveed, Xiaorui Pan, XiaoFeng Wang, and Yan Huang. Practicing oblivious access on cloud storage: the gap, the fallacy, and the new way forward. In *ACM Conference on Computer and Communications Security (CCS)*, pages 837–849, 2015.

[5] Muhammad Naveed. The fallacy of composition of oblivious ram and searchable encryption. Cryptology ePrint Archive, Report 2015/668, 2015.

[6] Christopher Fletcher, Muhammad Naveed, Ling Ren, Elaine Shi, and Emil Stefanov. Bucket oram: Single online roundtrip, constant bandwidth oblivious ram. Cryptology ePrint Archive, Report 2015/1065, 2015.

[7] Xiaoyong Zhou, Soteris Demetriou, Dongjing He, Muhammad Naveed, Xiaorui Pan, XiaoFeng Wang, Carl A Gunter, and Klara Nahrstedt. Identity, location, disease and more: Inferring your secrets from android public resources. In *ACM Conference on Computer and Communications Security (CCS)*, pages 1017–1028, 2013.

[8] Muhammad Naveed, Xiaoyong Zhou, Soteris Demetriou, XiaoFeng Wang, and Carl A Gunter. Inside job: Understanding and mitigating the threat of external device mis-bonding on android. In *ISOC Network and Distributed System Security Symposium (NDSS)*, pages 23–26, 2014.

[9] Xiaoyong Zhou, Yeonjoon Lee, Nan Zhang, Muhammad Naveed, and XiaoFeng Wang. The peril of fragmentation: Security hazards in android device driver customizations. In *IEEE Symposium on Security and Privacy (SP)*, pages 409–423, 2014.

[10] Tongxin Li, Xiaoyong Zhou, Luyi Xing, Yeonjoon Lee, Muhammad Naveed, XiaoFeng Wang, and Xinhui Han. Mayhem in the push clouds: Understanding and mitigating security hazards in mobile push-messaging services. In *ACM Conference on Computer and Communications Security (CCS)*, pages 978–989, 2014.

[11] Soteris Demetriou, Xiaoyong Zhou, Muhammad Naveed, Yeonjoon Lee, Kan Yuan, XiaoFeng Wang, and Carl A Gunter. What's in your dongle and bank account? mandatory and discretionary protection of android external resources. In *ISOC Network and Distributed System Security Symposium (NDSS)*, 2015.

[12] Nan Zhang, Kan Yuan, Muhammad Naveed, Xiaoyong Zhou, and XiaoFeng Wang. Leave me alone: App-level protection against runtime information gathering on android. In *IEEE Symposium on Security and Privacy (SP)*, pages 915–930, 2015.

[13] Yan Michalevsky, Aaron Schulman, Gunaa Arumugam Veerapandian, Dan Boneh, and Gabi Nakibly. Powerspy: Location tracking using mobile device power analysis. In *USENIX Security Symposium*, pages 785–800, 2015.

[14] Yan Michalevsky, Dan Boneh, and Gabi Nakibly. Gyrophone: Recognizing speech from gyroscope signals. In *USENIX Security Symposium*, pages 1053–1067, 2014.

[15] Muhammad Naveed. Hurdles for genomic data usage management. In *IEEE Security and Privacy Workshops (SPW)*, pages 44–48, 2014.

[16] Muhammad Naveed, Erman Ayday, Ellen W Clayton, Jacques Fellay, Carl A Gunter, Jean-Pierre Hubaux, Bradley A Malin, and XiaoFeng Wang. Privacy in the genomic era. *ACM Computing Surveys (CSUR)*, 48(1):6, 2015.