REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
09-08-2017	Final Report	1-May-2013 - 30-Apr-2017

4. TITLE AND SUBTITLE
Final Report: Jana: Confidential Communications on Social Networks

5a. CONTRACT NUMBER
W911NF-13-1-0120

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
206022

6. AUTHORS

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
University of California - Riverside
200 University Office Building

Riverside, CA 92521 -0001

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
62954-CS-REP.3

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Srikanth Krishnamurthy
UU	UU	UU	UU		19b. TELEPHONE NUMBER 951-827-2348

RPPR Final Report
as of 17-Nov-2017

Agency Code:

Proposal Number: 62954CSREP **Agreement Number: W911NF-13-1-0120**
INVESTIGATOR(S):

Name: Harsha Madhyastha
Email: harsha@cs.ucr.edu
Phone Number: 9518272479
Principal: N

Name: Srikanth Krishnamurthy
Email: krish@ucr.edu
Phone Number: 9518272348
Principal: Y

Organization: **University of California - Riverside**
Address: 200 University Office Building, Riverside, CA 925210001
Country: USA
DUNS Number: 627797426 EIN: 956006142

Report Date: 31-Jul-2017 Date Received: 09-Aug-2017
Final Report for Period Beginning 01-May-2013 and Ending 30-Apr-2017
Title: Jana: Confidential Communications on Social Networks
Begin Performance Period: 01-May-2013 **End Performance Period:** 30-Apr-2017
Report Term: 0-Other

Submitted By: Srikanth Krishnamurthy Email: krish@ucr.edu
Phone: (951) 827-2348

STEM Degrees: 3 **STEM Participants:** 3

Major Goals: The inability of users to communicate secretly on online social networking (OSN) platforms is a key obstacle to overcome, if these platforms are to be used in the tactical world. While exclusive military networks such as MilBook and Service-Connected [exist, they do not support secret group communications. Furthermore, access to such social networks via mobile platforms raises a series of concerns like leakage of private data. Finally, any secret communications can be blocked by censorship firewalls that maintain state and look for specific keywords or features. In this project, we try to address all of these issues.

Accomplishments: We have the following contributions, which we describe in some detail in the final report (Further fine-grained details can be found in the papers that were published on these): (A) We design a system that facilitated in-band embedding of secrets (limited in size) in shared content on OSNs; (B) We design Hermes, a cost-effective decentralized OSN architecture that allows exchange of secret information among a group, without revealing any details with regards to either group membership or posting patterns (C) We design ZapDroid that quarantines OSN or other applications on smartphones to reduce their attack surface, and thereby prevent them from leaking any information that needs to be secret and (D) We perform an in depth measurement study that characterizes what firewalls such as the Great Firewall of China might do in order to prevent confidential communications, and how to evade such preventive censorship.

To date, we have four conference papers and one journal paper either published or accepted for publication. They are mostly in top tier conference venues viz., IEEE CNS 2014, ACM UbiComp 2015, SecureComm 2015 and in ACM IMC 2017, and a top journal viz., the IEEE Transactions on Mobile Computing.

Training Opportunities: The work also supported multiple graduate students. Graduated PhD students Jianxia Ning is now at Cisco, Indrajeet Singh joined Akamai, Masoud Akhoondi.

Results Dissemination: There were conference and journal papers that were published and dissseminated. All of these except one are available on the PIs website. The latest paper which will appear in IMC 2017 will be made available after changes are made to produce the camera ready version.

RPPR Final Report
as of 17-Nov-2017

Honors and Awards: The paper in IEEE CNS 2014 was awarded the "Best Paper Runner Up Award."

Protocol Activity Status:

Technology Transfer:
Nothing to report. However, the code that was produced during this project can be made available on request.

Final Report: Proposal: 62954CSREP
Title: Jana: Confidential Communications on Social Networks

The inability of users to communicate secretly on online social networking (OSN) platforms is a key obstacle to overcome, if these platforms are to be used in the tactical world. While exclusive military networks such as MilBook [1] and Service-Connected [2] exist, they do not support secret group communications. Furthermore, access to such social networks via mobile platforms raises a series of concerns like leakage of private data. Finally, any secret communications can be blocked by censorship firewalls that maintain state and look for specific keywords or features. In this project, we try to address all of these issues. We have the following contributions, which we describe in some detail in the final report (Further fine-grained details can be found in the papers that were published on these): (A) We design a system that facilitated in-band embedding of secrets (limited in size) in shared content on OSNs; (B) We design Hermes, a cost-effective decentralized OSN architecture that allows exchange of secret information among a group, without revealing any details with regards to either group membership or posting patterns (C) We design ZapDroid that quarantines OSN or other applications on smartphones to reduce their attack surface, and thereby prevent them from leaking any information that needs to be secret and (D) We perform an in depth measurement study that characterizes what firewalls such as the Great Firewall of China might do in order to prevent confidential communications, and how to evade such preventive censorship.

To date, we have four conference papers and one journal paper either published or accepted for publication. They are mostly in top tier conference venues viz., IEEE CNS 2014 [3], ACM UbiComp 2015 [4], SecureComm 2015 [5] and in ACM IMC 2017 [6], and a top journal viz., the IEEE Transactions on Mobile Computing [7].

1. Secret Message Sharing Using Online Social Media

In this work, we undertake a study to obtain a fundamental understanding of the challenges in creating a viable covert channel for confidential communications on OSNs or other photo-sharing sites. These challenges include the following. First, photo-sharing sites often process uploaded images. While some of the processing functions are clearly specified on the photo-sharing sites (e.g., any photo exceeding a pre-specified size limit will be re-sized), not all such functions are publicly known. These (possibly unknown) processing functions often interfere with the use of steganography, which we use to create the covert channel. Second, it is well known that steganography does not offer perfect secrecy. Censors can try to read the embedded message by applying a variety of extraction algorithms on a carrier image. Thus, to prevent exposure in the rare cases of interception, one will have to encrypt the secret information embedded in the shared photographs. Encryption requires the establishment of secret keys between communicating entities, for which prior work often assumes the existence of an out-of-

band channel. However, the creation of such an out-of-band channel is difficult because phone calls, e-mail exchanges, and Internet communication may be monitored.

Our next goal is to address the above challenges and build a framework for confidential communication on public photo-sharing sites. Towards this, we make three key contributions. First, to understand how secretly embedded messages are affected by processing done on photo-sharing sites, we perform an in-depth measurement study. We analyze photos uploaded on four popular sharing sites—Google+, Facebook, Twitter, and Flickr. We consider both photos wherein secret information is embedded and photos without any such embedding. We observe that, while the integrity of hidden messages is preserved on some sites (e.g., Google+), other sites (e.g., Facebook and Flickr) perform various processing functions on uploaded images and hence the extraction of secret messages from downloaded images fails. Our study sheds light on the processing performed on different sites and provides an understanding of why secret content is affected.

Second, based on the understanding obtained above, we propose simple changes to the steganographic encoding process, which ensure that unlike prior approaches, the embedded secret messages survive the image processing performed by photo- sharing sites. Specifically, unlike prior approaches that modify the least significant bit (LSB) of the DCT co-efficients of an image, we propose to modify the second least co-efficient bit (2-LSB); this ensures that the secret message is retained in spite of processing done on the OSN or photo-sharing site. The robustness offered allows the usage of less intense forward error correction codes (FEC) thereby increasing the secret message carrying capacity in an image.

Though simple, our approach is not apparent without the detailed study on the different photo-sharing sites. Importantly, this improved reliability does not come at the expense of greater likelihood of detection of hidden messages. We evaluate our approach by applying two state-of-the-art steganalysis tools and observe that, for a fixed amount of secret data, the likelihood of detecting secret information embedded with our approach is similar (or even lower in some cases) to the probability of detection when prior approaches for steganographic embedding are applied (while surviving the processing done on the site). In the table below we show the reduction in the FEC overhead and the higher resistance to steganalysis with our 2-LSB approach.

Table 1: The 2-LSB approach offers lower detection likelihood and FEC overhead compared to traditional LSB schemes.

Method	BER	FEC overhead	Detection likelihood (ensemble classifier)	Detection likelihood (StegAlyzerAS)
LSB	0.15239	0.0	0.44	0.69
LSB + 2-LSB	0.08144	0.0	0.47	0.68
2-LSB	0.00968	0.0	0.50	0.63
LSB+FEC *[15,13]*	0.09375	0.1333	0.45	0.69
LSB+FEC *[15,11]*	0.01125	0.2667	0.48	0.69
LSB+FEC *[7,3]*	0.0	0.5714	0.53	0.72
LSB+2-LSB+FEC *[15,13]*	0.02993	0.1333	0.50	0.68
LSB+2-LSB+FEC *[15,11]*	0.0	0.2667	0.51	0.69
2-LSB+FEC *[15,13]*	0.0	0.1333	0.51	0.63

Finally, as discussed above, encrypting the secretly embedded messages is a must. Therefore, to enable recipients of the shared photo to extract the raw data, a key exchange between the sender and recipients is essential. Towards this, we propose a protocol for bootstrapping the private communication without any out-of-band channel (unlike what is assumed in prior work). Our bootstrapping phase uses the very same channel, i.e., uploaded images, to exchange keys. The work was published in IEEE CNS 2014 [3] and was awarded the "best paper runner up award."

2. Reducing the attack surface of mobile applications to prevent leakage of confidential information with ZapDroid

The Google Play Store has more than 1.3 million apps, and the number of app downloads is roughly 1 billion per month. However, after users interact with many such apps for an initial period following the download, they almost never do so again. Statistics indicate that for a typical app, less than half of the people who downloaded it use it more than once. Reports also suggest that more than 86 % of users do not even revisit an app, a day after the initial download. Uninstall rates of apps however (longer term), of about 15 to 18 % are considered high. This means that users often leave installed apps on their phones. Many of these applications are social network applications. Users install portals to OSNs, many times to never use them again.

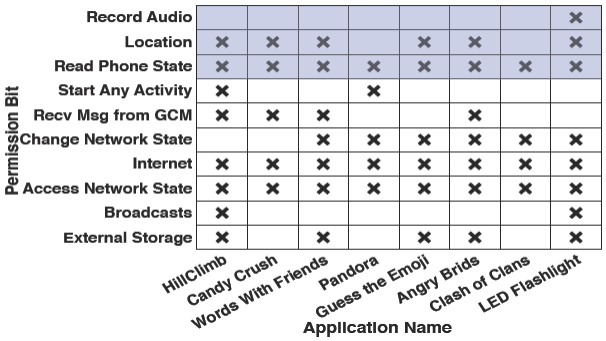
More generally, users may only interact with some downloaded apps or OSN portals infrequently (i.e., not use them for prolonged periods). These apps continue to operate in the background and have significant negative effects (e.g., leak private information or significantly tax resources such as the battery). Unfortunately, users are often unaware of such app activities. We call such seldom-used apps, which indulge in undesired activities, "zombie apps."

In this work, we seek to build a framework, ZapDroid, to identify and subsequently quarantine such zombie apps to stop their undesired activities. Since a user can change her mind about whether or not to use an app, a zombie app must be restored quickly if the user chooses. The classification of an app as a zombie app is inherently subjective. An app unused for a prolonged period should be classified as a zombie app if its resource usage during the period is considered significant and/or if its access of private data is deemed serious. Thus, instead of automatically classifying zombie apps, we seek to empower the user by exporting the information that she would need to make this decision. Moreover, the way in which a zombie app should be quarantined depends on whether the user is likely to want to use the app again in the future (e.g., an OSN app that the user tried once and decided is not interesting vs. a Skype app that the user uses infrequently). The apps that a user is likely to use again fairly soon must not be fully uninstalled; real time restoration (when needed) may be difficult if there is no good network connectivity. We seek to enable users to deal with these different scenarios appropriately.

Challenges: We address many challenges en route designing and building ZapDroid. First, to motivate the need for ZapDroid, we ask the question: "How often do users download apps and leave them on their phones, and how do these apps adversely affect the user in terms of consuming phone resources and privacy leakage?" We address this challenge via an extensive user study. Next, we ask "How can we detect background apps that either consume high resources or violate privacy in a lightweight manner?" Such apps are the candidates for being zombie apps. Continuous app monitoring can be too resource-intensive to be practical. Further, application-level implementations are infeasible since Android does not allow any app to track the permission access patterns of other apps. The third challenge is to effectively quarantine apps, i.e., "How can we design effective methods to ensure that zombie apps are quarantined and remain in that state unless a user wants them restored?" With current approaches, apps' background activities are constrained only temporarily, until they are woken up due to time-outs or external stimuli. Finally, " How can we restore previously quarantined apps in a timely way, even under conditions of poor network connectivity (if the user desires)?" The restored app must be in the same state that it was in prior to the quarantine. Reinstalls from the Google Play Store can be hard if network connectivity is poor and hence, should not be invoked when it is highly likely that the user will restore the app. Further, clean uninstalls can result in loss of application state.

Contributions: Our framework, ZapDroid, addresses the above challenges and allows users to effectively man- age infrequently used apps. In designing and building ZapDroid, we make the following contributions.

. *Showcase the unwanted behaviors of candidate zombie apps*: We conduct a month-long study where we enlist 80 users on Amazon's Mechanical Turk to download an app (TimeUrApps) we develop. TimeUrApps identifies (other) apps that have not been used for the month, on the users' phones. Once we identify these apps, we undertake an in-house, comprehensive experimental study to understand their behaviors when they are not being actively used. We find that a zombie app on a typical user's phone (the median user in our targeted experiments) could consume as much as 58 MB of bandwidth and more than 20% of the total battery capacity in a day. Further, many of such apps access information such as the user's location and transmit this over the network. The following figure shows some of the permissions that some of the apps in our study obtain that can leak private information (e.g., recorded audio). Further, there is no need for these permissions to be granted to these apps.



. *Identify candidate zombie apps that are most detrimental to the user's device*: We design mechanisms that are integrated within the Android OS (we make changes to the underlying Android Framework's activity management, message passing, and resource management components) to track (i) a user's inter- actions with the apps on her device to identify unused apps, and (ii) the resources consumed and the private information accessed by these apps to determine candidate zombie apps, from which the user can choose to quarantine those she considers to be zombie apps.

. *Dynamically revoke permissions from zombie apps, or offload them to external storage*: The quarantine module of ZapDroid is invoked based on user input. She has to categorize a zombie app as either "likely to restore" or "unlikely to restore"; the two categories are quarantined differently. For the first category, only permissions enjoyed by the zombie app are revoked but all relevant data/binaries are stored on the device itself. For the second category, the associated data/binaries are removed from the device and user-specific app state is moved to either the cloud or to a different device (a desktop) owned by the user; the transfers occur when there is good network connectivity (e.g., WiFi coverage or a USB cable).

. *Restore an app with all its permissions if the user desires*: ZapDroid restores a zombie app on the user's device if she so desires. The state of the app is identical to that prior to the quarantine. For the "likely to restore" category of apps, the restoration time is < 6ms. For the "unlikely to restore" category, restoration depends on the network connectivity to where the app was stored during the quarantine and is typically on the order of a few seconds.

We evaluate ZapDroid via extensive measurements on 5 different Android smartphones (from 4 vendors). We show that the overhead of ZapDroid is low (< 4% of the battery is consumed per day). We show that ZapDroid saves more than 2× the energy expended due to zombie app activities, as compared to other popular apps on the Google Play Store used to kill undesired background processes; further, unlike these apps, it prevents access to undesired permissions by the zombie apps.

Note that ZapDroid does not require changes to an external cloud store (for quarantine or restoration); all modifications are made only in the Android OS. We envision that the features of ZapDroid will be useful in general, and our hope is that this could lead to an integration of the functions within the Android OS. A preliminary paper on ZapDroid appears in ACM UbiComp 2015 [4] and an extended version appears in the IEEE Transactions on Mobile Computing [7].

3. Privacy Preservation of Online Social Media Conversations

Today, leakage of information from OSN servers, coupled with the need for OSN providers to mine user data (e.g., for targeted advertisements), has concerned users. While posting encrypted data on OSNs can work in theory, it compromises the profit motives of an OSN if done at scale. Alternatively, one could share private content with

OSN friends by storing data outside the OSN provider's control. Prior works that follow this approach either store private content in the cloud or across client machines. The former simply leaks private information to the cloud providers in lieu of the OSN providers, and also increases user costs. The viability of an approach based on the latter depends on the availability of consistent access to client machines.

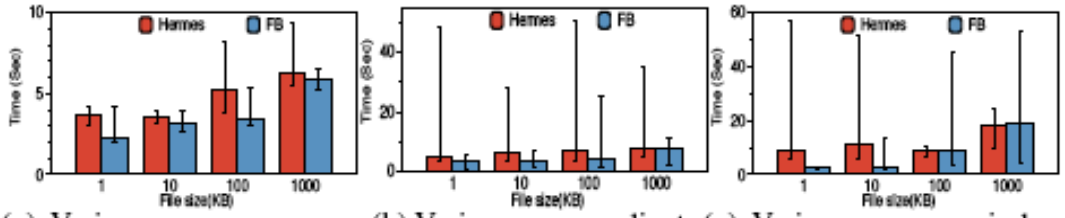
In this work, we design a decentralized OSN architecture, Hermes, with cost-effective privacy in mind. Hermes seeks to ensure that both the content shared by a user and her sharing habits are kept private from both the OSN provider and undesired friends. In doing so, Hermes seeks to (i) minimize the costs borne by users, and (ii) preserve the interactive and chronologically consistent conversational structure offered by a centralized OSN.

Hermes uses three key techniques to meet these goals. First, it judiciously combines the use of compute and storage resources in the cloud to bootstrap conversations associated with newly shared content. This also supports the high availability of the content. Second, it employs a novel cost-effective message propagation mechanism to enable dissemination of comments in a timely and consistent manner. It identifies and purges (from cloud storage) content that has been accessed by all intended recipients. Lastly, but most importantly, Hermes carefully orchestrates how fake postings are included in order to hide sharing patterns from the untrusted cloud providers used to store and propagate content, while minimizing the additional costs incurred in doing so. A key feature of Hermes is its flexibility in deployment; it can either be implemented as a stand-alone distributed OSN or as an add-on to today's OSNs like Facebook (while maintaining the decentralized nature of content sharing). To summarize, our contributions are:

Design of Hermes: As our primary contribution, we design Hermes. It utilizes ex- tremely small amounts of storage, bandwidth, and computing on the cloud to facilitate real-time, consistent and anonymous exchange of private content. Importantly, Hermes ensures that cloud providers cannot discover the users involved in private conversations and is robust to the intersection attack (where an attacker can correlate the participants across different conversations).

Analyzing OSN data to determine resource requirements: Based on 1.8 million posts crawled from Facebook, we 1) perform an analysis to determine key parameters for implementing Hermes, and 2) conduct realistic simulations to show that (a) Hermes effectively anonymizes users' sharing patterns and (b) Hermes's use of cloud resources is low enough to facilitate its practical deployment. Our analysis suggests that, for 90% of users, Hermes would typically require 1) cloud storage of much less than 5 MB, and 2) a compute instance on the cloud that is active for roughly 4 days every month. This corresponds to a monthly cost of less than $5 per user. With this budget, Hermes ensures that cloud service providers are unable to guess the members or the group size of any private conversation. If the cloud provider attempts to randomly guess the group members, it is correct less than 15% of the time.

Implementation and evaluation: We implement a prototype of Hermes as a rudimentary add-on to Facebook. Our evaluations show that Hermes incurs low cost, and the user experience, in terms of delays, is similar to that with Facebook as shown in the figures below. This work appears in SecureComm 2015 [5].



(a) Variance across measurements (b) Variance across clients (c) Variance across wireless clients

1. INTANG: A Practical measurement based tool for censorship evasion

Internet censorship and surveillance are prevalent nowadays. Censorship systems such as the Great Fire- wall (GFW) of China, have the capability of analyzing terabyte-level traffic across the country in realtime. Protocols with plaintext (e.g., HTTP, DNS, IMAP), are directly subject to surveillance and manipulation by the governors, while protocols with encryption (e.g., SSH, TLS/SSL, PPTP/MPPE) and Tor, can be identified via traffic fingerprinting, leading to subsequent blocking at the IP-level.

The key technology behind these censorship systems is Deep Packet Inspection (DPI), which also powers Network Intrusion Detection Systems (NIDS). As previously reported, most censorship NIDS are deployed “on-path" in the backbone and at border routers.

In order to examine application-level payloads, DPI techniques have to correctly implement the underlying protocols like the TCP protocol, which is the cornerstone of today’s Internet. Earlier work has shown that any NIDS is inherently incapable of always reconstructing a TCP stream the same way as its endpoints. The root cause for this is the discrepancies between the implementations of the TCP (and possibly other) protocol at the end-host and at the NIDS. Even if the NIDS perfectly mirrored the implementation of one specific TCP implementation, it may still have problems processing a stream of packets generated by another TCP implementation.

Because of such ambiguity in packets process, it is possible for a sender to send carefully crafted packets to desynchronize the TCP Control Block (TCB) maintained by the NIDS with the TCB on the receiver side. In some cases, the NIDS can even be tricked to completely deactivate the TCB (e.g., after receiving a spurious RST packet), effectively allowing an adversary to “manipulate” the TCB on the NIDS. Censorship monitors suffer from the same fundamental flaw—a client can evade censorship if the TCB on a censorship monitor can be successfully desynchronized with the one on the server. Different from other censorship evasion technologies such as VPN, Tor, and Telex that rely on additional network infrastructure (e.g., proxy node), TCB-manipulation-based

evasion techniques only require crafting/manipulating packets on the client-side and can potentially help all TCP-based application-layer protocols “stay under the radar”. Based on this idea, some prior work has explored several practical evasion techniques against the GFW, by studying its behaviors at the TCP and HTTP layers. The West Chamber Project provides a practical tool that implemented a few of evasion strategies but has ceased development since 2011; unfortunately none of the strategies are effective during our measurement. Besides these attempts, there is no recent data point showing how this evasion technique works in the wild.

In this work, we extensively evaluate the TCP-layer censorship evasion against the GFW. By testing from 11 vantage points inside China spread across 9 cities (and 3 ISPs), we are able to cover a variety of network paths that potentially include different types of GFW devices and middleboxes. We measure how TCB manipulation can help HTTP, DNS, and Tor evade the GFW.

First, we measure how existing censorship evasion strategies work in practice. Interestingly, we find that most of them no longer work well due to challenges in network conditions, interference from the network middleboxes, or more importantly, new updates to the GFW (different from models considered previously). These initial measurement results motivate us to construct probing tests to infer the “new" updated GFW model. Finally, based on the new GFW model and lessons learned from other practical challenges in deploying TCP-layer censorship evasion, we develop a list of new evasion strategies. Our measurement results show that the new strategies have a 90% or above evasion success rate. We also evaluate how these new strategies can help HTTP, DNS, Tor, and VPN evade the GFW.

In addition, during the course of our measurement study, we design and implement a censorship evasion tool which we call INTANG, integrating all of the censorship evasion strategies mentioned in this paper and is easily extensible. The tool requires zero configuration and runs in the background to help normal traffic evade censorship. We plan to open source the tool, which will support future research in this direction.

We summarize our contributions as the follows:

. We are the first to extensively measure the GFW’s behaviors with TCP-layer censorship evasion techniques.

. We demonstrate that existing strategies are either not working or are limited in practice.

. We develop an updated and more comprehensive model of the GFW based on the measurement results.

. We propose new, measurement-driven strategies that can bypass the new model.

. We measure the success rate of our improved strategy on censorship evasion for HTTP, DNS, VPN, and Tor. The results show very high success rates.

. We develop a tool to automatically measure the GFW's responsiveness, and can also be used for censorship circumvention. The tool is extensible as a framework for the integration of additional evasion strategies for future research.

Below we present a table that showcases the effectiveness of INTANG in successfully evading GFW. The work will appear in ACM IMC 2017 [6]

Vantage Points	Strategy	Success			Failure 1			Failure 2		
		Min	Max	Avg.	Min	Max	Avg.	Min	Max	Avg.
Inside China	Improved TCB Teardown	89.2%	98.2%	**95.8%**	1.7%	6.7%	**3.1%**	0.0%	5.4%	**1.1%**
	Improved In-order Data Overlapping	86.7%	97.1%	**94.5%**	2.9%	8.9%	**4.4%**	0.0%	5.2%	**1.1%**
	TCB Creation + Resync/Desync	88.5%	98.1%	**95.6%**	1.9%	7.0%	**3.3%**	0.0%	5.5%	**1.1%**
	TCB Teardown + TCB Reversal	90.2%	98.2%	**96.2%**	1.7%	5.6%	**2.6%**	0.0%	5.7%	**1.1%**
	INTANG Performance	93.7%	100.0%	**98.3%**	0.0%	3.0%	**0.9%**	0.0%	3.5%	**0.6%**
Outside China	Improved TCB Teardown	85.6%	92.9%	**89.8%**	4.6%	7.6%	**6.8%**	0.3%	6.8%	**3.5%**
	Improved In-order Data Overlapping	89.4%	96.0%	**92.7%**	1.3%	6.2%	**3.6%**	0.6%	7.0%	**3.7%**
	TCB Creation + Resync/Desync	78.1%	95.6%	**84.6%**	2.4%	18.6%	**12.9%**	0.9%	4.0%	**2.6%**
	TCB Teardown + TCB Reversal	84.6%	93.1%	**89.5%**	5.5%	8.7%	**7.1%**	0.1%	7.9%	**3.3%**

[1] U.S. military turns to social neworking to encourage sharing of official and sensitive info. http://gigaom.com/2010/01/22/u-s-military-turns-to-social-networking-to encourage-sharing-official-and-sensitive-info/.

[2] Service-connected. http://www.service-connected.com/

[3] "Secret Message Sharing Using Online Social Media," Jianxia Ning, Indrajeet Singh, Harsha Madhyastha, Srikanth Krishnamurthy, Guohong Cao and Prasant Mohapatra, IEEE CNS 2014 (Best paper runner up), San Francisco.

[4] "ZapDroid: Managing Infrequently Used Applications on Smartphones," Indrajeet Singh, Srikanth V. Krishnamurthy, Harsha Madhyastha and Iulian Neamtiu, ACM UbiComp 2015, Osaka.

[5] ``Resource Efficient Privacy Preservation of Online Social Media Conversations," Indrajeet Singh, Masoud Akhoondi, Mustafa Y. Arslan, Harsha Madhyastha, Srikanth V. Krishnamurthy, SecureComm 2015, Dallas.

[6] "Your State is not Mine: A closer look at Evading Stateful Internet Censorship," Zhongjie Wang, Yue Cao, Zhiyun Qian, Chengyu Song, and Srikanth V. Krishnamurthy, ACM IMC 2017, London (accepted to appear).

[4] ``ZapDroid: Managing Infrequently Used Applications on Smartphones," Indrajeet Singh, Srikanth V. Krishnamurthy, Harsha Madhyastha and Iulian Neamtiu, IEEE Transactions on Mobile Computing, 2017.