CURSORY SECURITY EVALUATION OF THE TESLA MODEL S
WE CAN'T PROTECT CARS LIKE WE PROTECT OUR WORKSTATIONS

by NITESH DHANJANI

2014

The innovation behind Tesla's electric vehicles has set us in the right direction towards a more sustainable future. What Elon Musk[1] is doing with Tesla and SpaceX[2] is inspirational and a triumph for humankind.

Given the fantastic future of IoT (Internet of Things) devices ahead of us, it is the responsibility of the security community and device manufacturers to do our best to enable these devices securely. The IoT devices in scope include remotely controllable thermostats, baby monitors, light bulbs, door locks, cars, and many more. The impact of security vulnerabilities targeting such devices can lead be physical in nature in addition to contributing to loss of privacy.

The purpose of this document is to outline the mechanisms by which the Tesla Model S communicates with car owners and the Tesla infrastructure using a variety of TCP/IP mechanisms. The goal of this document is to advise the owners on security issues they should be aware of as well as to kick off a dialogue between security researchers and Tesla Motors that will ultimately drive deeper analysis and assurance.

The Tesla Model S P85+

The Tesla Model S is currently configurable within the following different options:
- 60kWh battery: 208 miles range, 302 horse-poser
- 85kWh battery: 265 miles range, 362 horse-power
- P85Wh battery: 265 miles range, 416 horse-power Performance model (also configurable with Performance+ for better handling) [Figure 1]

Figure 1: The Tesla Model S P85+

The Tesla Model S is fully electric. In addition to charge stations available in most metro areas, they can also

[1] Elon Musk: http://en.wikipedia.org/wiki/Elon_Musk

[2] SpaceX: http://en.wikipedia.org/wiki/SpaceX

be charged for free (for life) at any of the Tesla Super-charging stations[3].

Figure 2: The center display.

The center display depicted in Figure 2 is one of the popular features of the car. The display not only lets you control media, access navigation, turn on the rear view camera, but it also lets you adjust the suspension, open the panoramic roof, lock and unlock doors, and adjust the hight and breaking of the vehicle. This is all done via the touch screen.

The Tesla Model S is a truly innovative product. In the next section, we will take a look at some preliminary security issues that may be helpful to owners as well as to other security researchers to assist with deeper level analysis.

Threats

In this section, we will discuss potential security issues.

1. Six character password can lead to car being located and unlocked via Malware, Phishing, and Password Leaks.

To order a Tesla Model S, you have to register for an account on http://www.teslamotors.com.

[3] Tesla Super-chargers: `http://www.teslamotors.com/supercharger`

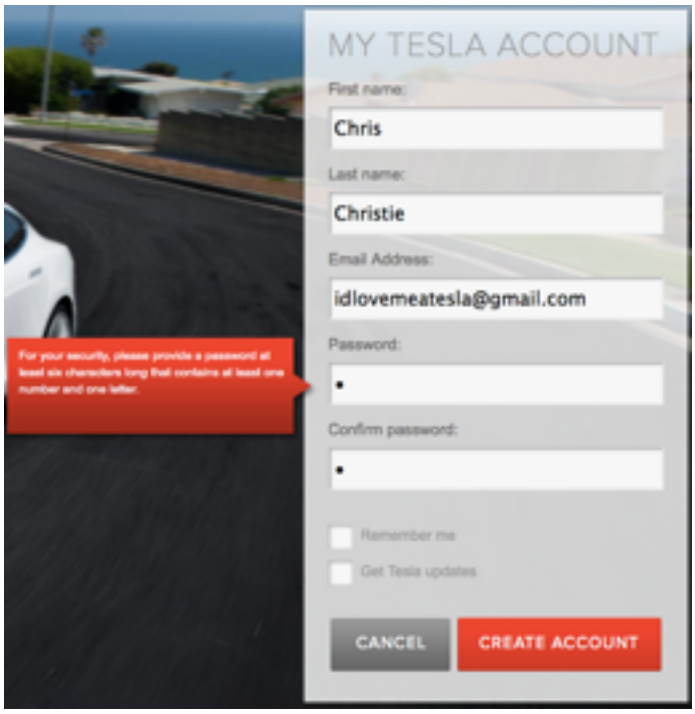

Figure 3: Tesla's password requirements

The password requirement for a new user account is 6 characters with at least one number and one letter (Figure 3).

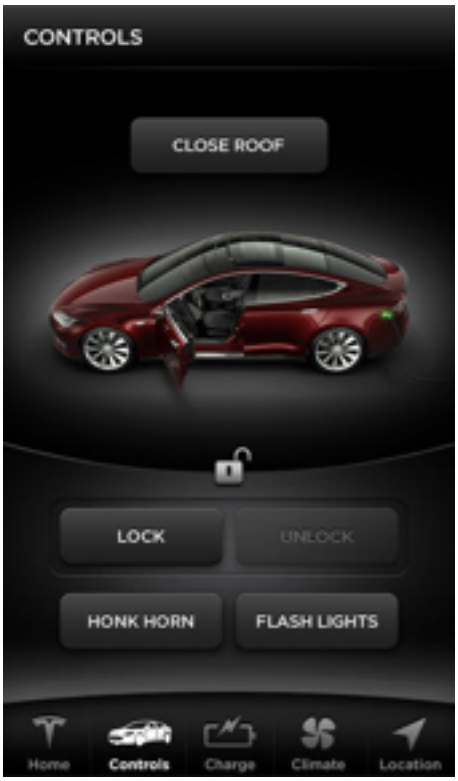

Figure 4: Tesla Model S iPhone App

Once the car is delivered, the user can use the iOS app[4] to control the car, inclusive of unlocking the car, checking on the car's location and charge status (Figure 4).

The following are the implications as a result of this design:

1. Brute-force attacks: The Tesla website doesn't seem to have any particular account lockout policy per incorrect login attempts. This puts owners at risk since a malicious entity can attempt to brute-force the account and gain access to the iPhone functionality.

2. Phishing attacks: Given that the only control around the iPhone app is a password, the situation is ripe for potential attackers to steal credentials using phishing attacks. Once credentials are gathered, phishers can easily check the location of the cars for the accounts they have compromised by using the Tesla REST API[5] (destination `https://portal.vn.teslamotors.com/`) by following these steps:

    A. Login by submitting to `/login` and setting the `user_session[email]` and `user_session[password]` parameters.
    B. Use the session token from A. to obtain the vehicle list by submitting a `GET` request to `/vehicles`.
    C. User the vehicle `id` obtained in B. to query the location of the vehicle by submitting a GET request to `/vehicles/{id}/command/drive_state`. This will return the location in the form of `latitude` and `longitude`.

    Once the phisher has obtained the location of the vehicles mapped to the compromised accounts he or she can unlock a particular vehicle or a set of vehicles (buy invoking the following in a loop): `GET` request to `/vehicles/{id}/command/door_unlock`.

3. Malware attacks: Future generation malware is likely to pick up static 1-factor passwords pertaining to vehicles such as the Tesla and ferry them to botnet herders giving them substantial power into locating and controlling (unlocking the car, for example) vehicles.

4. Password leaks: Users have a tendency to re-use their credentials on other services as well. This creates a situation where an attacker that has compromised a major website can attempt to try the same password credentials on Tesla website and iPhone app. We also see situations of major password leaks[6] on a daily basis. An attacker can easily use usernames and passwords from such leaks and attempt login on the Tesla iOS app (or automate the process described in 2. using the REST API) to locate and unlock cars.

5. Social engineering and Tesla employees: In addition to these issues, it is widely known amongst Tesla owners that Tesla customer service has the ability to unlock cars remotely[7]. It is unclear what consistent requirements owners have to go through to verify their identity. Without clear requirements, it is possible that a malicious entity may be successful in social engineering Tesla customer service to unlock someone else's car. It is also unclear what background checks Tesla employees are subject to prior to be given the power to unlock any Tesla car.

---

[4] Tesla Model S iPhone app: `https://itunes.apple.com/us/app/tesla-model-s/id582007913`

[5] Tesla REST API: `http://docs.timdorr.apiary.io/`

[6] Example: `https://twitter.com/PastebinLeaks`

[7] "Locked Out - Let In": `http://www.teslamotors.com/it_IT/forum/forums/locked-out-let`

6. Email account compromise: Any user with temporary access to the owner's email can reset the owner's password. The user will not be required to answer any secret questions or any additional information. For an expensive car such as the Tesla Model S and the physical consequences of theft of material inside the car, it is recommended that owners protect their email accounts by:

   A. Setting up a separate GMail account that is not tied to any other service and enable 2 factor auth[8].
   B. Link this GMail email address to their Tesla profile.

   On a somewhat positive note, it was noted that the Tesla website incorporates an anti-CSRF[9] token (`form_token`) which prevents malicious website from taking over the user's account by invoking a POST request to the `/user/me/edit` functionality which lets users change their password and username.

2. Tesla REST API Implicitly Encourages Credential Sharing with Untrusted Third Parties.

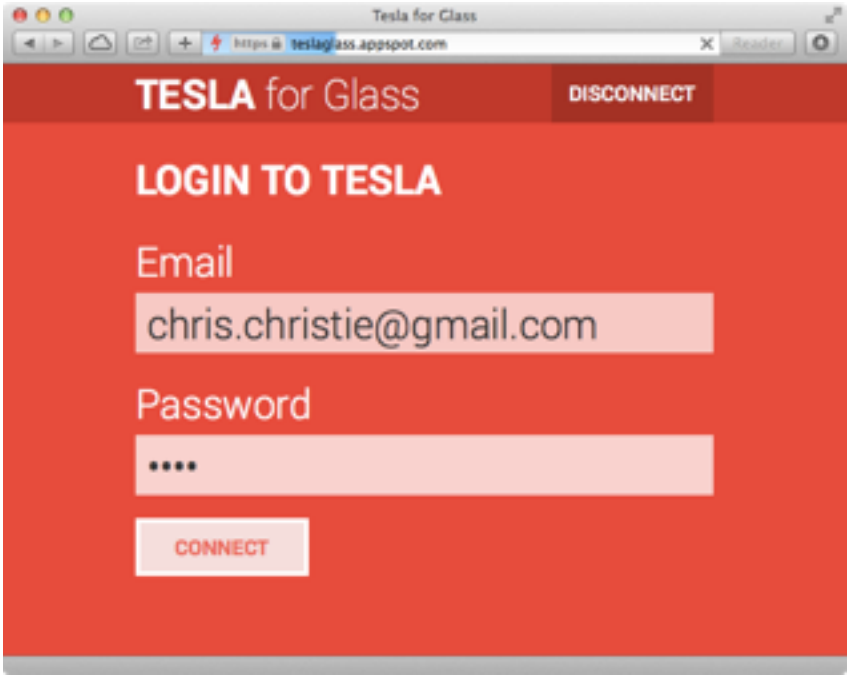

Figure 5: Tesla for Glass login page

The Tesla iOS App uses a REST API to communicate and send commands to the car. Tesla has not intended for this API to be directly invoked by 3rd parties. However, 3rd party apps have already started to leverage the Tesla REST API to build applications.

[8] `https://www.google.com/landing/2step/`

[9] `http://en.wikipedia.org/wiki/Cross-site_request_forgery`

As an example, The Tesla for Glass[10] application lets users monitor and control their Teslas using Google Glass.

In order to setup this application, Google glass owners have to authorize and add the app. Once this step is complete, the user is redirected to a login page as shown in Figure 5. On this page, the user enters their http://www.teslamotors.com/ login information.

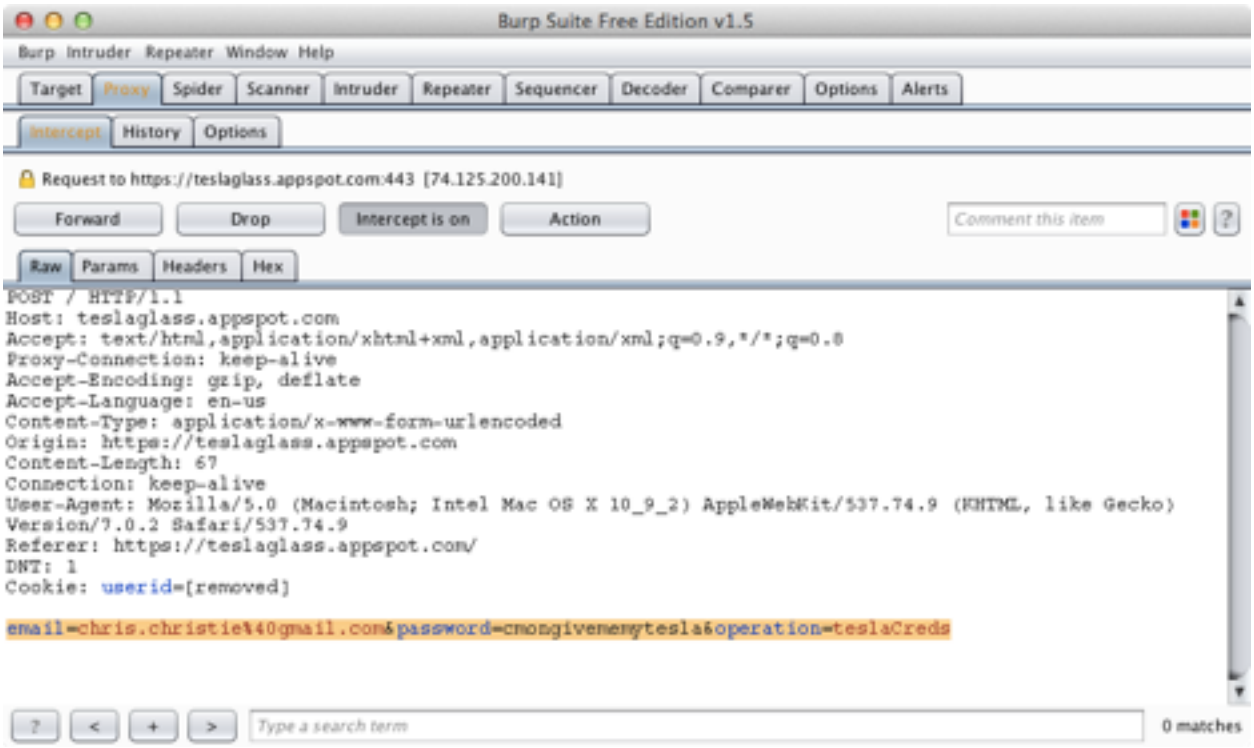

Figure 6: Tesla website credentials are collected by third party app

As shown in Figure 6, user credentials are sent to `teslaglass.appspot.com`. Therefore, in this case, it can be assumed that the 3rd party is using the user credentials to invoke the Tesla REST API on behalf of the user. This leads to the following risk for the owners:

1. Malicious 3rd party applications: The application owners can abuse this situation to collect credentials of Tesla accounts.

2. Insecure design of 3rd party infrastructure: Should the 3rd party infrastructure be compromised, the malicious intruder can collect Tesla users' credentials and abuse the remote functionality.

[10] Tesla for Glass: `http://glasstesla.com/`. Another example is `https://smartcar.io/`.

This issue has been raised in the community by George Reese[11]. Elon Musk has confirmed[12] that Tesla has plans to eventually release an SDK for 3rd party developers. It is likely that the Tesla sponsored solution includes an SDK, access to a remote API, local sandbox, OAUTH like authorization functionality, and a vetting process that draws inspiration from the Apple App Store.

In the meanwhile, Tesla owners are strongly encouraged not to use third party applications.

3. Temporary physical possession of iPhone can be abused to locate and unlock car for a period of time.

The Tesla iOS app stores a session token obtained from successful authentication with the REST API in the `Library/Cookies/` directory within the App in the file called `Cookies.binarycookies`. Anyone with temporary access to your phone can steal the content of this file to make direct requests to control the REST API functionality. This cookie has been documented to be valid for 3 months at a time.

Owners are advised to set a strong device password on their iPhone and refrain from sharing their phone with others.

The probability of this issue is low because it requires physical access to the phone. Note, however, that unlike temporary access to a physical key (the role of which is played by the phone), the potential malicious entity will have prolonged access to the functionality even after returning the phone.

[11] `http://programming.oreilly.com/2013/08/tesla-model-s-rest-api-authentication-flaws.html`

[12] `http://transportevolved.com/2014/02/06/elon-musk-amsterdam-town-hall-meeting-short-and-medium-term-future/`

Potential Low Hanging Fruit

The Tesla connects outbound via 3G and can also hop onto a local Wi-Fi.

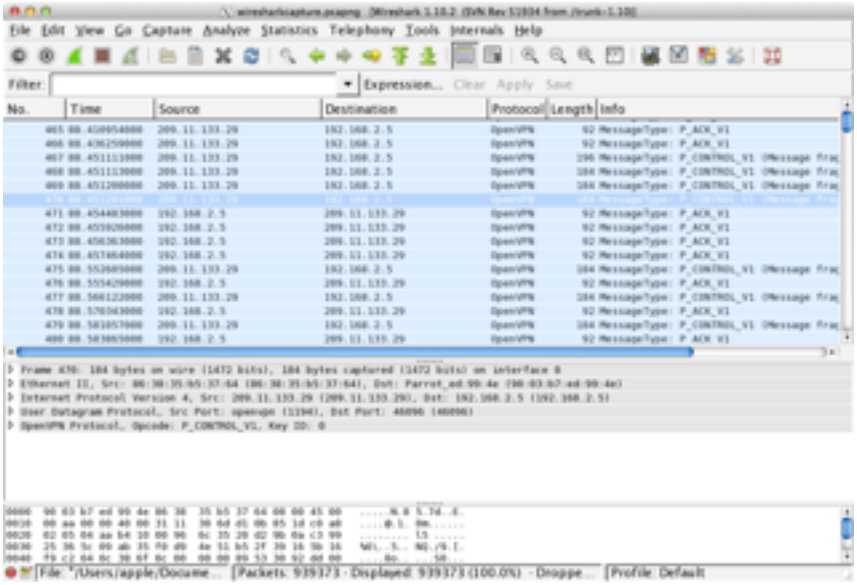

Figure 7: Capture of outbound connection from Tesla Model S on Wi-Fi

When the Model S is configured to use Wi-Fi, it was noted that it established an outbound connection to `209.11.133.29` using the OpenVPN protocol. It was also noted that a `HEAD` request was issued to `23.209.17.60` which resulted in a `400 bad-request` response.

Figure 8: Forum discussion about Tesla Model S' internal network

The Model S also has a 4-pin connector on the left of the dashboard. A M12 to RJ45 adapter[13] can be used to connect a laptop to this port. Users on the teslamotorsclub.com forum have reported various information about the internal network after having plugged into it[14]:

- Potential center console with IP address of 192.168.90.100 and the following services open:

```
22/tcp open ssh
53/tcp open domain
80/tcp open http
111/tcp open rpcbind
2049/tcp open nfs
6000/tcp open X11
MAC Address: FA:9E:70:EA:xx:xx (Unknown)
```

- Dashboard screen with IP address of 192.168.90.101 and the following services open:

```
22/tcp open ssh
111/tcp open rpcbind
6000/tcp open X11
MAC Address: 36:C4:1F:2A:xx:xx (Unknown)
```

- Another device with IP address of 192.168.190.102 with the following services open:

```
23/tcp open telnet
1050/tcp open java-or-OTGfileshare
MAC Address: 00:00:A7:01:xx:xx (Network Computing Devices)
```

- The SSH service on 192.168.90.100 has the banner of `SSH-2.0-OpenSSH_5.5p1 Debian-4ubuntu4`

- The DNS on 192.168.90.100 is of version `dnsmasq-2.58`

- The HTTP server on 192.1168.90.100 appears to expose /nowplaying.png which is the album art displayed on the dashboard.

- The NFS service on 192168.90.100[15] exposes /opt/navigon which contains the following structure:

```
dr-xr-xr-x 5 1111 1111 4096 Mar 21 2013 .
drwxrwxrwt 20 root root 20480 Mar 18 17:01 ..
dr-xr-xr-x 4 1111 1111 4096 Mar 21 2013 EU (Contains /maps and /data)
dr-xr-xr-x 2 1111 1111 4096 Mar 21 2013 lost+found
-r--r--r-- 1 1111 1111 7244 Mar 21 2013 MD5SUM-ALL
dr-xr-xr-x 2 1111 1111 4096 Mar 21 2013 sound
-r--r--r-- 1 1111 1111 150 Mar 21 2013 VERSION
```

Contents of the VERSION directory:

```
UI/rebase/5.0-to-master-238-g734c31d7,EU
NTQ312_EU,14.9.1_RC1_sound.tgz
build/upgrade/mknav-EU-ext3.sh
```

---

[13] http://www.l-com.com/ethernet-m12-4-position-d-coded-male-rj45-male-cable-assembly-20m

[14] http://www.teslamotorsclub.com/showthread.php/28185-Successful-connection-on-the-Model-S-internal-Ethernet-network

[15] http://www.teslamotorsclub.com/showthread.php/28185-Successful-connection-on-the-Model-S-internal-Ethernet-network/page9?p=608435&viewfull=1#post608435

- Majority of the data noticed by plugging into the connector appears to be broadcast UDP packets with car status information.

Here are the potential implications and low hanging fruit:

- The outgoing connection using OpenVPN can be configured using pre-shared keys, or username and password based, or using certificates. It will be interesting to see where in the internal filesystem this information is located. Once this information is obtained, a potential intruder count test the internal network infrastructure of the OpenVPN end-point and also the integrity of how software updates are performed.

- It is currently unclear if the UDP broadcast data can be abused to comply the car into settings that could be potentially dangerous and/or to over-ride safety precautions.

- The exposure of the raw internal network just by plugging in appears to be dangerous in the case where a malicious valet service may abuse temporary physical access.

Conclusions

Based on the issues outlined in this document, the following are the take-away points:

1. Tesla should address the issue of using static passwords with low complexity requirements.
2. Tesla owners should be aware of risks based on the current situation and take precautions outlined in this document.
3. Until Tesla announces an SDK and methods they are going to outline to sandbox applications, users should refrain from using third party applications.
4. The forum discussion referred to in Footnotes 14 and 15 is fascinating. It is clear that Tesla owners want to engage in an open dialogue where they are assured by Tesla what security architectures are being utilized to secure the cars. This is analogous to how Apple described how the iMessage infrastructure is secured[16] to put personal and corporate users at ease.

The Tesla Model S is a great car and a fantastic product of innovation. Owners of Tesla as well as other cars are increasingly relying on information security to protect the physical safety of their loved ones and their belongings. Given the serious nature of this topic, we know we can't attempt to secure our vehicles the way we have attempted to secure our workstations at home in the past by relying on static passwords and trusted networks. The implications to physical security and privacy in this context have raised stakes to the next level.

Tesla has demonstrated innovation leaps and beyond other car manufacturers. It is hoped that this document will encourage owners to think deeply about doing their part as well as for Tesla to have an open dialogue with it's owners on what they are doing to take security seriously.

[16] http://images.apple.com/iphone/business/docs/iOS_Security_Feb14.pdf

About the Author

Nitesh Dhanjani is a well known security researcher, author, and speaker. Dhanjani is the author of “Hacking: The Next Generation” (O’Reilly), "Network Security Tools: Writing, Hacking, and Modifying Security Tools" (O'Reilly) and "HackNotes: Linux and Unix Security" (Osborne McGraw-Hill). He is also a contributing author to "Hacking Exposed 4" (Osborne McGraw-Hill) and "HackNotes: Network Security". Dhanjani has been invited to talk at various information security events such as the Black Hat Briefings, RSA, Hack in the Box, Microsoft Blue Hat, and OSCON.

Dhanjani is currently an Executive Director at a large consulting firm where he advises some of the largest corporations around the world on how to establish enterprise wide information security programs and solutions. Dhanjani is also responsible for evangelizing brand new technology service lines around emerging technologies and trends such as smart devices, cloud computing, and mobile security.

Prior to his current job, Dhanjani was Senior Director of Application Security and Assessments at a major credit bureau where he spearheaded brand new security efforts into enhancing the enterprise SDLC, created a process for performing source code security reviews & Threat Modeling, and managed the Attack & Penetration team.

Dhanjani graduated from Purdue University with both a Bachelors and Masters degree in Computer Science.