# Malware Tolerance

Report 7

Michael Denzel <m.denzel@cs.bham.ac.uk>

April 12, 2017

**Supervisor:** Prof. Mark Ryan
**Thesis group:** Dr. Flavio Garcia,
Dr. David Parker (RSMG Member)

# CONTENTS

# List of Figures



# List of Abbreviations

**AES** Advanced Encryption Standard
**APT** Advanced Persistent Threat
**BD** Burmester-Desmedt Group Key Agreement Protocol
**CKD** Centralised Key Distribution
**DoS** Denial of Service
**GDH** Group Diffie-Hellman
**ICS** Industrial Control System
**IDS** Intrusion Detection System
**IP** Internet Protocol
**MITM** Man-in-the-Middle
**MPC** Multi-Party Computation
**mRSA** mediated RSA
**OS** Operating System
**RSA** Rivest-Shamir-Adleman Cryptosystem
**RTOS** Real-Time Operating System
**SGX** Intel Software Guard Extensions
**STR** Skinny Tree Key Agreement Protocol
**TCB** Trusted Computing Base
**TCP** Transmission Control Protocol
**TEE** Trusted Execution Environment
**TGDH** Tree-Based Group Diffie-Hellman
**TPM** Trusted Platform Module
**VANET** Vehicular Ad-Hoc Network
**VLAN** Virtual Local Area Network
**WANET** Wireless Ad-Hoc Network
**WSN** Wireless Sensor Network

# 1 Introduction

As recent attacks demonstrate ([10, 12]), it is challenging to give security guarantees about the honesty of a system. Our previous work focused on *malware tolerance* which we want to apply to networks now. For this, we adjusted our definition:

*We aim to distribute trust over several independent components in a way that an individual component infected with malware cannot gain access (spreading), deny access (DoS, excluding radio jamming), or control access (MITM) of devices on the network it did not control before. No part of the network is assumed invulnerable to attacks.*

In simple words, we want to ensure that most of the network still operates securely when parts of it are compromised and that the attack is contained and limited from spreading further.

# 2 Malware-Tolerant Mesh Networks

As a first step, we will now examine mesh networks as they are vulnerable to attacks of their own devices: black hole attacks, Denial of Service (DoS), routing table overflows, and impersonation. [7]

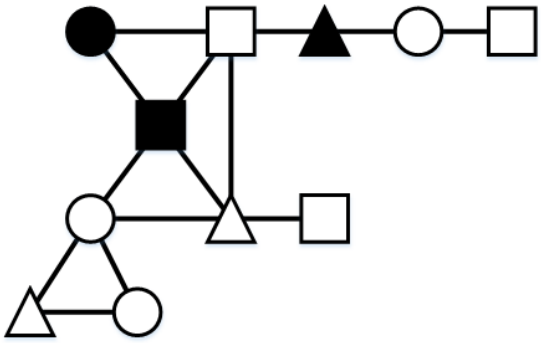
Mesh networks were used in former times in isolated settings, e.g. by rescue teams in remote areas, and thus the main focus was on safety and reliability. Nowadays, these networks are more and more applied in consumer scenarios like smart-home networks, Wireless Ad-Hoc Networks (WANETs) – especially Vehicular Ad-Hoc Networks (VANETs) – and Wireless Sensor Networks (WSNs). [7] As mesh networks are flat, compromised devices immediately supply the adversary with access to the entire network. This is in particular of concern when taking node capture attacks into account.

To improve upon this situation, we propose an isolation technique and an automatic containment mechanism based on voting for mesh networks.

## 2.1 Isolation

Let us consider the network shown in Fig. 1 where e.g. the triangles are a TV and two receivers, squares are audio speakers, and circles are smart light bulbs in a smart-home setting. The router is not in the living room and therefore out of reach. While we do not trust the smart light bulbs, we still want to watch a movie with sound from the speakers.

Figure 1: Example geographic network layout

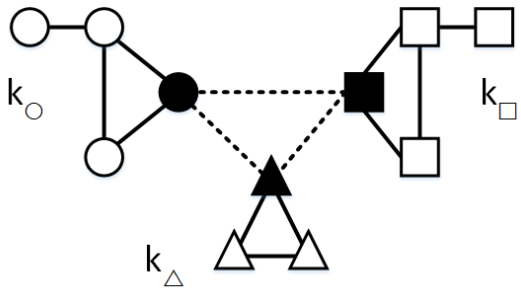


To counter attacks of the light bulbs we would isolate them from the other devices. Isolation is not new in traditional TCP/IP networks where subnetworks, Virtual Local Area Networks (VLANs), and firewalls are employed on a central point (e.g. the router). However, the central point represents a single point-of-failure and particularly in mesh networks such a central point might not exist.

We propose to add isolation in form of network groups to the mesh network creating a "virtual" view of the network (see Fig. 2) without single point-of-failure that does not rely on centralised firewalls or similar.

Figure 2: Example virtual network layout for Fig. 1

Virtual network with three groups: group circle, triangle, and square. "Admin" devices are black shapes while normal group members are white.




Our idea is to use cryptography and represent each group by a cryptographic key. To enable communication over the group boundaries, mes-

sages encrypted with one group key have to be re-encrypted. To do this, we propose a new group called "routing" group.

The routing group consists of at least two devices of each group (the "admin" devices), which are determined by the group itself. The admin devices have (together) all the group keys and can, thus, re-encrypt every message if one admin device of each group agrees. That is why we need two admin devices per group – to tolerate failures and attacks.

If e.g. AES in counter mode is used, the destination group could first encrypt the message before the source group decrypts it. Filtering is also possible, similar to a firewall, and works in a distributed fashion. In Fig. 1 and 2 the admin devices are shown as black shapes while normal group members are white. Re-encryption is possible if all keys are available which makes this scenario a problem of key distribution. [3]

## 2.2 Automatic Containment and Self-Management

To enable the framework to react to attacks, we allow devices to "veto vote" against devices. If a certain threshold of vetoes (minimum 2) is reached, the device is put into a separate isolated group or removed entirely from the network depending on if this is feasible.

Veto votes against normal group members is straight forward as the routing group devices have a higher privilege and can set the group of each device. Outvoting an admin device is also possible via the other admin devices. Each successful veto vote will trigger a re-setup of the key of the affected group.

As a result, the whole architecture has a variable setup with dynamically determined groups.

## 2.3 Definitions

To realise this architecture, seven operations are needed at first glance. We will introduce them now.

1. SETUP: Zones as well as initial keys have to be setup. This should happen as independently as possible but will likely require the presence of a user to define which devices belong to which groups.
2. JOIN: This event happens each time a device wants to join the network or a group. It should connect to the nearest mesh device and then request an authorisation from the routing group if not done yet – we call this authorisation (equivalent to the Kerberos protocol) *ticket*. Note that, this ticket must be acknowledged by at least two admin devices. Afterwards, the new device can contact the particular group, and receive or generate the key of that group.
3. LEAVE: Each time a device leaves a group or is forced to leave a group, the group key has to be re-established or has to evolve.
4. VOTE: When a group is introduced or updated, new admin devices might be necessary. These should be decided by the group members themselves via a voting protocol. Devices should be picked depending on their resources, geographic location, and trustworthiness. We imagine weaker or untrusted devices being able to opt-out (or begin forcefully excluded) of the election.
5. VETO: At any point in time, devices must be able to outvote any device of the network. This must be also malware-tolerant as a compromised device can possibly misuse this feature to take over the network.
6. ADMIN JOIN: After devices being promoted by a vote to an admin device, they have to join the routing group similar to the JOIN operation above.
7. ADMIN LEAVE: If a device leaves or forcefully leaves the routing group, all the key material of this group and possibly further groups has to be re-established.

## 2.4 First Draft Implementation

We suggest the following implementations for these operations. This is a first draft of an implementation and, therefore, does not take performance, battery life, cost etc. into account. Optimisation will be considered at a later stage.

### 2.4.1 SETUP

Every device is setup or generates a public private key pair $k_i^{pub}$ and $k_i^{priv}$ and is provided with the public key of the initial routing group $k_{admin}^{pub}$.

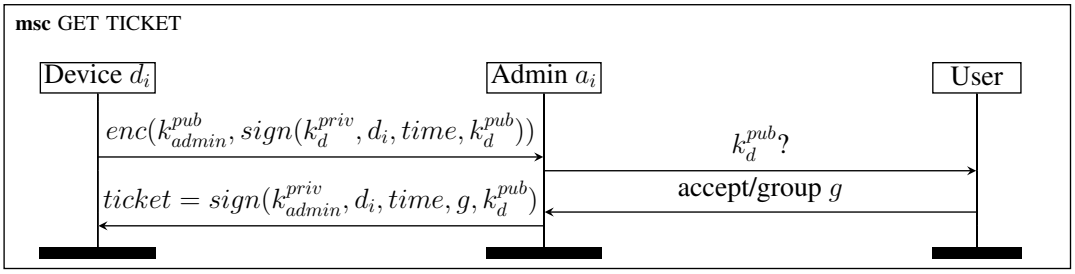



Figure 3: GET TICKET protocol

To setup the initial routing group, the user chooses two devices and pre-loads them with they necessary keys. So far, we consider two possibilities: threshold-based cryptography and intrusion-resilient signatures (see ADMIN JOIN below).

All devices then request a ticket from the routing group. We define this as the helper function GET TICKET (see Fig. 3).

### 2.4.2 JOIN

A JOIN is a combination of a GET TICKET operation (see Fig. 3) and a group key agreement protocol which can use the certified keys of GET TICKET. E.g. Group Diffie-Hellman (GDH), Tree-Based Group Diffie-Hellman (TGDH), Burmester-Desmedt Group Key Agreement Protocol (BD), and Skinny Tree Key Agreement Protocol (STR) are applicable[1]. [1]

We suggest to use BD as it is stateless or our own group key exchange which, however, is not yet proven.

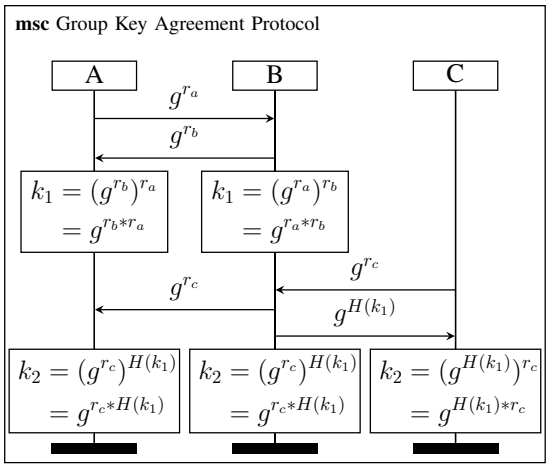



Figure 4: Group Key Agreement Protocol ($H(...)$ indicates a hash function)

**Idea for a group key exchange**

Our group key exchange (see Fig. 4) is incremental in the sense that we assume there is already a group and another device wants to join this group. We run a Diffie-Hellman key exchange with the random input parameter of the group being the current key. If there is no group yet, we run a normal Diffie-Hellman key exchange.

The resulting key for three parties is: $k_2 = g^{r_3 * H(k_1)} = g^{r_3 * H(g^{r_1 * r_2})}$.

One open issue to consider is that a malicious $C$ could sent multiple $g^{r_c}$ to desynchronise devices.

### 2.4.3 LEAVE

LEAVE is similar to JOIN. An admin device will initiate the protocol with the additional information who is leaving. This party should be excluded from participating in the next JOIN group key agreement.

### 2.4.4 VOTE

For now, we let every device cast a vote, signed with its private key, to the routing group. An admin device can verify the signature and record the vote. Devices which do not have the capabilities (e.g. battery), can opt-out by adding this to their vote. Voting is done based on routing tables or randomly.

E.g. if $device_1$ is the next hop for most connections in the routing table of $device_2$, it should vote for $device_1$. If no decision can be made, a device is picked at random. If after the vote there is a tie, the admins are picked at random. For later, we imagine there could be a list of the trustworthiness level for each device (e.g. based on packet drops, connection failures, etc.) which every device takes into

[1]Note that Centralised Key Distribution (CKD) relies on a trusted central system and can, therefore, not be used.

account.

#### 2.4.5 VETO

A veto is simply sent signed and encrypted to the routing group. If the misbehaving device is an admin device, the veto should be sent to another admin. When the vetoes reach a certain threshold (minimum 2), the misbehaving device is either put into an isolated group (LEAVE event previous group) or removed from the network (broadcast to all groups). This might trigger a VOTE event if the device was an admin.

#### 2.4.6 ADMIN JOIN

The ADMIN JOIN protocol is still under development. For the signatures of the routing group, we consider two approaches: a threshold-based cryptosystem and an intrusion-resilient encryption or signature scheme. A combination of both could provide the best results. As there is currently no such scheme to our knowledge, the scheme and how to distribute its keys are ongoing research.

We will now introduce our first thoughts:

**Threshold-based Cryptosystem**

For the signature scheme, we would like to enforce that at least two admin devices have to agree, before a certain key is certified in the GET TICKET protocol. However, it should be two devices from different groups. We thus considered mediated RSA (mRSA) [2] which splits keys into two (or more) parts: $k_{admin}^{priv} = k_{admin_1} + k_{admin_2}$.

For this, we would have to securely generate and distribute $k_{admin_1}$ and $k_{admin_2}$.

**Intrusion-Resilient Cryptosystem**

The idea of intrusion-resilient signature and encryption systems is that the private key keeps on evolving while the public key stays the same. [6] This is achieved by adding a time value into the cryptosystem which identifies the current private key to use. To be resilient against intrusions, the private key is split into a base and a secret key, both evolving. In order to evolve the next secret key, information of the base is needed. If an attacker compromised the secret key, he can read messages of the corresponding time period but does not get further keys. If he compromises the base, he gets nothing. The attacker would have to compromise both, the base and the secret key, at the same time in order to succeed.

We found two potentially applicable cryptosystems: the one of Dodis et al. [4] and of Itkis et al. [9, 8].

To not generate too many keys, we propose to have two groups of admins ($admin_A$ and $admin_B$) who both run an intrusion-resilient signature scheme. Only messages signed with both keys are considered trustworthy. If an admin leaves the group, the secrets must evolve. If an admin joins the group, he gets the appropriate secret.

The base could be managed by the other admin group (e.g. $admin_A$ has its own secret and the base of $admin_B$).

**Improvement**

Our goal is to combine these both techniques in a way so that both groups only have a share of the entire key (as with mRSA) which both evolve. That way, clients would not have to verify two signatures but only one.

Another crucial factor is the performance of the scheme. Verifications have to be fast, as they happen often while ADMIN JOIN and ADMIN LEAVE events should not occur during normal execution.

#### 2.4.7 ADMIN LEAVE

ADMIN LEAVE is like LEAVE but this time the ADMIN JOIN, JOIN, and VOTE protocol has to be rerun.

## 3 Other Networks

After having seen how mesh networks can be made malware-tolerant, we will now have a brief look at other infrastructures.

### 3.1 By Topology

- Bus: Since broadcasts are cheap in bus networks and there are no single point-of-

failures, our isolation overlay would even be faster. Admin devices could be at any point in the network.

- Star: Star networks have the disadvantage of a single point-of-failure in the middle. Separation in hardware through the central device might be cheaper than our approach but not malware-tolerant.

  Our architecture has to minimise the workload for the centre. Also, if a device is outvoted of the network, the network is partitioned, which should therefore be omitted.
- Daisy chain (linear, ring): In linear networks every device is a single point-of-failure, thus these networks should be avoided from a security perspective. Ring networks, on the other hand, always provide two paths from a device to another. Here, our architecture provides additional benefit and outvoting a device does not partition the network.
- Hybrid: For hybrid networks, it depends on the specific layout if our approach is useful. As those networks are closer to mesh networks, our architecture is applicable but certain devices at intersection points should not be removed from the network.

### 3.2 By Network Types

- Home networks: These networks are usually star networks with the before mentioned points. However, newer developments like Google Wifi[2] suggest that these network move to different topologies – especially mesh networks – for better connection.
- Business networks: Companies usually split their networks already into (hardware-based) zones. As we expect the outvoting of devices to be rather slow, the cost-overhead of our architecture is too high for these networks.
- Embedded networks: Networks like the CAN bus are a bus network and hence suited for our architecture.
- Ad-hoc networks: Ad-hoc networks rely on safety and try to compensate for bad connections. To achieve this, they often employ mesh routing.
- Smart-Home networks: Smart-homes move towards mesh networks for smart-meters, home entertainment, and so on.
- Internet: Even though creating a model of the infrastructure of the internet is difficult, it is a hybrid network. Despite this, our architecture will not fit, since some countries have a much higher number of devices than others. As a result, choosing admin devices would be biased and will likely not work efficiently.
- Overlays: Virtual overlays like peer-to-peer networks are possible in combination with our architecture but impose an additional overhead. Inside of the groups, some overlays might make sense but we doubt this will be practical.

We estimate that our architecture is beneficial for highly interconnected, flat networks where the geographic setup does not represent the actual working groups (that means with a lot of Man-in-the-Middle (MITM) devices). It should also enhance networks where the geographic setup matches the virtual one.

## 4 Problems

We will now discuss a few issues briefly:

1. Evolution of $k_{admin}^{priv}$: The distribution and evolution of the private key of the routing group is still ongoing research. We are focusing on intrusion-resilient signatures as this is a promising approach.
2. Stealthy attacks: Our architecture will not automatically defend stealthy attacks. However, an adversary is restricted to his own group; compromised admin devices would give access to two groups but even they cannot interfere with other groups. We estimate that the system will converge to a trusted state in the long run, because as soon as the attack becomes visible the device will be automatically isolated. The infrastructure becomes a moving target for the adversary which is much harder to compromise.
3. DoS: Attacks targeting the cryptographic routines are unavoidable but we think that they are easily identified and could even lead to devices being entirely removed from the network (in contrast to being isolated).

[2] https://madeby.google.com/wifi/

4. Early attacks: The system is more vulnerable during setup. That is why we specified that the user has to setup two devices. This gives us two advantages. First, the attacker has to compromise two systems. Second, the user chooses the devices and the attacker does not necessarily know which devices these are.
5. Performance: For now we ignored the performance overhead the architecture places upon the system. Improvements are faster cryptographic routines but cryptography is definitely necessary. Normal execution relies on symmetric cryptography and is thus reasonably fast. The special events JOIN and LEAVE are expensive but only happen during setup, failures, and attacks. They should occur seldom.

All in all, our architecture can provide isolation for flat networks and enables them to automatically contain compromised devices. It is better suited for interconnected networks but can improve other types of networks, too.

## References

[1] Yair Amir, Yongdae Kim, Cristina Nita-Rotaru, and Gene Tsudik. On the performance of group key agreement protocols. *ACM Transactions on Information and System Security (TISSEC)*, 7(3):457–488, 2004. URL http://dl.acm.org/citation.cfm?id=1015045. accessed: 2017-04-04.

[2] Dan Boneh, Xuhua Ding, Gene Tsudik, and Chi-Ming Wong. A method for fast revocation of public key certificates and security capabilities. In *USENIX Security Symposium*, 2001. URL http://static.usenix.org/publications/library/proceedings/sec01/full_papers/boneh/boneh.ps. accessed: 2015-02-17.

[3] Seyit A Camtepe and Bülent Yener. Key distribution mechanisms for wireless sensor networks: a survey. *Rensselaer Polytechnic Institute, Troy, New York, Technical Report*, pages 5–7, 2005. URL http://www.cs.rpi.edu/research/pdf/05-07.pdf. accessed: 2017-02-24.

[4] Yevgeniy Dodis, Matt Franklin, Jonathan Katz, Atsuko Miyaji, and Moti Yung. A generic construction for intrusion-resilient public-key encryption. In *Cryptographers' Track at the RSA Conference*, pages 81–98. Springer, 2004. URL http://link.springer.com/chapter/10.1007/978-3-540-24660-2_7. accessed: 2017-04-08.

[5] David Drummond. A new approach to china. online (official Google blog), January 2010. URL https://googleblog.blogspot.co.uk/2010/01/new-approach-to-china.html. accessed: 2016-07-27.

[6] Matt Franklin. A survey of key evolving cryptosystems. *International Journal of Security and Networks*, 1(1-2):46–53, 2006. URL http://web.cs.ucdavis.edu/~franklin/ecs228/pubs/extra_pubs/key_evolve_survey.pdf. accessed: 2017-04-08.

[7] Yih-Chun Hu and Adrian Perrig. A survey of secure wireless ad hoc routing. *IEEE Security & Privacy*, 2(3):28–39, 2004. URL http://robotics.eecs.berkeley.edu/~sastry/pubs/PDFs%20of%20Pubs2000-2005/Pdfs%20of%20Misc.Others/Hu,Jih%20Chun/YihChunHU.ISnP.pdf. accessed: 2016-10-18.

[8] Gene Itkis. Intrusion-resilient signatures: generic constructions, or defeating strong adversary with minimal assumptions. In *International Conference on Security in Communication Networks* Itkis and Reyzin [9], pages 102–118. URL http://link.springer.com/chapter/10.1007/3-540-36413-7_8. Accessed: 2017-04-10.

[9] Gene Itkis and Leonid Reyzin. Sibir: Signer-base intrusion-resilient signatures. *Advances in Cryptology–Crypto 2002*, pages 101–116, 2002. URL http://www.springerlink.com/index/JD2TTH28DR6J6YY3.pdf. Accessed: 2017-04-10.

[10] Ralph Langner. Stuxnet: Dissecting a cyberwarfare weapon. *Security & Privacy, IEEE*, 9:49–51, Nov 2011. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5772960. accessed: 2014-08-06.

[11] Craig Schmugar. More details on "operation aurora". online (McAfee), January 2010. URL https://blogs.mcafee.com/mcafee-labs/more-details-on-operation-aurora/. accessed: 2016-07-27.

[12] Nikos Virvilis, Dimitris Gritzalis, and Theodoros Apostolopoulos. Trusted computing vs. advanced persistent threats: Can a defender win this game? In *10th International Conference on Autonomic and Trusted Computing (ATC)*, pages 396–403. IEEE, 2013. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6726235. accessed: 2015-02-13.

# Appendices

## A Dissertation Outline

(Changes to the previous outline highlighted in blue)

1. Introduction
   - 1.1. Motivation: Rootkits, Advanced Persistent Threat (APT) attacks, Shodan vulnerability scan, Industrial Control System (ICS) attacks (Stuxnet [10], Operation Aurora [5, 11] etc.)
   - 1.2. Previous research: Mitigating attacks, Intrusion Detection System (IDS), detection, defending against a few attacks (signature/anomaly/specification based)
   - 1.3. Contributions
2. Literature review
   - 2.1. APT attacks
   - 2.2. Software vulnerabilities
   - 2.3. Intrusion detection/tolerance/prevention
   - 2.4. Sandboxing
   - 2.5. Trusted Execution Environments
     - 2.5.1. Hardware (new architectures, additional devices)
     - 2.5.2. Software (compiler)
     - 2.5.3. Virtualisation and hypervisor
     - 2.5.4. Integrated hardware (Intel Intel Software Guard Extensions (SGX), TrustZone/Knox, Trusted Platform Module (TPM))
   - 2.6. Multi-Party Computation (MPC)
   - 2.7. Self-healing systems
   - 2.8. Networks: smart-home, mesh network, WSN, WANET
3. The malware tolerance concept
   - 3.1. Definition
   - 3.2. Scenarios
     - 3.2.1. E-voting
     - 3.2.2. Online banking
     - 3.2.3. ICS
     - 3.2.4. Smart-homes
     - 3.2.5. (Cloud)
     - 3.2.6. (Bitcoin)
     - 3.2.7. (Anonymity, TOR)
   - 3.3. Assumptions and attacker model
   - 3.4. Multiple Trusted Computing Base (TCB) model
4. ARM TrustZone
   - 4.1. In-depth Presentation
   - 4.2. Security Analysis
   - 4.3. Performance Analysis TrustZone
5. User input: Smart-Guard
   - 5.1. Introduction/problem

# B Timetable

Table 1: Milestones

| | |
|---|---|
| 2014-03-14 | Start |
| 2014-05-19 | **Report 1** |
| 2014-06 | Literature review |
| 2014-09-20 | **Report 2** |
| 2014-10 | Explored scenario “storage” under two assumptions |
| 2014-12 | Idea: double encryption |
| 2015-01 | Researched formal representation of distributed systems, Idea: Trusted input (“sign-what-you-type”) |
| 2015-02/-03 | **Report 3:** Thesis proposal |
| 2015-05 | Expanded trusted input idea, verification with ProVerif |
| 2015-06 | **Paper 1:** Smart-Guard |
| 2015-07 | Presentation at CryptoForma Workshop (CSF 2015) |
| 2015-08 | Researched TrustZone (papers, OS, *FriendlyARM* board) |
| 2015-09-20 | **Report 4** |
| 2015-10 | Researched Multi-Party Computation/Industrial Control Systems |
| 2015-12 | Reworked Smart-Guard paper |
| 2016-01 | Examined TrustZone of *FriendlyARM Mini6410* |
| 2016-02 | Researched self-healing systems; adjusted *FreeRTOS* to run on *FriendlyARM* board |
| 2016-03/-04 | Improved scenario of Smart-Guard; adjusted *FreeRTOS* to run on *FreeScale i.MX53* |
| 2016-04-03 | **Report 5** |
| 2016-07 | Smart-Guard accepted and presented at ATC 2016 (Best paper award) |
| 2016-08 | **Paper 2:** A malware-tolerant, self-healing ICS Framework |
| 2016-09 | Researched Smart-Homes, Timing-Analysis of TrustZone |
| 2016-10-02 | **Report 6** |
| 2016-12 | Submitted self-healing ICS paper to IFIP SEC 2017 |
| 2017-01 | Researched Mesh Networks |
| 2017-02 | Self-healing ICS paper accepted at IFIP SEC 2017 |
| 2017-03-13 | Start of fourth year |
| 2017-04-09 | **Report 7** |
| 2017-05 | Finishing paper 3 |
| 2017-06 | **Paper 3:** Malware-Tolerant Networks |
| 2017-06 | Focusing entirely on Ph.D. thesis |
| 2017-10-08 | **Report 8** |
| 2018-03-13 | Submission deadline |

# C Attended Workshops

Table 2: Attended Workshops

| Date | Workshop/Conference | Location |
|---|---|---|
| 7th May 2014 | Google Hack Day (Certificate Transparency) | Google London |
| 19th-23rd May 2014 | Academic Writing Seminar | University of Birmingham |
| 27th-28th May 2014 | CryptoForma 2014 | University of York |
| 14th-18th Jul 2014 | Enterprise Summer School | University of Birmingham |
| 23rd-24th Jul 2014 | Publishing Academic Journals, Editing your writing | University of Birmingham |
| 22nd Oct 2014 | Time Management | University of Birmingham |
| Oct 2014 - Jan 2015 | Research Skills Seminar | University of Birmingham |
| 18th Nov 2014 | Speed Reading | University of Birmingham |
| 24th Nov 2014 | Note Taking | University of Birmingham |
| 2014 | Cryptography 1 | University of Stanford (Coursera) |
| 2014 - 2015 | Writing in the Sciences | University of Stanford (Coursera) |
| 14th-15th Jan 2015 | CryptoForma 2015 | University of Kent |
| 13th July 2015 | CryptoForma Workshop at CSF 2015 | University of Verona |
| 31th Aug-5th Sept 2015 | FOSAD Summer School 2015 | University Residential Centre of Bertinoro |
| 29th Jan 2016 | RITICS meeting | Imperial College London |
| 11th Jul-12th Jul 2016 | ACE-CSR 2016 | Solihull, Birmingham |
| 18th Jul-21st Jul 2016 | **ATC 2016 (Best paper award)** | University Paul Sabatier of Toulouse |
| 23th Aug-26th Aug 2016 | ICS-CSR 2016 | Queens University Belfast |
| 13th Mar-14th Mar 2017 | Symposium on Innovative Smart Grid Cybersecurity Solutions | Austrian Institute of Technology |
| 30th Mar 2017 | Symposium to explore potential applications for new sensor technologies in the marine sector | University of Birmingham |