Announcements

- Course website is up: [http://seclab.soic.indiana.edu/SysSec/index-2.html](http://seclab.soic.indiana.edu/SysSec/index-2.html)

- Please keep monitoring the schedule online

- Course workshop
  - You will receive a workshop PC invitation by the end of the week
  - You are supposed to bid for the papers you want to present next week
  - We will announce the paper assignment before 16
  - Your presentations and reviews start on 25
How to write a good review

- Read a paper carefully
  - What is it about? Is it on a right topic? Do you understand the techniques? Do you know related research?

- Compare its merits to the state of the art
  - Marginal? Incremental? a surprise?

- Evaluate its practical impacts
  - Do you believe it works? Is the evaluation part convincing? Any chance for its adoption?

- Comment on its presentation
  - Is the paper well organized? Are the main idea and technical details clearly and adequately stated? Any writing problems?

- Help the authors improve their paper
Review: Format

- Summary of the paper
- Strengths of the paper
- Weaknesses of the paper
- Detailed comments
- Your decisions and justifications
- Suggested improvement
Soundcomber
A Stealthy and Context-Aware Sound Trojan for Smartphones

Roman Schlegel
City University of Hong Kong

Kehuan Zhang, Xiaoyong Zhou, Mehool Intwala,
Apu Kapadia, XiaoFeng Wang
Indiana University Bloomington
The smartphone in your pocket is really a computer

- 1 GHz Processor
- 512MB / 16GB
- Android OS (Linux)
No surprise malware targets smartphones

- Android malware steals info from 1’000’000 users¹
- Trojan sends premium-rate text messages²
- Security experts release Android root-kit³

But “sensory malware” can do much more

What can malware overhear?

Do you think anybody will ever figure out that I keep a spare door key in the flower pot on my front porch?

Nah, how would anybody ever find out?
Some situations are easy to recognize

“Please enter or speak your credit card number now.”

Bank
Naive approach: record and upload

1’000’000 phones ≈ 1TB/day

≈ 500KB/day
Certain combinations of permissions are suspicious

Can easily be recognized and disallowed ([5] W. Enck et al.)
Our contributions over the naive approach

- *Targeted* and *local* extraction of valuable data
- inconspicuous permissions
- stealthiness
Naive approach: record and upload

Internet

Malware Master

Data

Valuable Information
Soundcomber approach: process and upload
Two trojans are stealthier than one
Soundcomber minimizes the necessary permissions

voice-memo app
wake-up alarm app
Hotline greetings can be fingerprinted easily

“Thank you for calling...”

“Welcome to ...”

Bank 1

Bank 2
Tricking the user into installing two apps

- pop-up ad
- packaged app
Soundcomber extracts sensitive information locally

Microphone

Soundcomber App

Profile Database

Extract Data

Record Audio

Process Audio

To Deliverer

Soundcomber App extracts sensitive information locally.
Profiles allow for context aware extraction
DTMF tones are “dual tones”

- 8 frequencies
- 2 simultaneous frequencies for each digit
- used to navigate hotline menus

<table>
<thead>
<tr>
<th></th>
<th>1209 Hz</th>
<th>1336 Hz</th>
<th>1477 Hz</th>
<th>1633 Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>697 Hz</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>A</td>
</tr>
<tr>
<td>770 Hz</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>B</td>
</tr>
<tr>
<td>852 Hz</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>C</td>
</tr>
<tr>
<td>941 Hz</td>
<td>*</td>
<td>0</td>
<td>#</td>
<td>D</td>
</tr>
</tbody>
</table>
Soundcomber dynamically adjusts thresholds to detect faint tones

audio signal

<table>
<thead>
<tr>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>amplitude [V]</td>
</tr>
</tbody>
</table>

Goertzel's algorithm

<table>
<thead>
<tr>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
</tr>
</tbody>
</table>

threshold

<table>
<thead>
<tr>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>852Hz</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1477Hz</td>
</tr>
</tbody>
</table>

threshold
Android introduces new covert channels

- vibration settings (87 bps)
- volume settings (150 bps)
- screen (5.3 bps)
- file locks (685 bps)
Vibration settings are broadcast to interested apps

<table>
<thead>
<tr>
<th>Setting</th>
<th>Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIBRATE_SETTING_ON</td>
<td>0</td>
</tr>
<tr>
<td>VIBRATE_SETTING_ONLY_SILENT</td>
<td>1</td>
</tr>
</tbody>
</table>
Volume settings can be modified and accessed by any app

<table>
<thead>
<tr>
<th>Volume</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
</tr>
<tr>
<td>1</td>
<td>001</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>7</td>
<td>111</td>
</tr>
</tbody>
</table>

Soundcomber App

Audio Manager

set volume setting

small delay

Audio Manager

get volume setting

small delay

Deliverer App
### Soundcomber is fast and accurate

<table>
<thead>
<tr>
<th></th>
<th>No Error</th>
<th>1 Error</th>
<th>&gt;= 2 Errors</th>
<th>1 missing</th>
<th>&gt;= 2 missing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Speech</strong></td>
<td>55 %</td>
<td>12.5 %</td>
<td>15 %</td>
<td>7.5 %</td>
<td>10 %</td>
</tr>
<tr>
<td><strong>Tone</strong></td>
<td>85 %</td>
<td>5 %</td>
<td>0</td>
<td>10 %</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Recording Length</th>
<th>Processing Time</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Speech</strong></td>
<td>20 s</td>
<td>7 s</td>
</tr>
<tr>
<td><strong>Tone</strong></td>
<td>45 s</td>
<td>8 s</td>
</tr>
</tbody>
</table>
Hotlines can be fingerprinted with reasonable accuracy

- 20 recorded samples of 5 different hotlines (4 each)
- 20 samples of normal conversation

<table>
<thead>
<tr>
<th>Correct</th>
<th>Missed</th>
<th>Wrong</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hotline</td>
<td>55 %</td>
<td>40 %</td>
</tr>
</tbody>
</table>

| Correct | Conversation | 100 % |
Keeping Soundcomber hidden and undetectable

- defer/throttle processing
- track user presence
- performance enhancements
Defense: disable recording when a sensitive number is called
In conclusion...

- stealthy, sensory malware is a real threat
- need to explore other such threats
- develop generalized defenses to such attacks
- Demo: http://www.youtube.com/watch?v=Z8ASb-tQVpU
Attacks against smartphones are a timely topic and the authors introduce an attack to steal sensitive data from a phone in an efficient manner. The attack scenario seems to be a bit constructed and not really general, which is my main concern with the paper.

Comments/questions regarding the paper:
- I never had to enter my credit card details or other sensitive information in an IVR system. Maybe I use my credit card in unusual ways, but can you estimate how common this threat really is? Are there any studies that show that many people actually use their sensitive data in such ways with their smartphone? Doing full speech recognition seems to be not possible on smartphones since a user will likely notice that something suspicious is going on due to the overhead. Furthermore, it might lead to quite a few FPs and the transmission overhead is higher.

- Besides installing Soundcomber, a user also needs to install the Deliverer app. I am not really convinced that this will work in practice in a reliable way since an attacker needs to convince a user to install two applications. I suggest to study other exfiltration techniques, maybe there is a way in which Soundcomber can exfiltrate information in a more stealthy way.
Can an attacker create the necessary information to fingerprint a phone menu system (IVR profile from Section 3.3) in an automated way or is this a manual task? Missing 40% of the hotline calls seems to be rather high, an attacker might miss a significant fraction of valuable data.

Covert channels are a well-studied topic. The covert channels discussed in the paper are not particularly interesting, basically the different sensors available on a smartphone are used to transport information. I recommend to tone down this part of the paper a bit since the novelty seems to be a bit limited. Furthermore, the computation at the beginning of Section 4 also sounds a bit exaggerated.

The authors state that a challenge for Soundcomber is the fact that "the tones are drowned out by background noise in the recordings". However, in the demo video there is no background noise at all and the volume of the key presses is rather high. How good does the approach work if it is not carried out in a lab environment? Some "field test" in a public space would be good.
Positive Review

Soundcomber is a sound Trojan which use a few innocuous permissions like accessing the microphone and yet can extract credit card and PIN numbers from tone- or speech-based IVR systems. It records the microphone input during the call and processes it on the phone itself to identify the IVR or the phone menu system and then process particular parts of the interaction with the system to extract any credit card numbers and that may have been supplied during the interaction. On-phone processing drastically reduces the communication cost as only a credit card number is sent to the malware master rather than the complete recording of the call. The application itself may not need to have network permissions but may use covert channels to send the data to a cooperating deliverer app which can then transmit the highly sensitive information to the outside world. Clearly, Soundcomber takes all steps to defeat any existing approaches. The authors also propose a defense architecture against such Trojans. Almost everything described in the paper is obvious or well known though I still consider the work (proposing the new attack and its defense) as a whole to be a quite good contribution. I have some detailed comments about which I give below.
Positive Review

In my opinion, a defense architecture should be general. It seems that it would work well only for the microphone sensor. I believe there could be sophisticated ways of using the camera or the accelerometer to determine what keys the user may have pressed. However, it is quite possible that these sensors are being used for some other application as well, unlike the microphone - no one would use a microphone, except for some recording purposes, during a call. Moreover, if the user uses VoIP to make the call, it is very difficult to implement something of the semblance of a reference monitor during sensitive calls. Although this architecture may work well for some sensory data, there are other kinds of private information - like a user's contacts or even the GPS sensor - which need to be protected. They may perhaps not be as critical but similar covert channel attacks apply to them as well.
I also think that it is easy to close most of the covert channels that the authors describe. They are perhaps kind of an overlook in the Android design but may easily be addressed. Moreover, covert channels, although having been talked of as a way of transmitting information in the paper, may easily be traced using dynamic tainting. A recent paper does that: