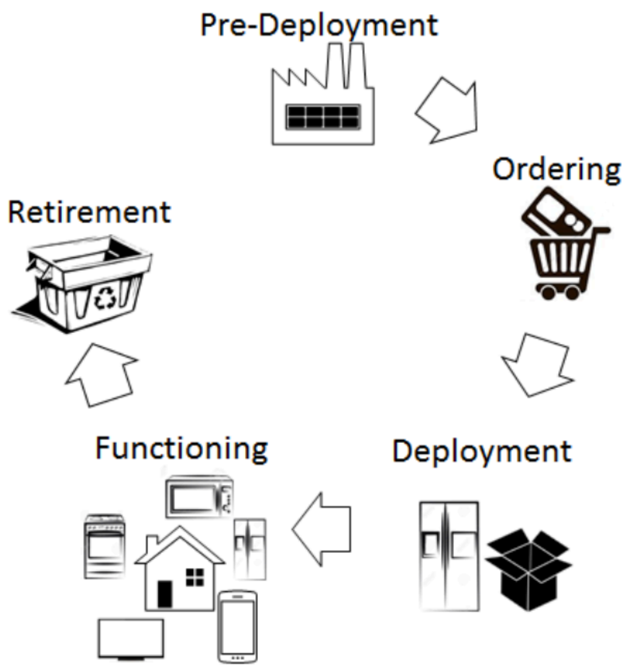
Department of Computer Science
Universidade Federal de Minas Gerais
Federal University of Minas Gerais, Brazil

AoT: Authentication and Access Control for the Entire IoT Device Life-Cycle

A. Neto, A. Souza, Í. Cunha, M. Nogueira, I. Nunes, L. Cotta, A. Loureiro, N. Gentille, D. Aranha, H. Patil, Leonardo B. Oliveira

IoT device life-cycle

- The IoT device life-cycle comprises a number of stages
  - Pre-deployment
  - Ordering
  - Deployment
  - Functioning
  - Retirement



Authentication is a must over the entire lifecycle

Problem

- Existing authentication proposals do not meet IoT needs completely

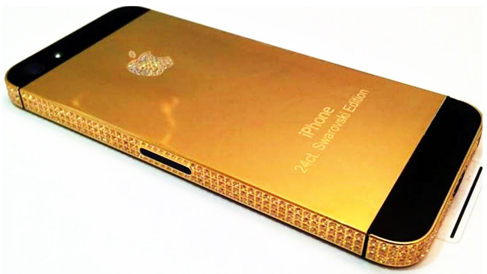
Problem

- Existing authentication proposals do not meet IoT needs completely

#1 Deployment schemes lack security

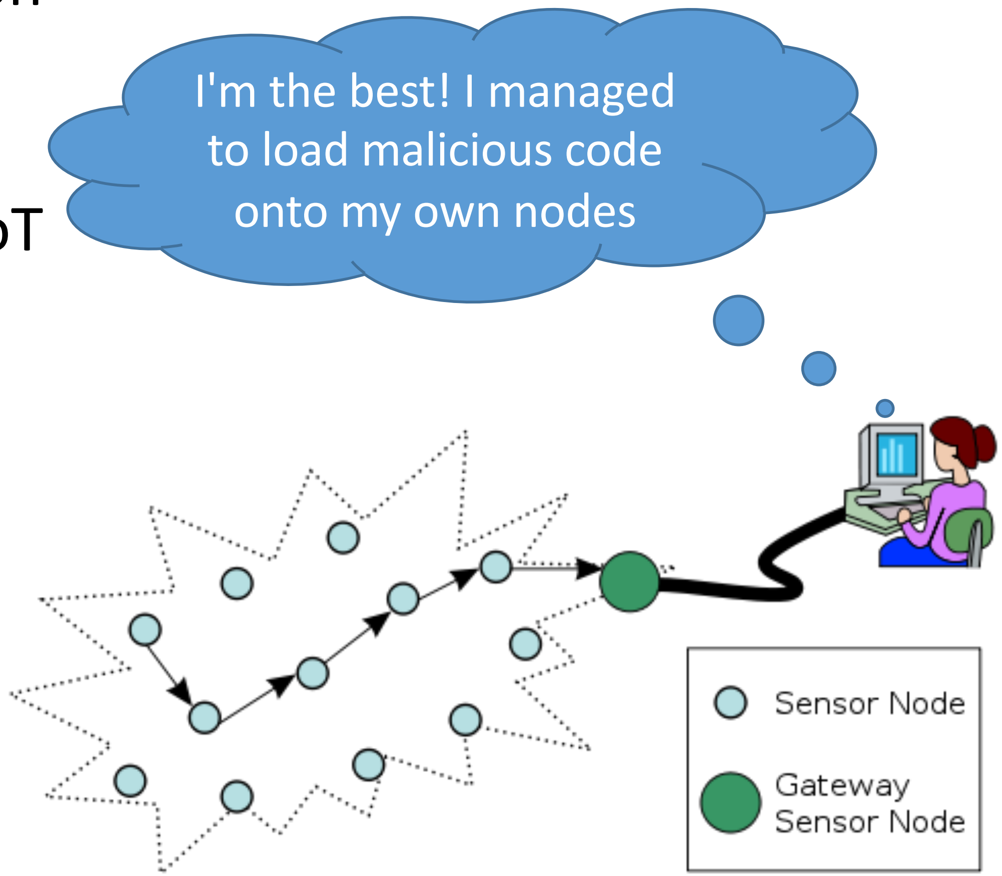
#2 Traditional auth schemes are expensive

- Traditional schemes based on PKI require public-key authentication
  - Process of ensuring that a public-key that supposedly belongs to Bob does in fact belong to Bob
- And public-key authentication is expensive because it is achieved using certificates
  - Certificate exchange implies communication overhead
  - Certificate verification implies computation overhead
  - Certificate storage implies memory overhead

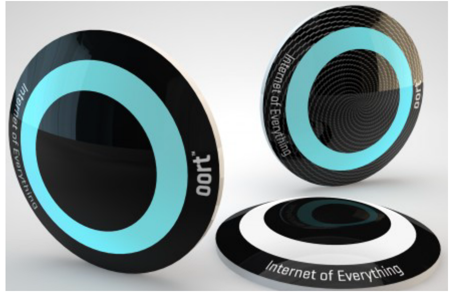


#3 Sensors are used by a single owner/domain

- Existing proposals rely on that to secure WSNs
- Those assumptions are false in the context of IoT



Our goal



- Design, develop, and evaluate a tailor-made authentication and access control solution for IoT
  - We target devices like home appliances that interact with other home appliances and personal devices
- The solution must
  1. Cover the entire IoT device life-cycle
  2. Meet the efficiency and security needs of IoT apps

Approach

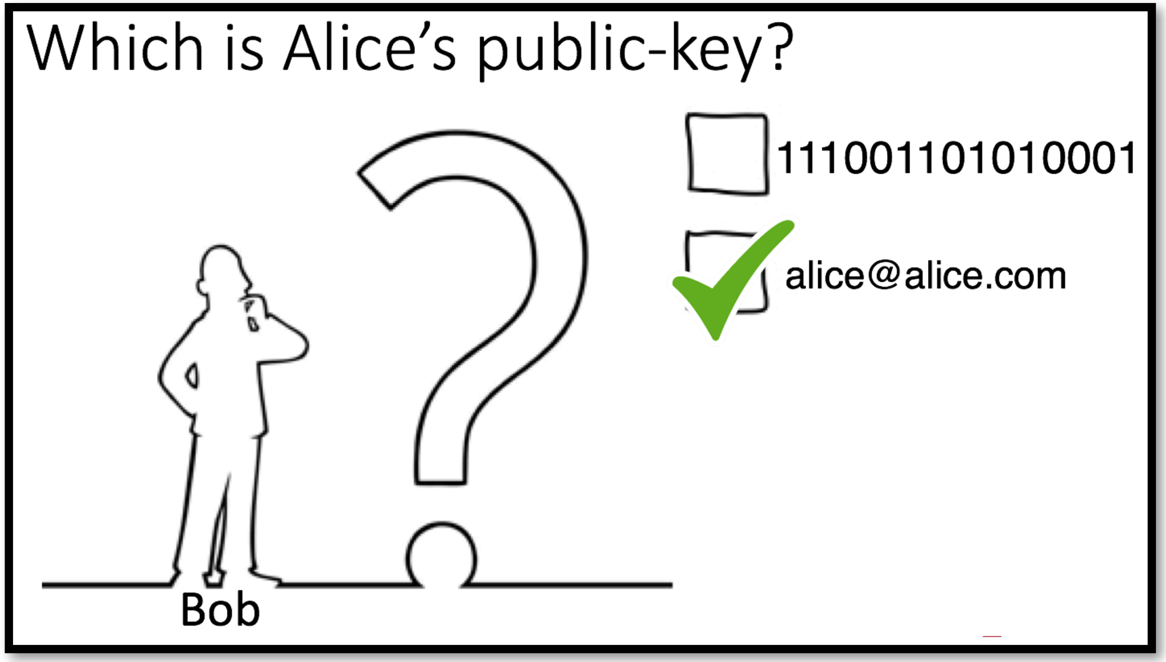
- We present *Authentication of Things* (AoT)
  - AoT can be set up over wireless mediums using strong cryptography
- AoT is based on a two-domain paradigm
  - A Cloud domain and a Home domain
- AoT uses Attribute-Based Signatures to provide authentication over the entire IoT lifecycle

Agenda

- Introduction
- Background
  - IBC, ABC, ABAC
- AoT
- Evaluation
- Conclusion

Identity-Based Cryptography

- In IBC, public-keys are meaningful rather than a random and unintelligible string of bits
  - It is easy to tie a key to its user
  - They are implicitly authenticated



Identity-Based Cryptography (Cont)

- Problem: IBC suffers from the key-escrow problem
  - IBC requires the existence of a totally trusted entity
  - This entity can impersonate any user in the system

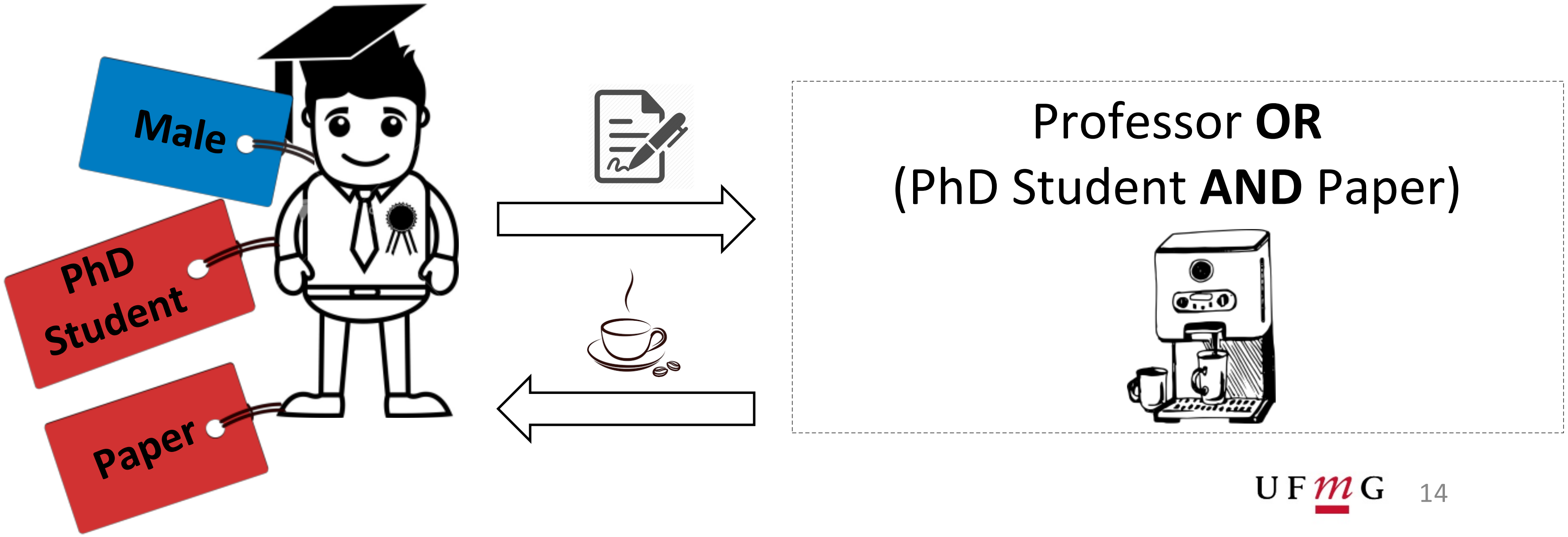
Sometimes such an entity simply does not exist

Agenda

- Introduction
- Background
  - IBC, ABC, ABAC
- AoT
- Evaluation
- Conclusion

Attribute-Based Cryptography (Cont)

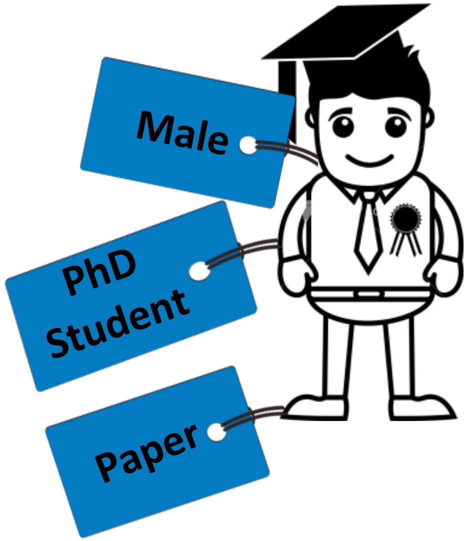
1. Users are assigned keys that reflect their attributes
2. Resources are tied to a logical expression of attributes called *predicate*
3. To access a resource, one needs to sign a request using keys associated to attributes that satisfy the predicate



Agenda

- Introduction
- Background
  - IBC, ABC, ABAC
- AoT
- Evaluation
- Conclusion

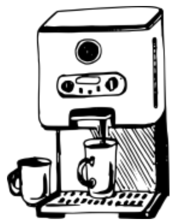
Attribute-Based Access Control



Male

Professor **OR** PhD Student

Professor **OR**
(PhD Student **AND** Paper)



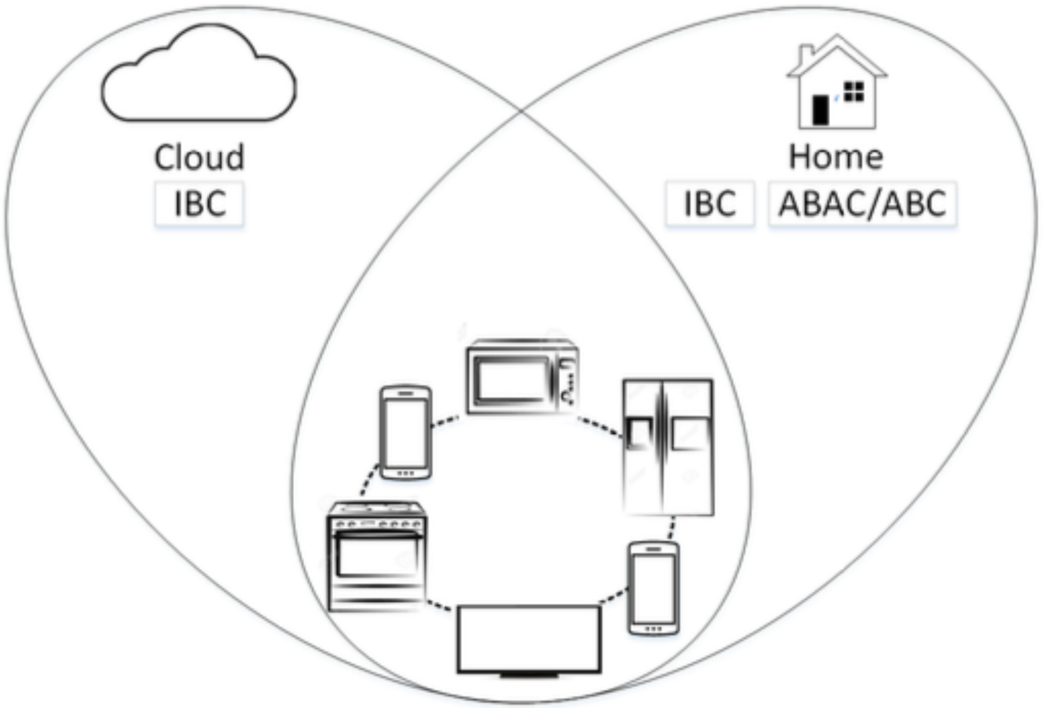
Note the synergy between ABC and ABAC

Agenda

- Introduction
- Background
- AoT
  - Overview, protocols, extra features
- Evaluation
- Conclusion

Overview 1/2

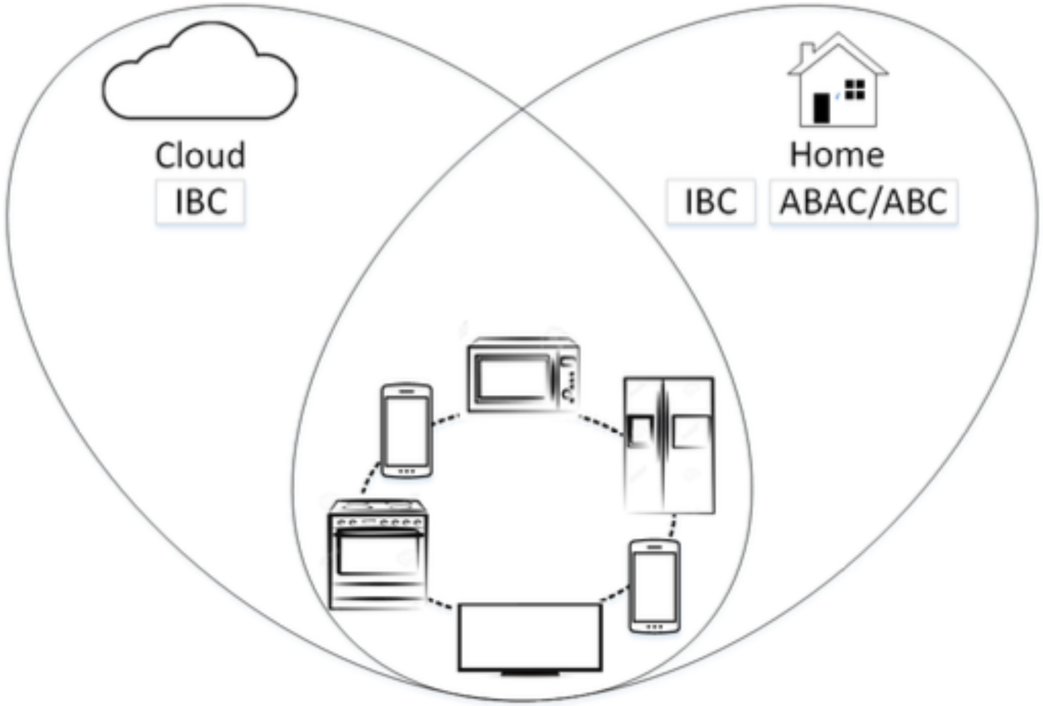
AoT Two Domain Architecture



- Our key insight to tackle the IBC key-escrow problem is to build two distinct IBC setups: Cloud and Home
  - The key escrow still holds in each IBC setup, individually
  - However, the key escrow is now no longer a problem
  - Because requests from one domain are null in the other

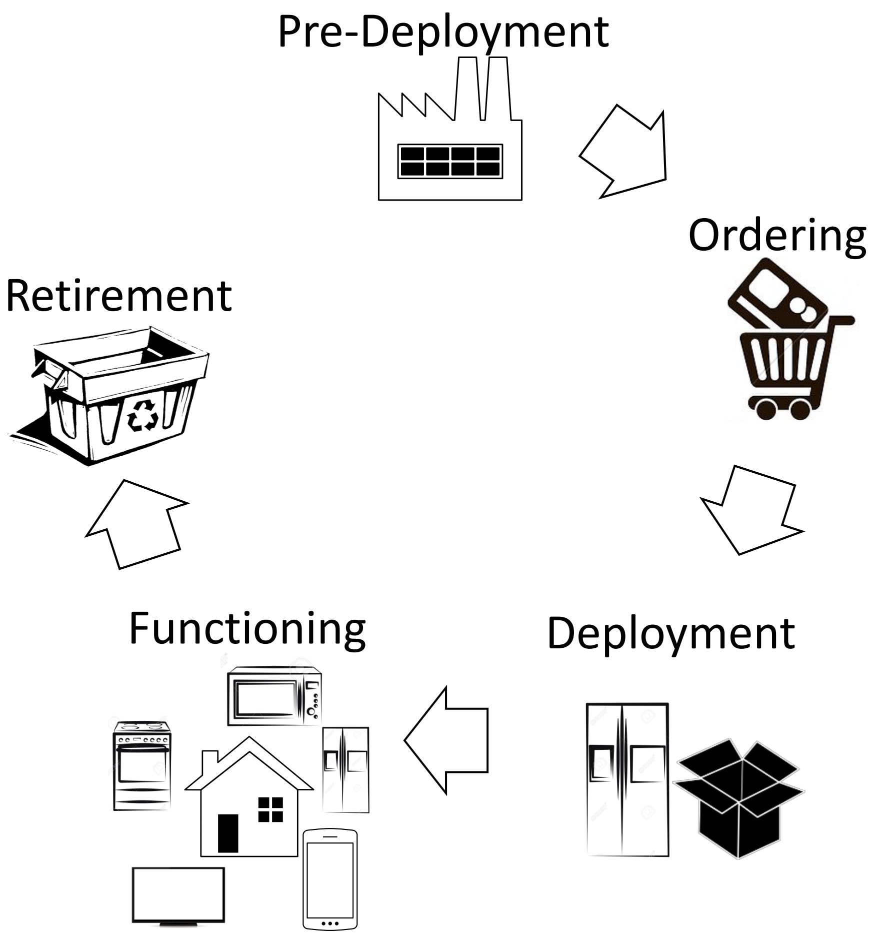
Overview 1/2

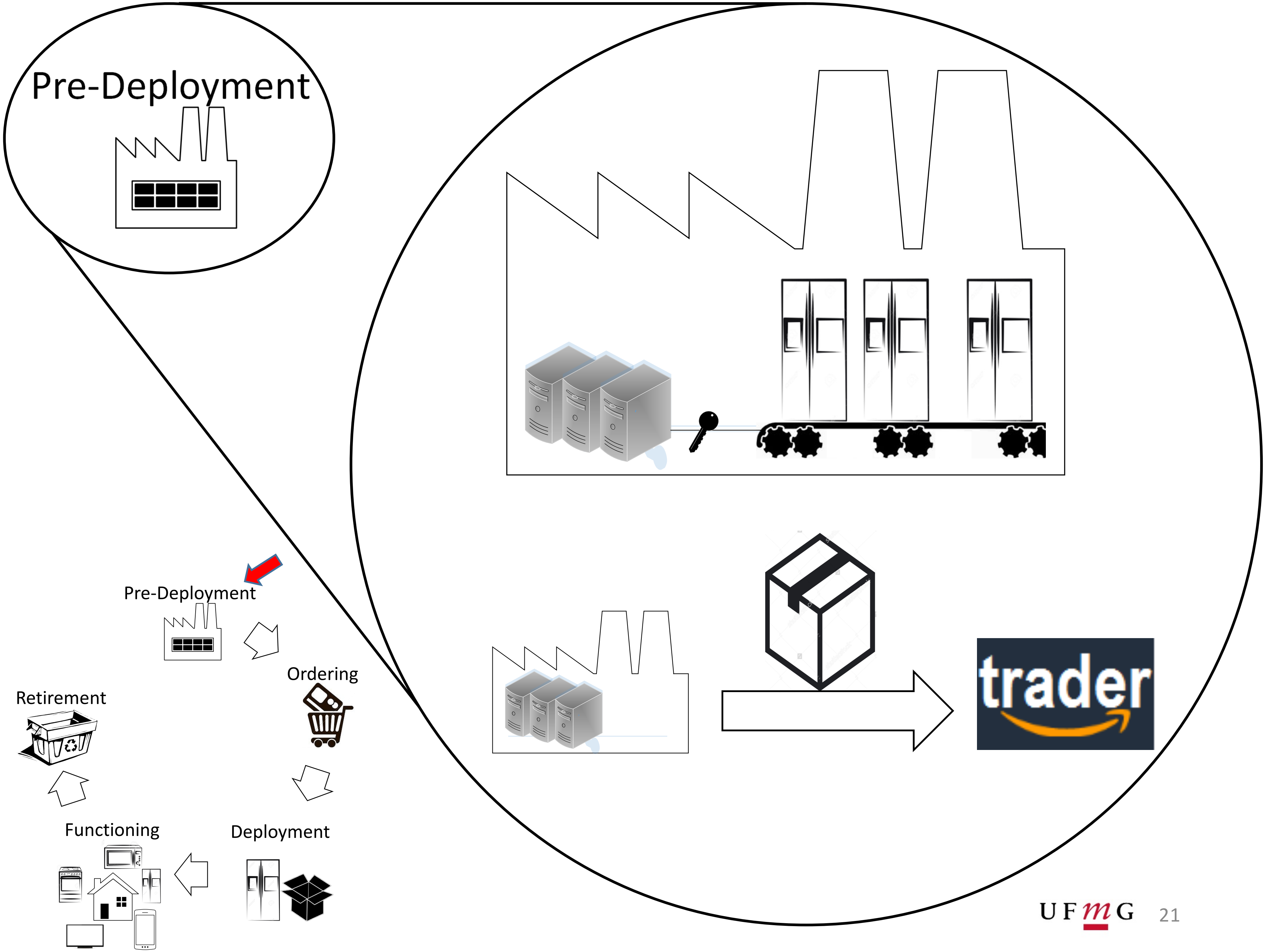
AoT Two Domain Architecture

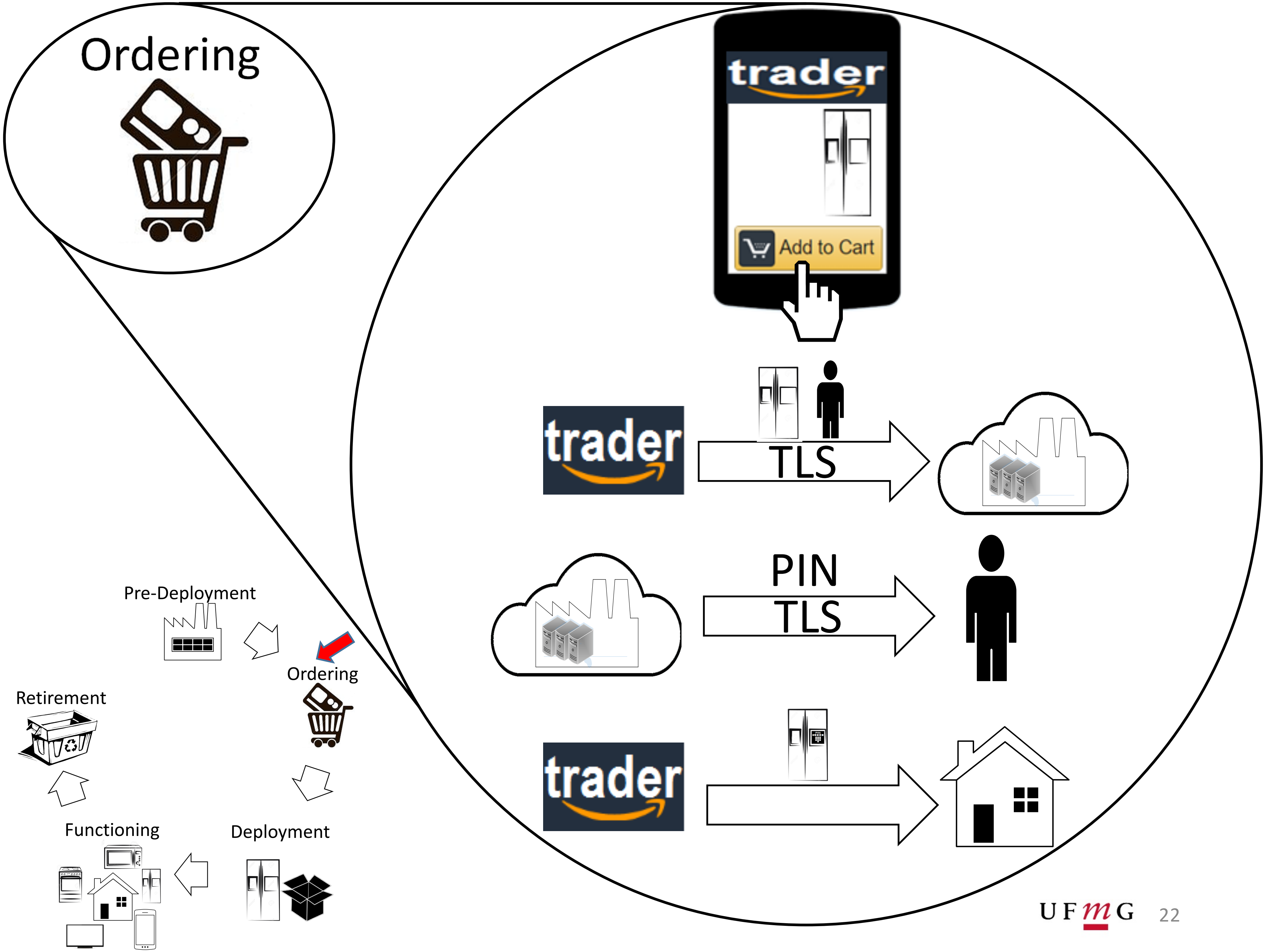


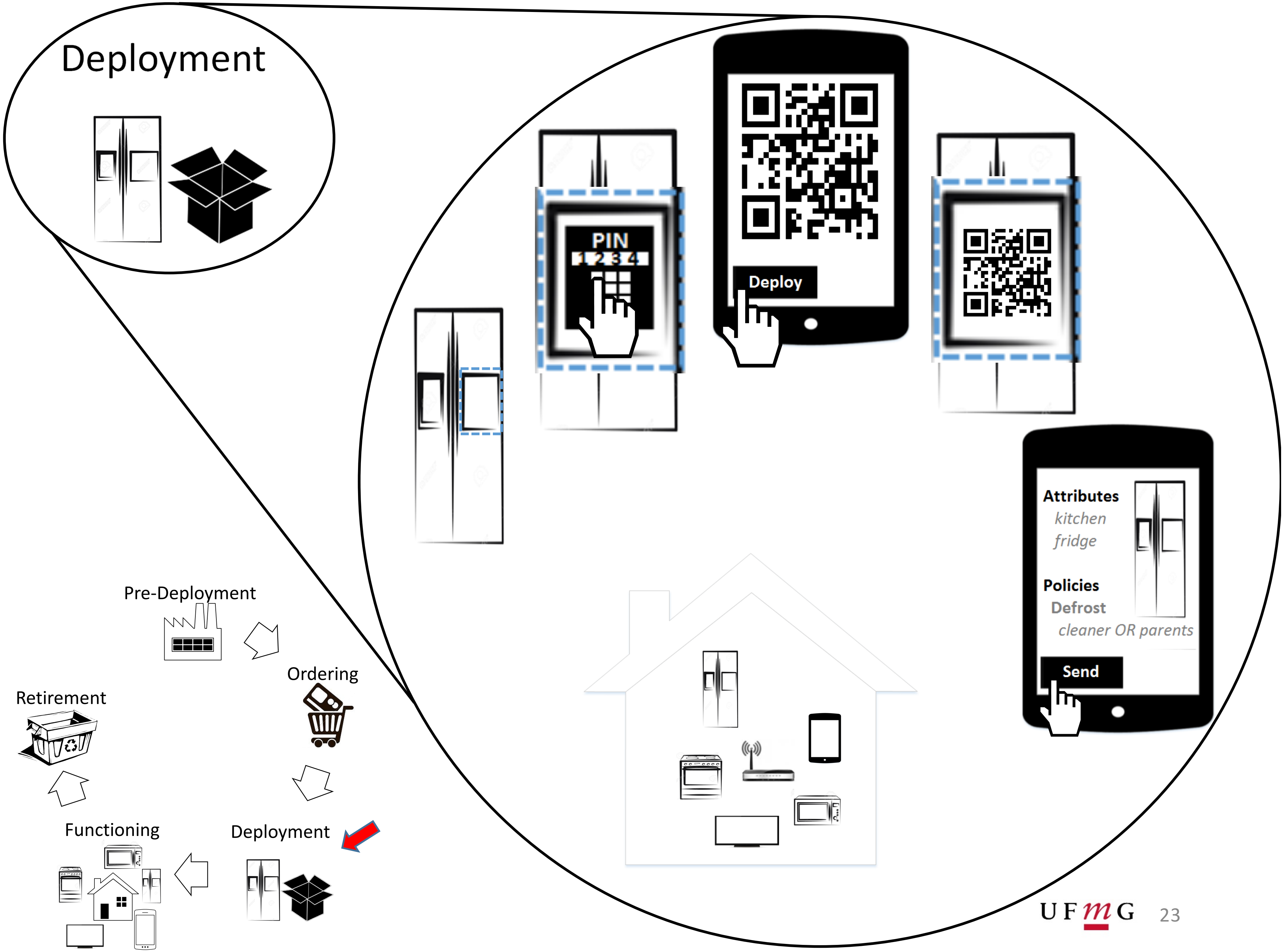
- IBC is used to distribute keys and authenticate the Cloud
- ABAC/ABC are used to control access to device operations

AoT comprises five main protocols One for each IoT device life-cycle

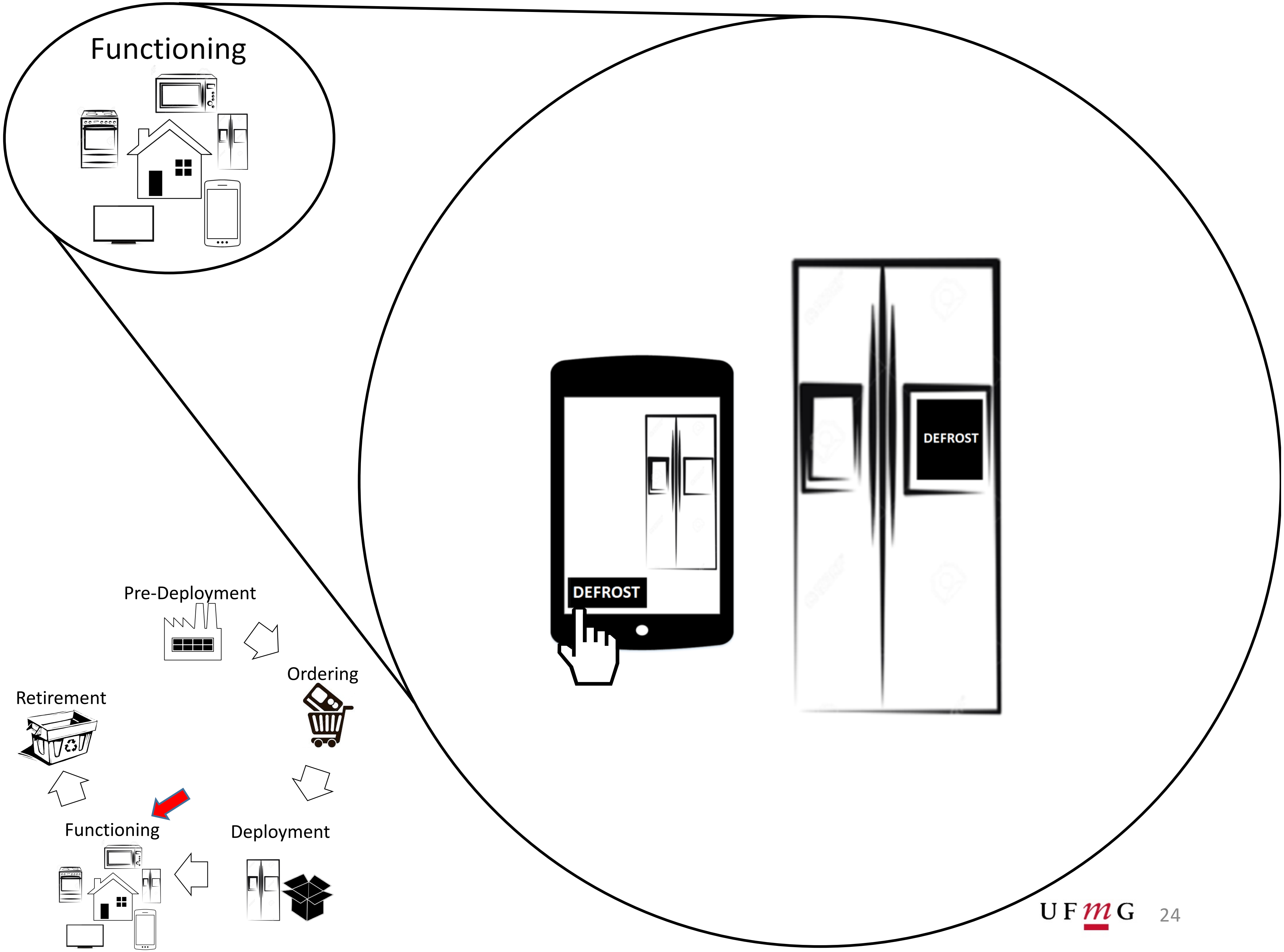


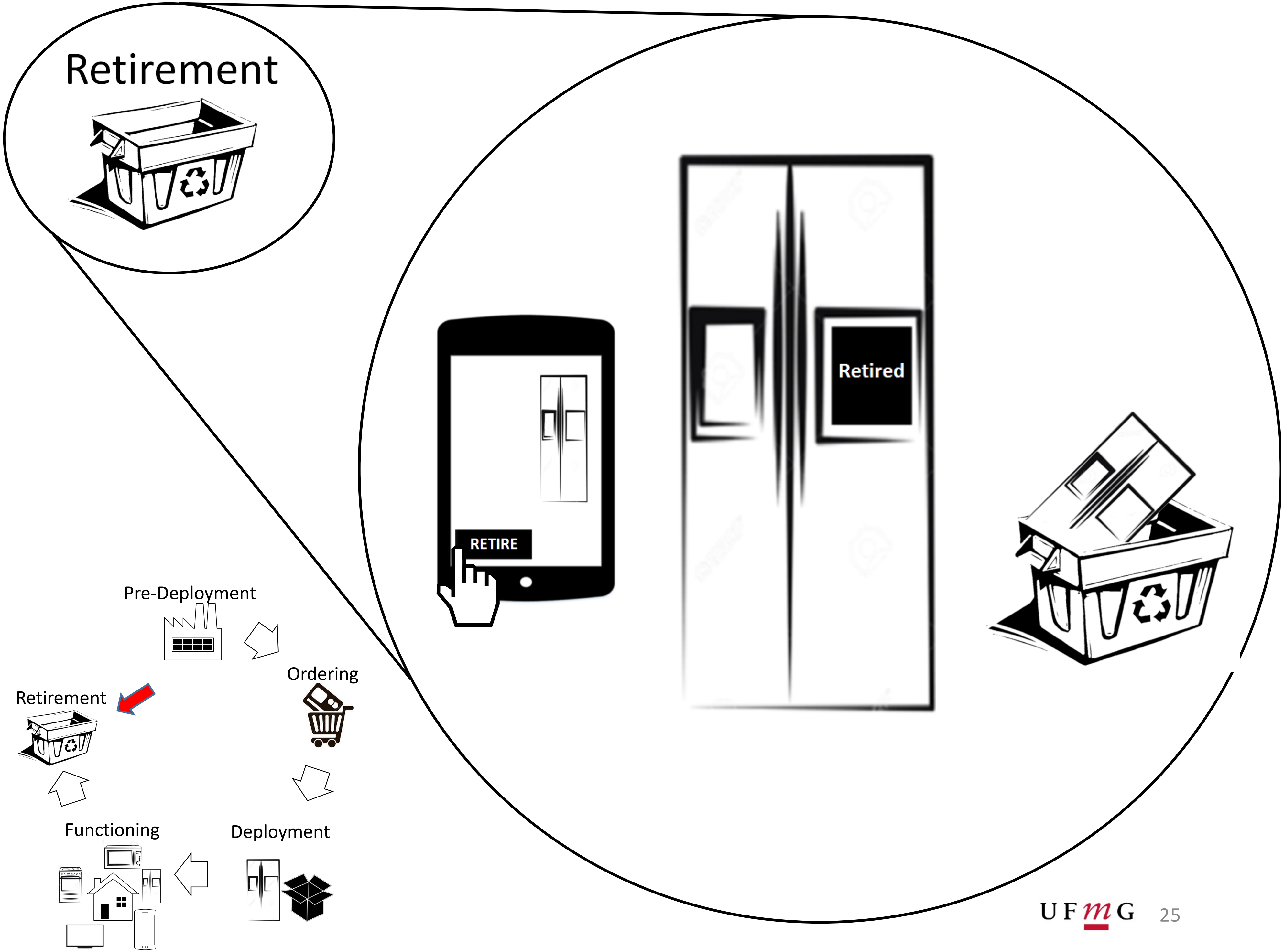

Pre-Deployment
Pre-Deployment
Ordering
Retirement
Functioning
Deployment
trader


Ordering
trader
Add to Cart
trader
TLS
PIN
TLS
trader
Pre-Deployment
Ordering
Retirement
Functioning
Deployment



Deployment
PIN
1 2 3 4
Deploy
Attributes
kitchen
fridge
Policies
Defrost
cleaner OR parents
Send
Pre-Deployment
Ordering
Retirement
Functioning
Deployment

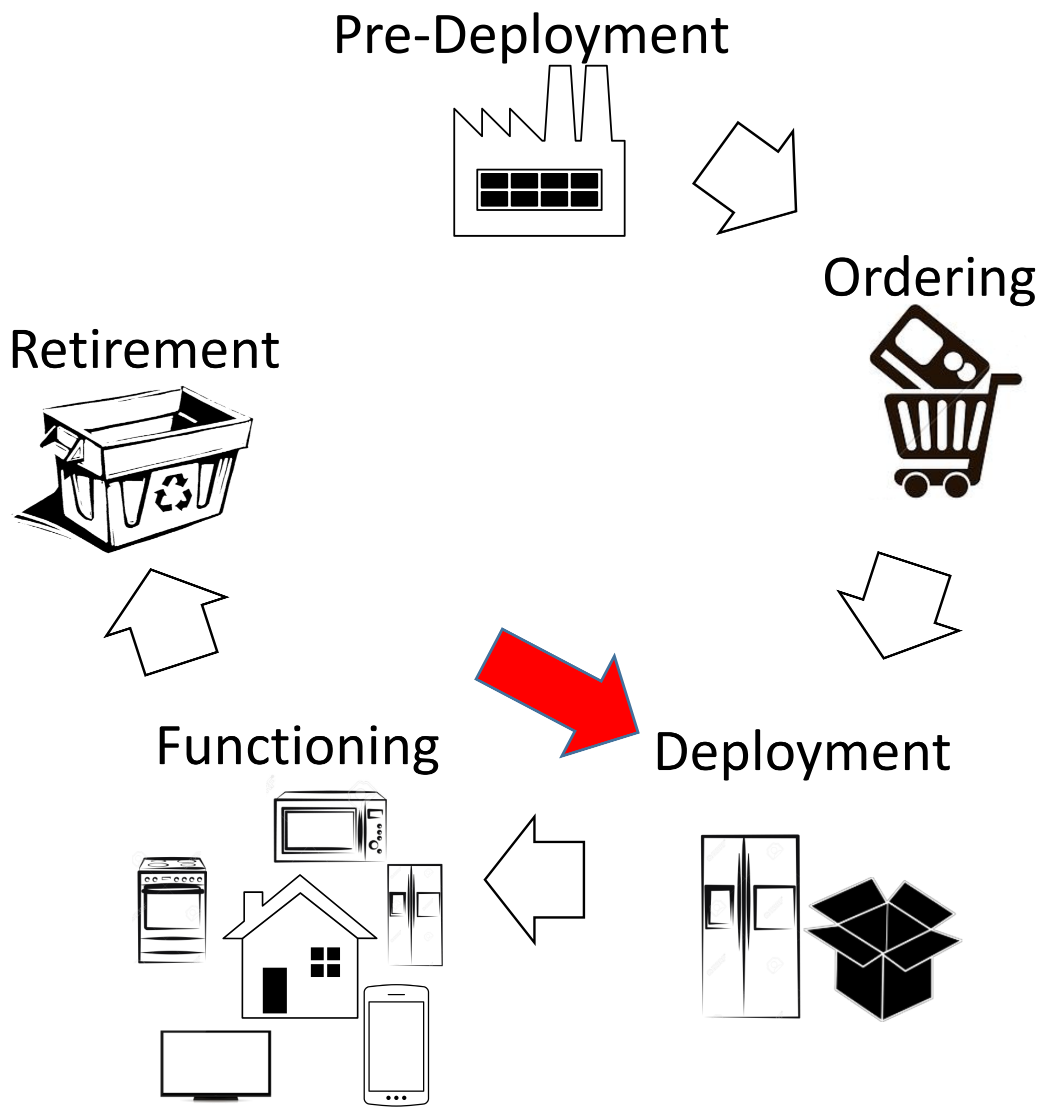

Functioning
DEFROST
DEFROST
Pre-Deployment
Ordering
Retirement
Functioning
Deployment


Retirement
RETIRE
Retired
Pre-Deployment
Ordering
Retirement
Functioning
Deployment

Agenda

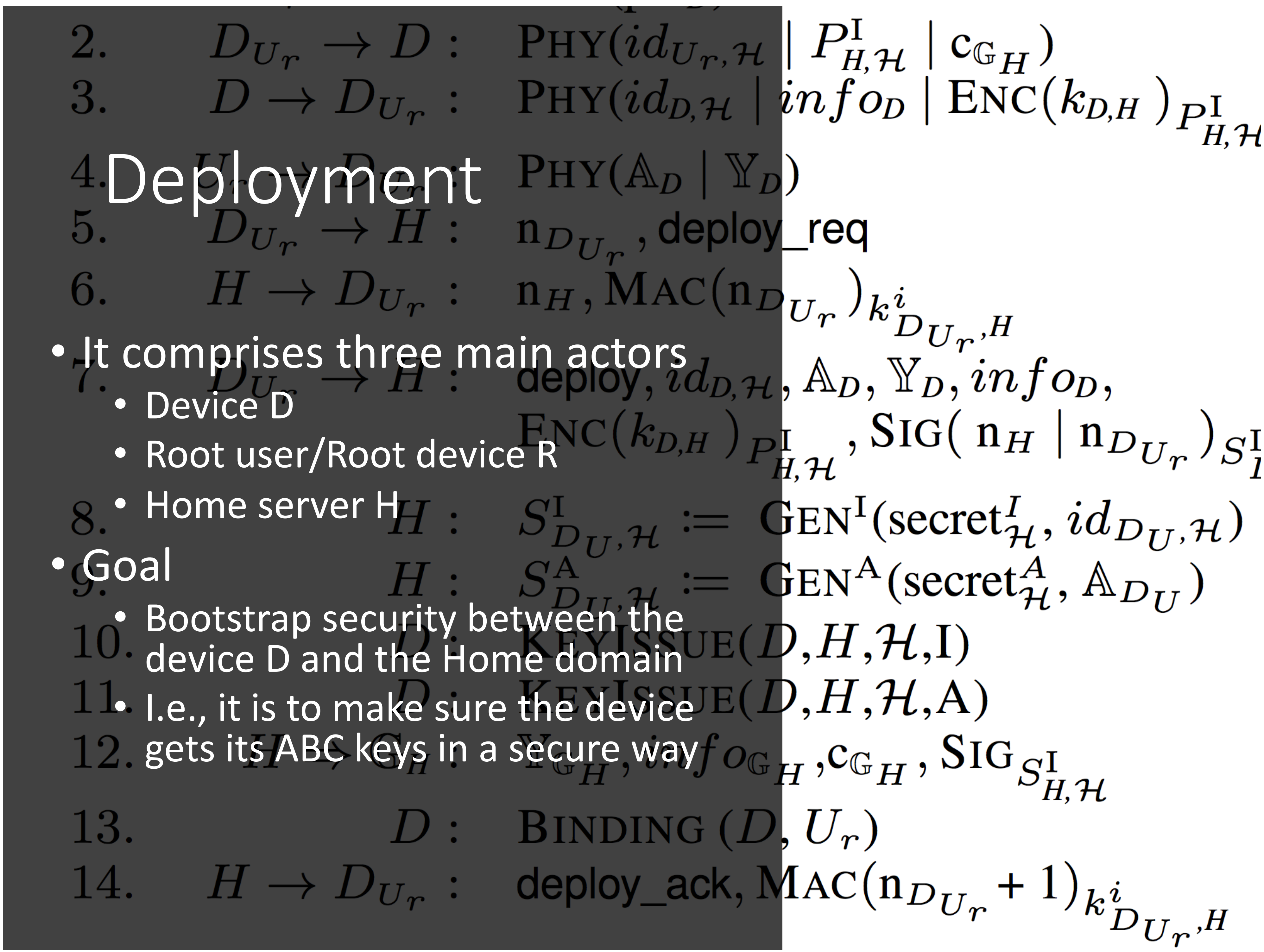
- Introduction
- Background
- AoT
  - Overview, protocols, extra features
- Evaluation
- Conclusion

Life-Cycle

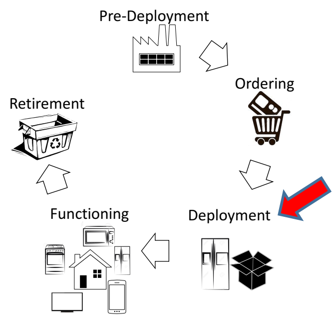

Pre-Deployment
Ordering
Deployment
Functioning
Retirement

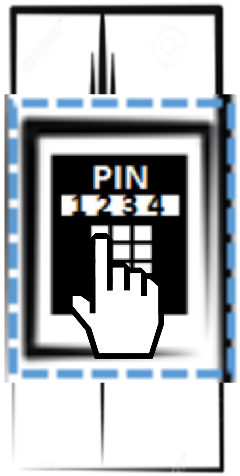
Deployment

- It comprises three main actors
  - Device D
  - Root user/Root device R
  - Home server H
- Goal
  - Bootstrap security between the device D and the Home domain
  - I.e., it is to make sure the device gets its ABC keys in a secure way

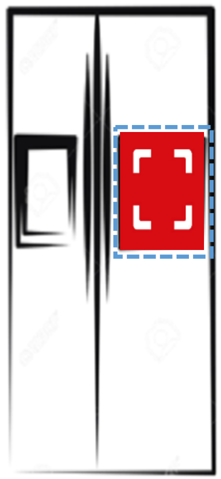


Root enters the PIN onto the device
Device is put into setup mode





PIN input



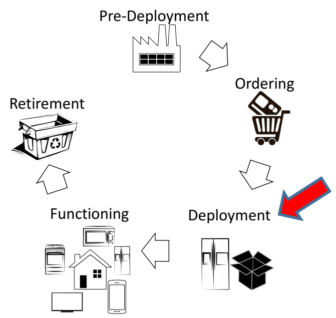
Device D

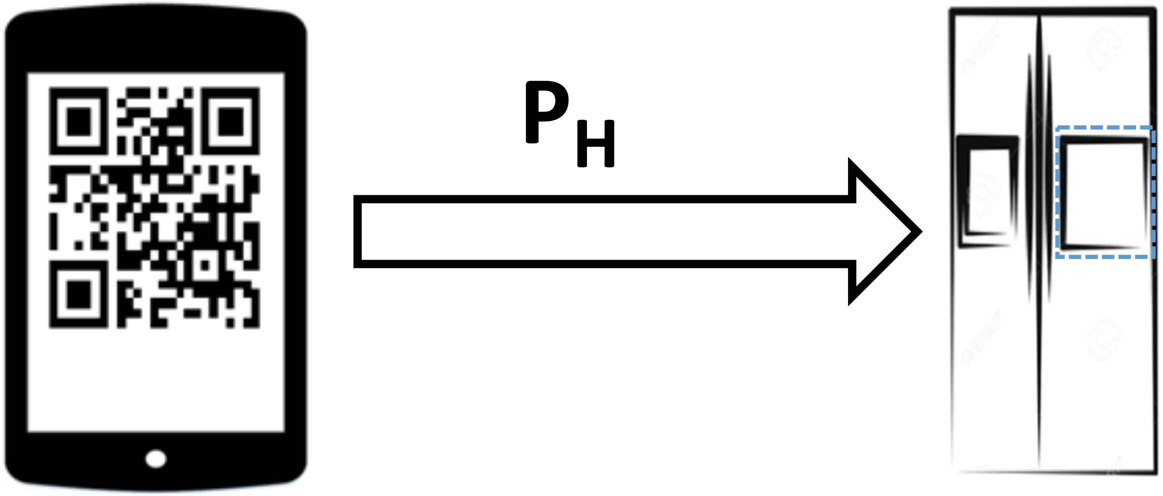
The PIN input leverages the physical security of the home

Only the Root holds the PIN and then can start the process

The security of the communication does not rely on the PIN

Device gets Home server's public-key P_H

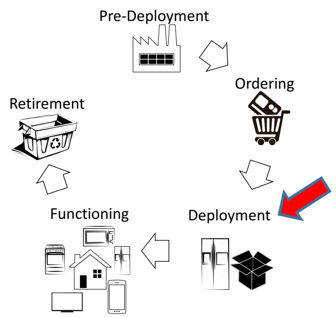


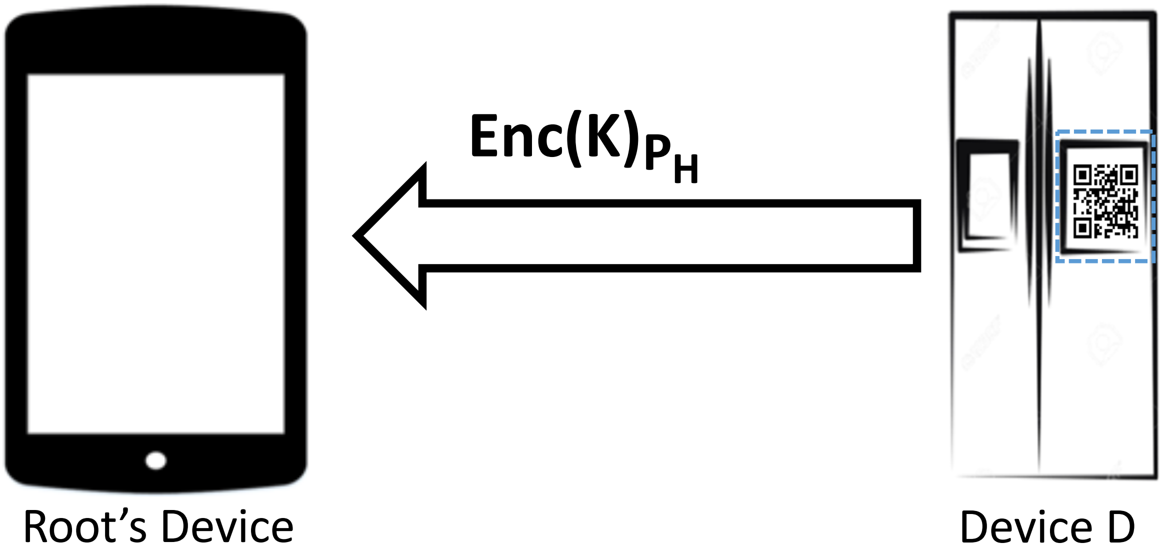


Device trusts the Root because he has the PIN

Device trusts the public-key P_H is authentic

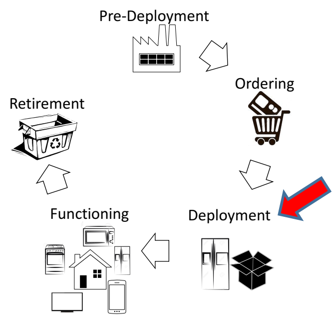
Device generates a secret key K
Device encrypts the key K and sends K to Root

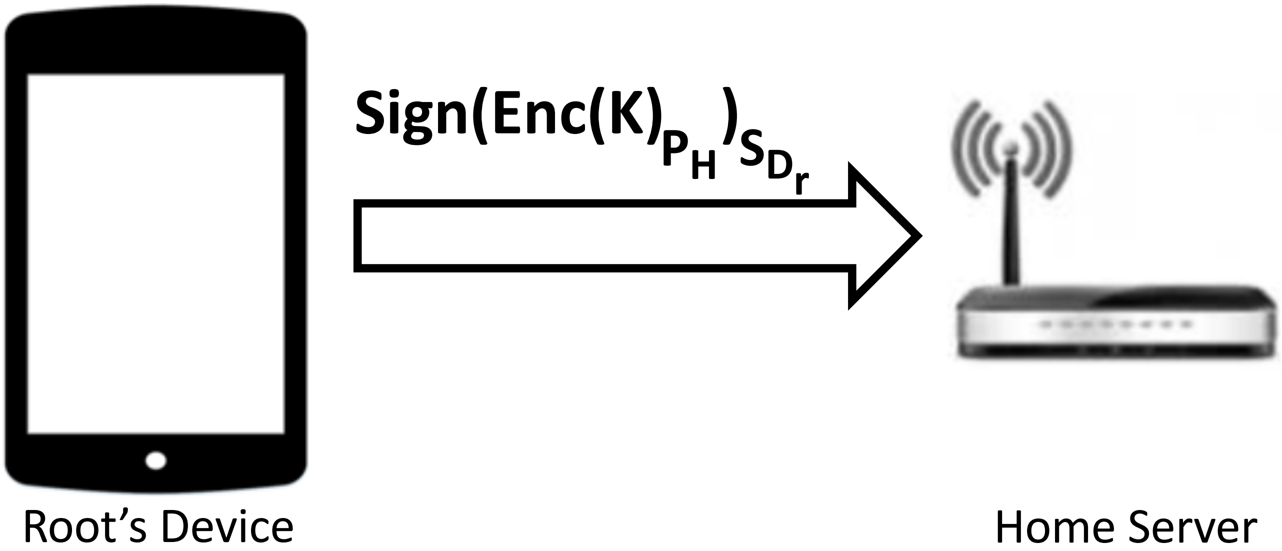




Even the Root does not have access to the key K

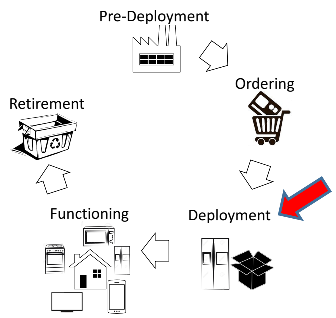
Root delivers the key K to Homer server

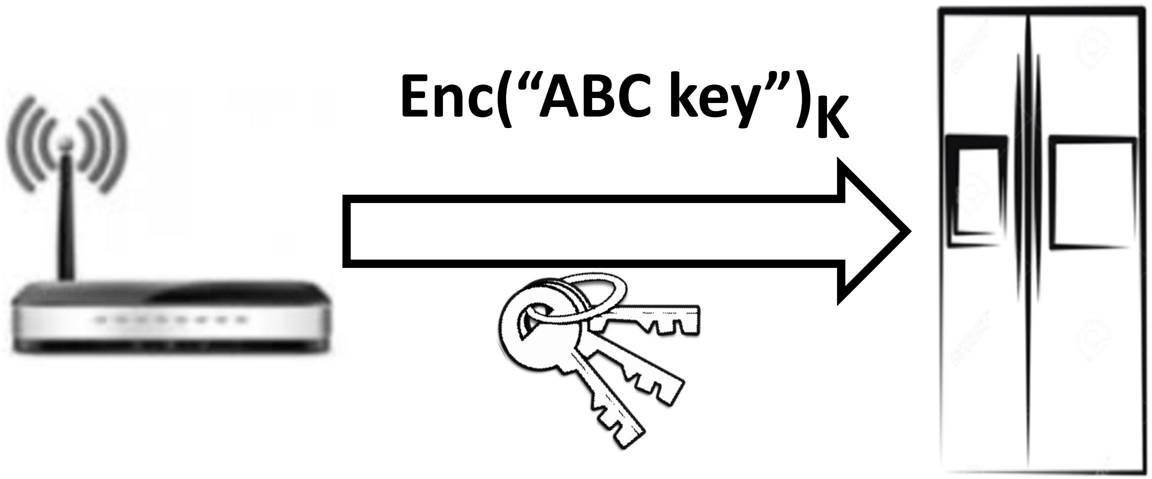




The secret key K is delivered in a secure manner

ABC keys are sent to the device



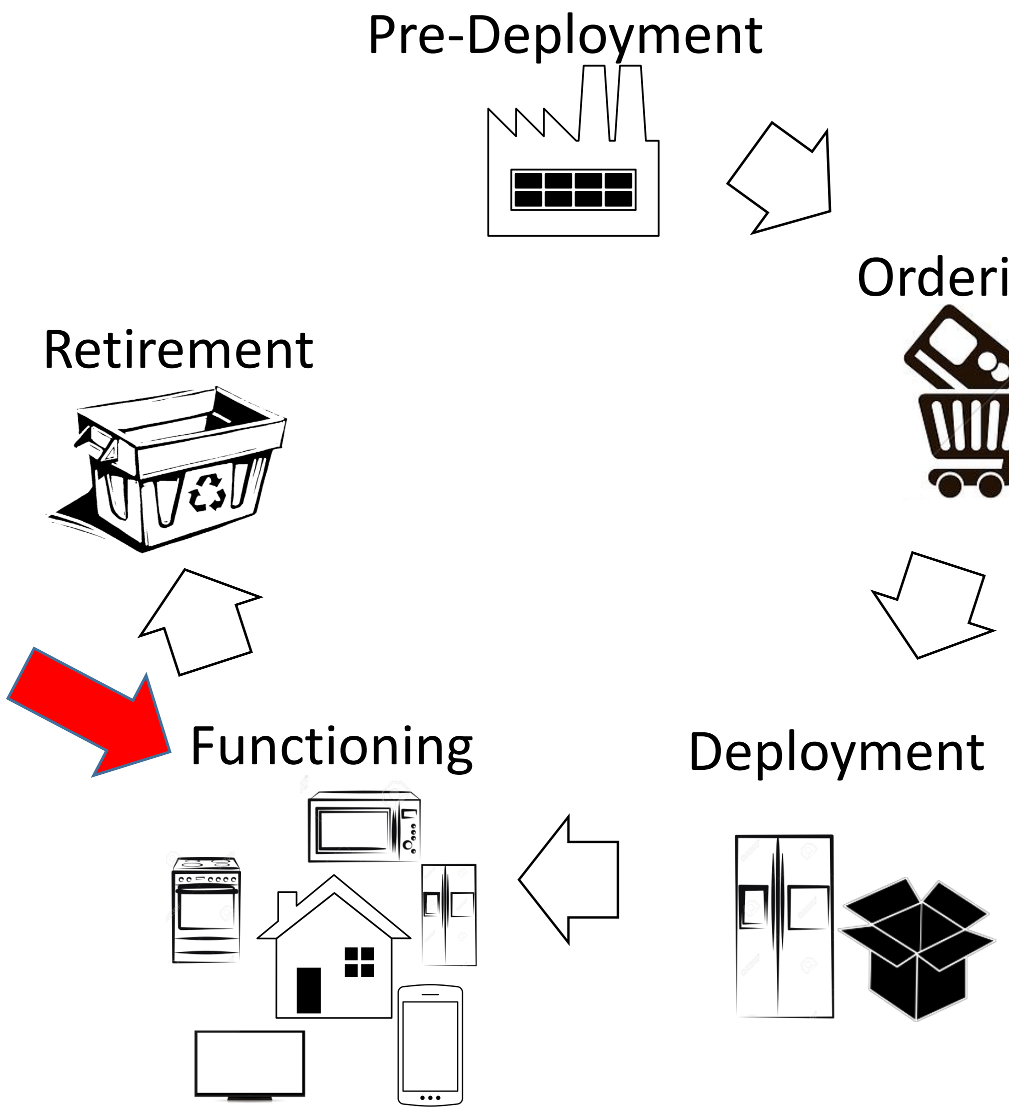


Home Server

Device D

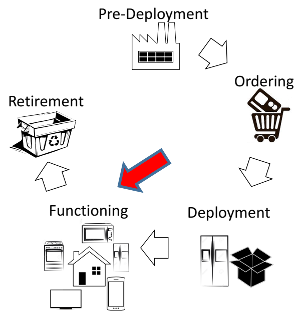
The ABC key issue is protected using the key K

Life-Cycle


Pre-Deployment
Ordering
Deployment
Functioning
Retirement

FUNCTIONING(*user* U, *device* A, *device* B, *operation* op)

1.	$A \rightarrow B$:	n_A, op_req
2.	$B \rightarrow A$:	$n_B, \Upsilon_{op}, \mathrm{MAC}(n_A)_{k^i_{A,B}}$
3.	$A \rightarrow B$:	op, $\mathrm{SIG}(n_B \mid n_A)_{S^A_{A,\mathcal{H}}}$
4.	B :	performs operation op



Description

Normal network operation

1. Device A requests an operation over device B
2. B tells A the attributes required to perform the operation
3. A presents his signature as a proof it has the attributes
4. B performs the operation

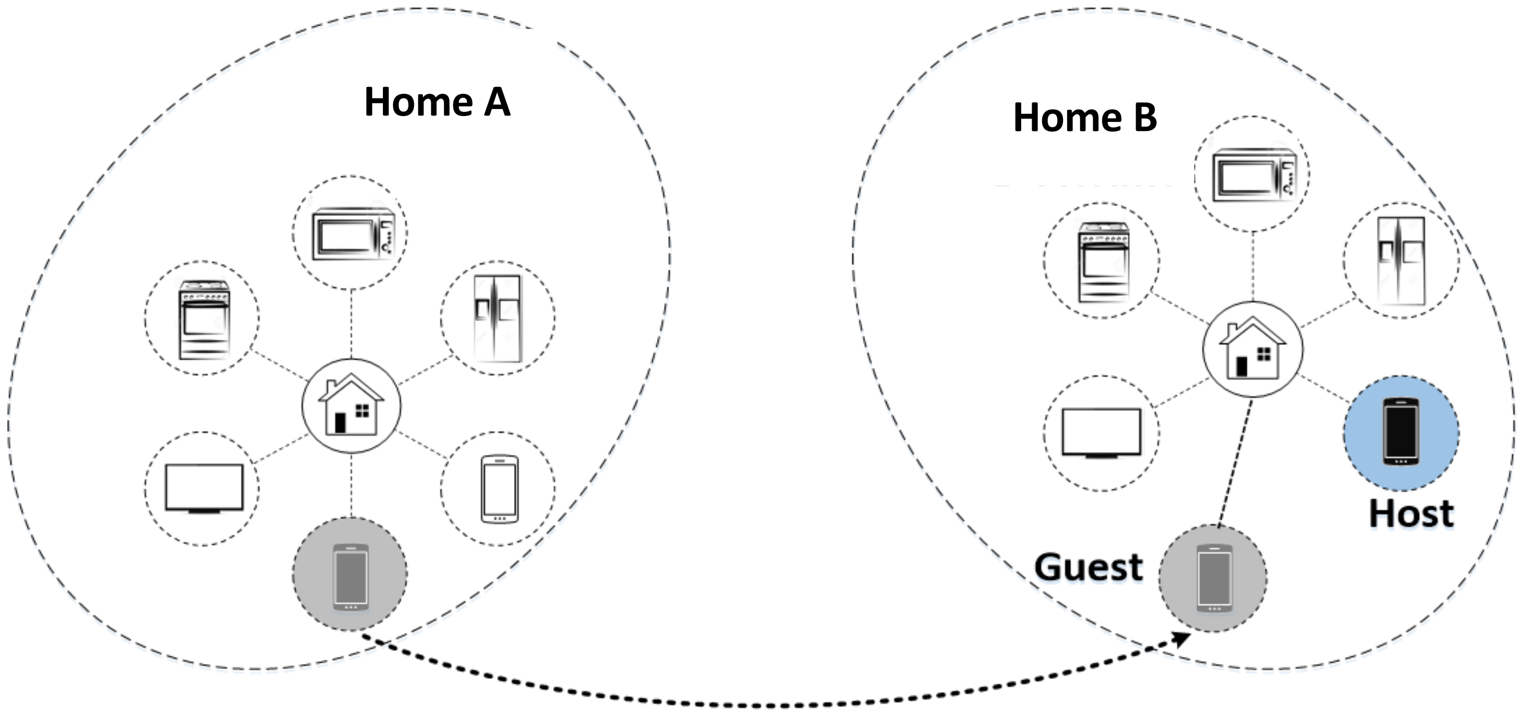
A challenge-response protocol is going on behind the scenes

Agenda

- Introduction
- Background
- AoT
  - Overview, protocols, extra features
- Evaluation
- Conclusion

InterDomain Key Agreement Protocol

- Allows a guest and a host device to run a protocol to agree on a symmetric secret key
  - As long as their Home servers agree on common public parameters of Elliptic Curve Cryptography



Agenda

- Introduction
- Background
- AoT
- Evaluation
- Conclusion

$$\prod_{i=1}^{\ell} e\Big(S_i, (A_j B_j^{u(i)})^{\mathbf{M}_{ij}}\Big)$$

ABS: Computation

- The core of our access control scheme is the ABS
  - The ABC signature scheme
- ABS computational complexity is dominated by the computation of pairings
  - Pairing is the crypto primitive used to implement IBC
- Pairings are known for being expensive in terms of both time and memory

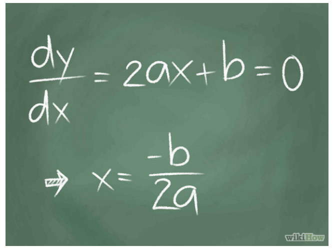
$$\prod_{i=1}^{\ell} e\left(S_i, (A_j B_j^{u(i)})^{\mathbf{M}_{ij}}\right)$$

ABS: Computation

- The core of our access control scheme is the ABS
  - The ABC signature scheme
- ABS computational complexity is dominated by the computation of pairings
  - Pairing is the crypto primitive used to implement IBC
- Pairings are known for being expensive in terms of both time and memory

ABS requires not only the computation of a pairing, but rather the computation of a <u>product</u> of pairings

ABS: Optimization & Code



- Optimization
  - We optimized our implementation of ABS by using a multi-pairing operation
  - Results of operations in common are computed only once, stored, and then shared by all the pairings
- Code
  - RELIC is our underlying cryptographic library
  - Our code is publicly available http://www.dcc.ufmg.br/~lemosmaia/aot

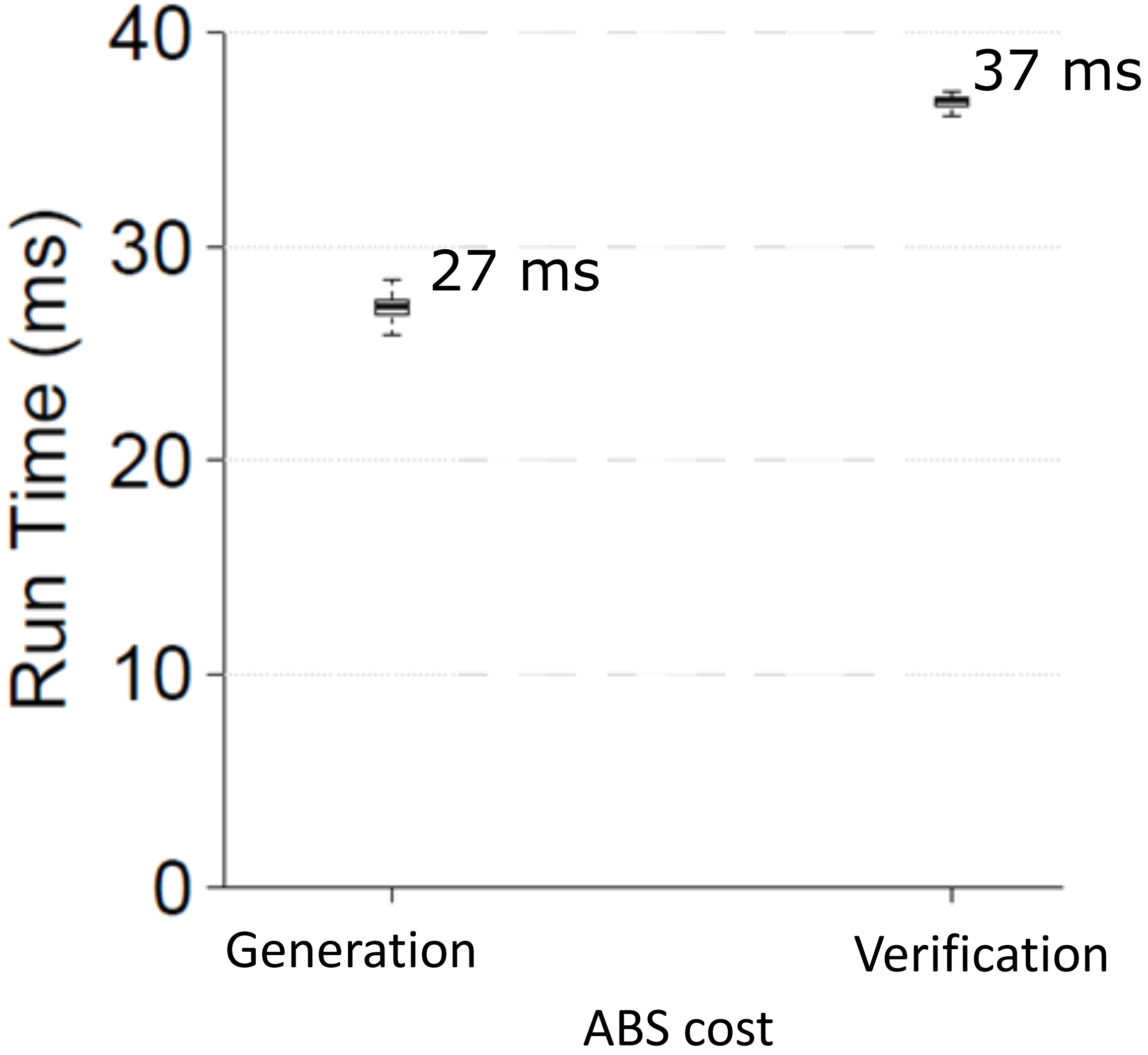
Code

Attribute Based Signature (ABS) (download)

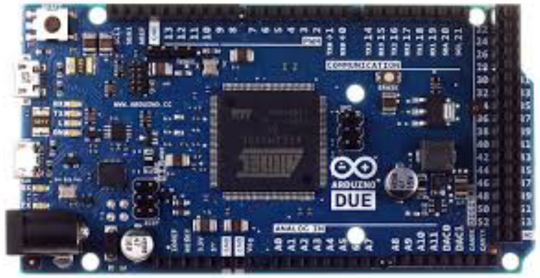
Devices

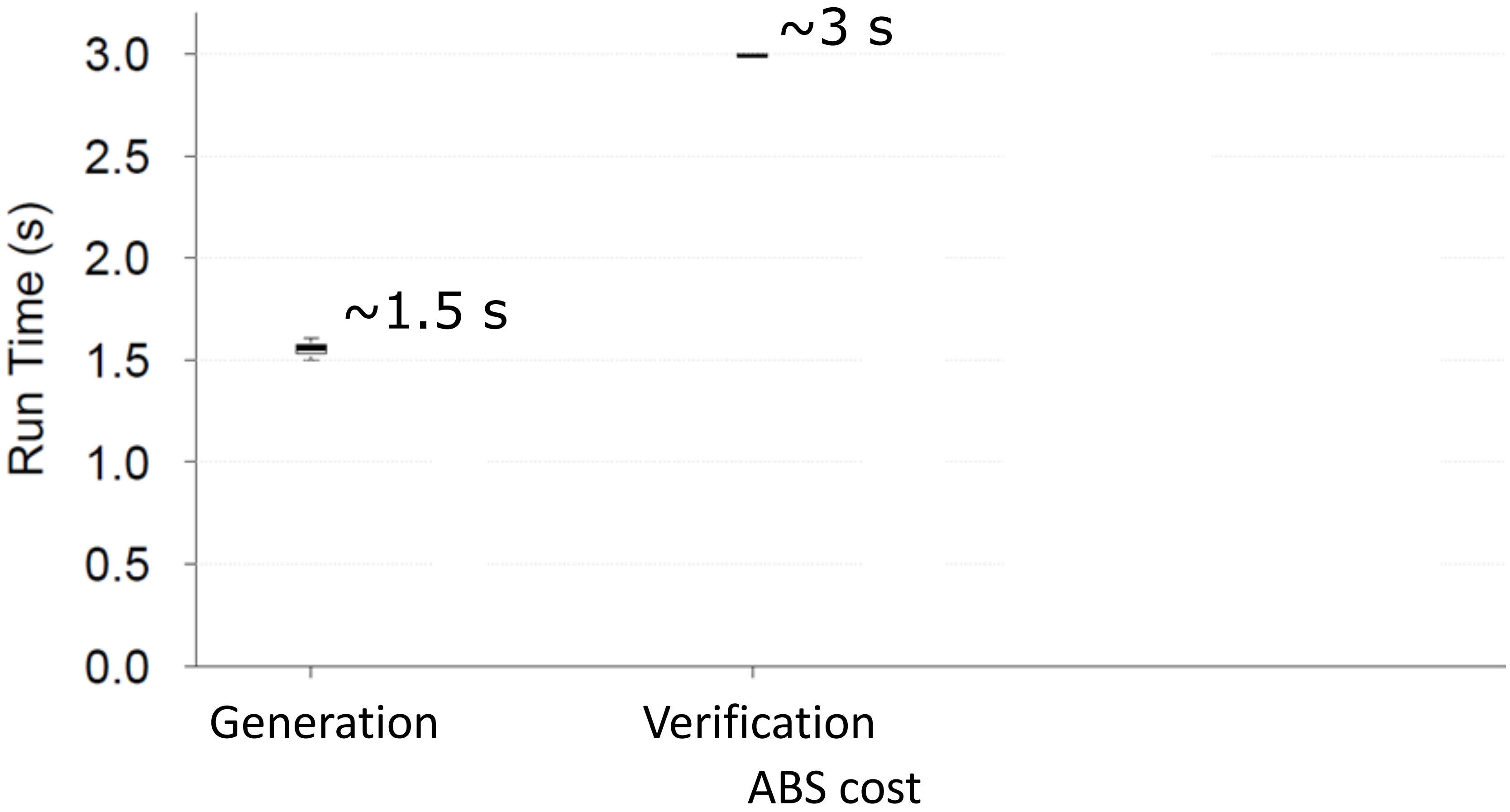
Device	LG G4	Arduino Due
Word Size	64 bits	32 bits
Clock	2 GHz	84 MHz
RAM	3 GB	96 KB

Resourceful

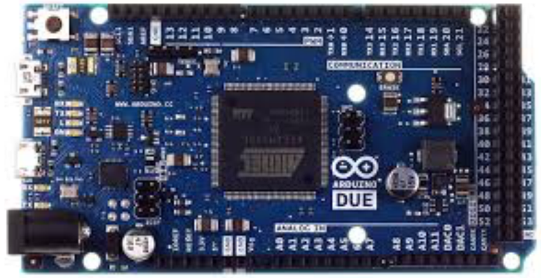


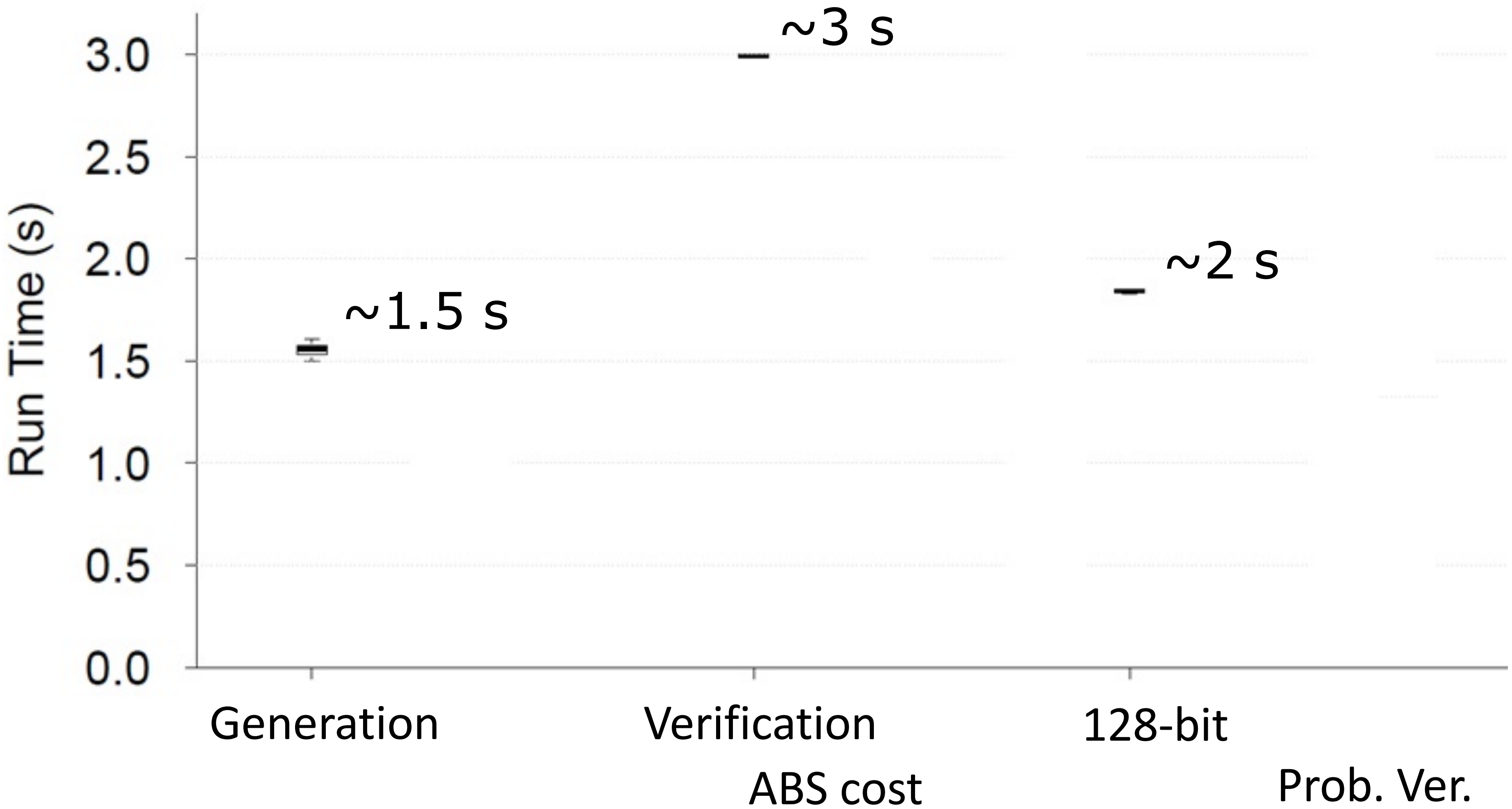
Resource-constrained



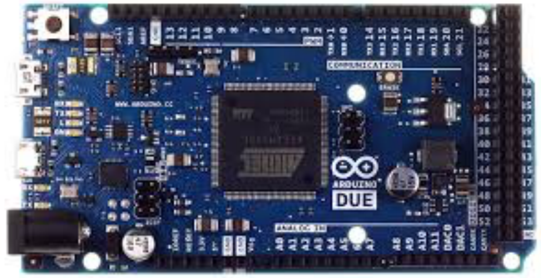


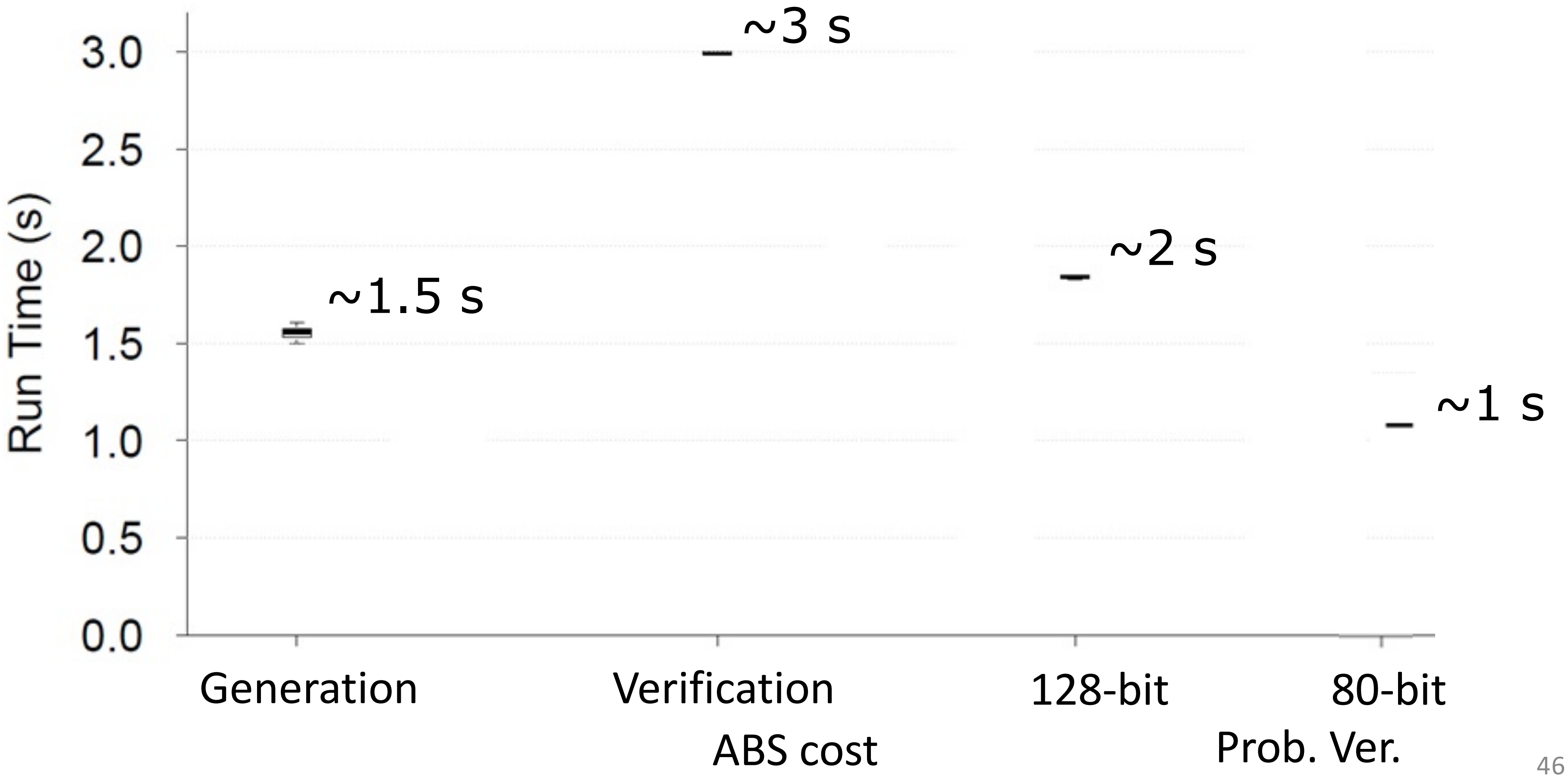
Resource-constrained





Resource-constrained





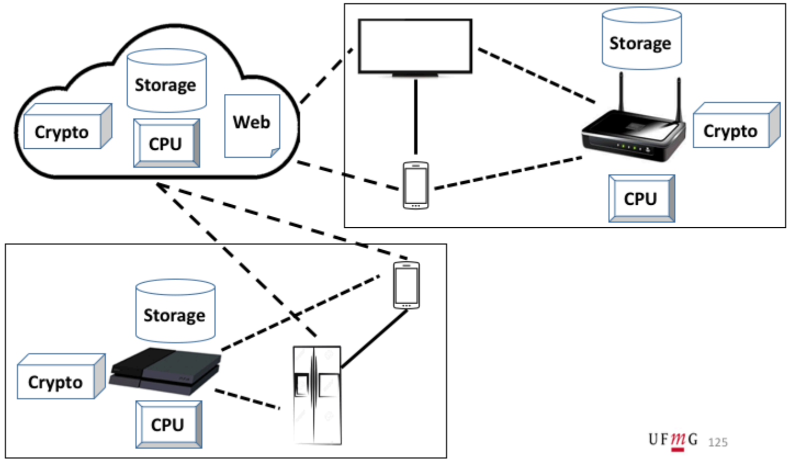
Agenda

- Introduction
- Background
- AoT
- Evaluation
- Conclusion

Conclusion

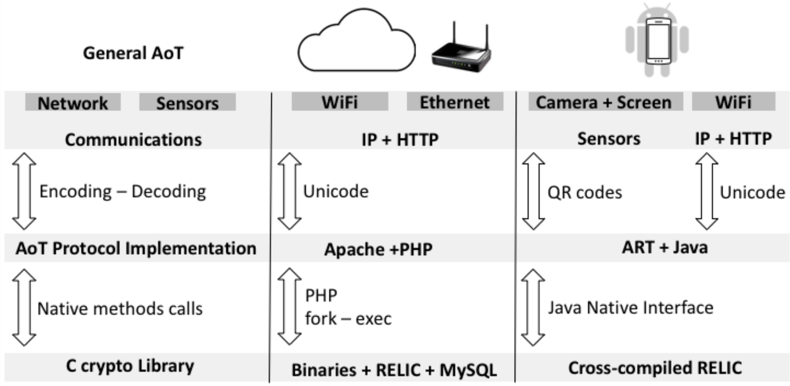
- AoT is an authentication and access control scheme for the entire IoT device life-cycle
- AoT relies on strong cryptography to bootstrap security
- AoT addresses real-world IoT challenges like interdomain authentication
- We evaluated performance and security tradeoffs on diverse hardware

Architecture



UFMG 125

Software Stack

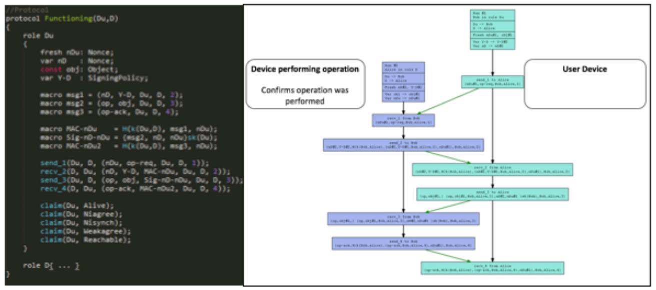


UFMG 126

Be my guest!

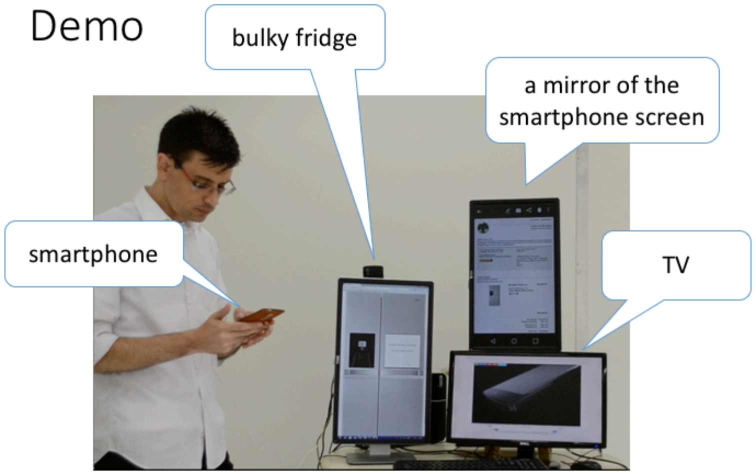
Security evaluation

- We use Scypher to formally verify that AoT indeed provides the aforementioned properties



UFMG 128

Demo



UFMG 129

LG Life's Good

leo@dcc.ufmg.br

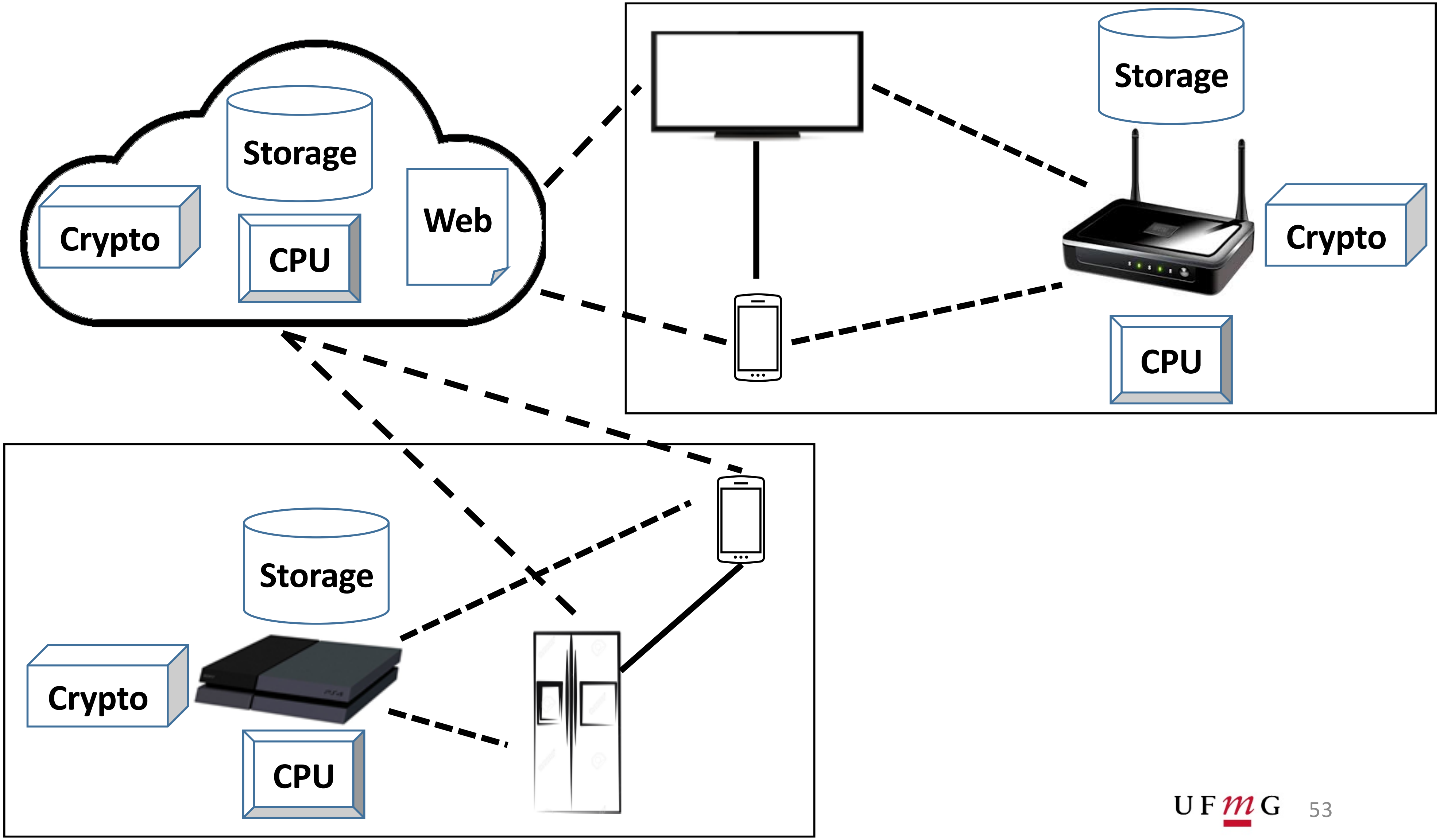
Thanks

leonardo.barbosa@dcc.ufmg.br

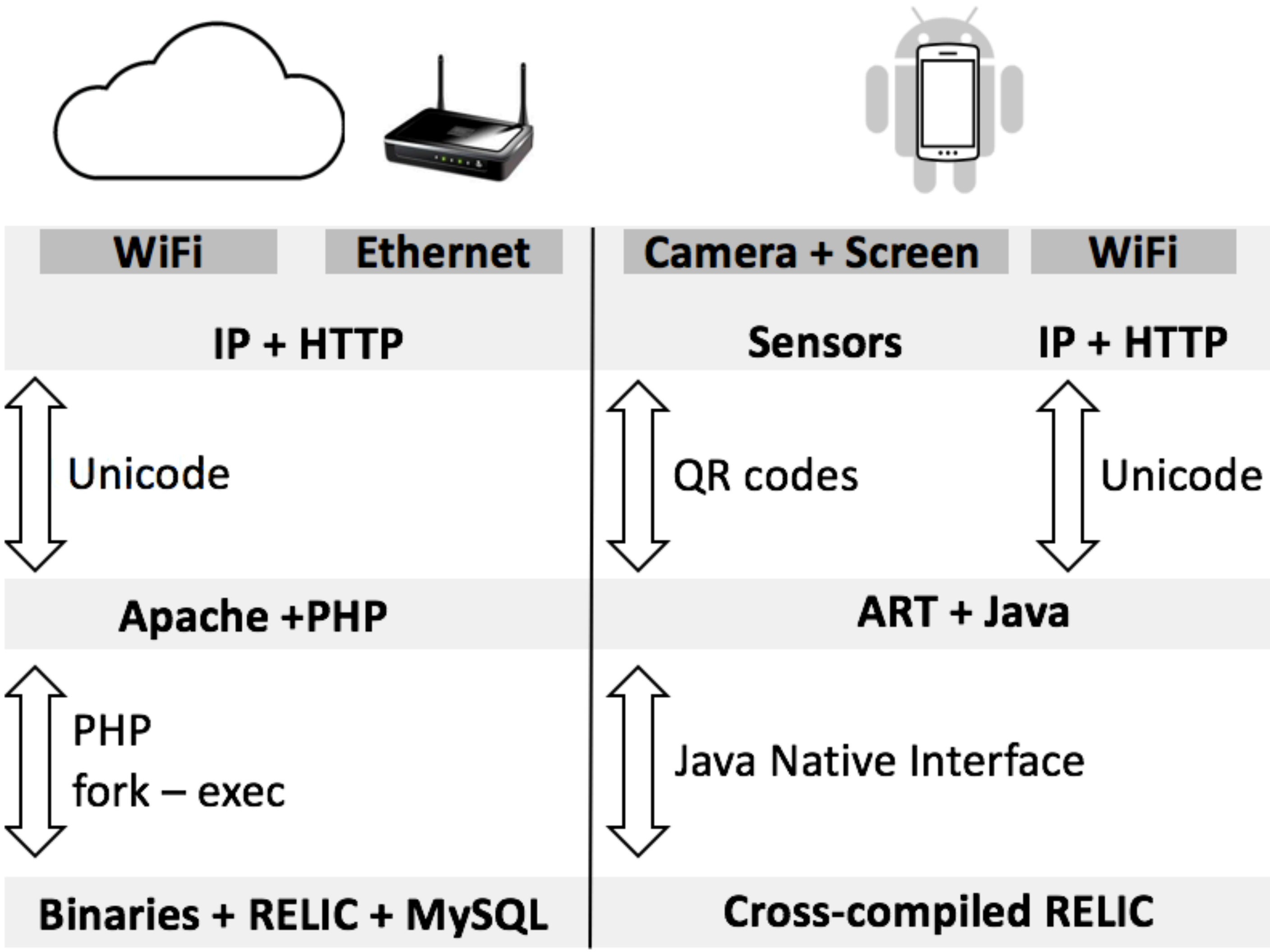
Backup

Development

Architecture



Software Stack



Demo


bulky fridge
a mirror of the smartphone screen
Who is this guy!?
smartphone
TV

Attribute-Based Signature

- The core of our access control scheme is the ABS
- Our ABS implementation is based on the work

Attribute-Based Signatures:
Achieving Attribute-Privacy and Collusion-Resistance

Hemanta Maji* Manoj Prabhakaran* Mike Rosulek*

April 15, 2008

Abstract

We introduce a new and versatile cryptographic primitive called *Attribute-Based Signatures* (ABS), in which a signature attests not to the identity of the individual who endorsed a message, but instead to a (possibly complex) claim regarding the attributes she posseses. ABS offers:

RELIC Cryptographic Library

- We used RELIC as our cryptographic library as RELIC implements some cryptosystems required by AoT
    1. Sakai-Ohgishi-Kasahara key agreement protocol
    2. Boneh-Franklin Identity-Based Encryption scheme
    3. Bellare-Namprempre-Nevem ID-Based Signature
    4. Etc.
- RELIC supports resource-constrained devices
    - 8-, 16-, 32-bit embedded processors
    - Under 4KiB of RAM

Auxiliary Protocols

Auxiliary protocols

- SessionKey
  - Uses a PRF and a shared counter to generate session keys on demand
  - C.f. Perrig, Szewczyk, Wen, Culler, and Tygar 2001
- KeyAgreement
  - Authenticated ID-based protocol for key agreement
  - It is nonintereactive and saves storage and memory
  - Based on the work of Sakai, Ohgishi, and Kasahara 2001
- Binding
  - Binds a device to a user on the Cloud server
- KeyIssue
  - Issues *signing* keys to devices

SESSIONKEY(*key* k, *counter* i)

1. $k^i := \mathrm{PRF}(i)_k$
2. $i := i + 1$
3. return k^i

KEYAGREEMENT(*device* A, *device* B)

1. $A \rightarrow B:$ n_A, session_req
2. $B:$ $P^{I}_{A,\mathcal{H}} := \mathrm{MAP}(id_{A,\mathcal{H}})$
3. $B:$ $k_{A,B} := \hat{e}(S^{I}_{B,\mathcal{H}}, P^{I}_{A,\mathcal{H}})$
4. $B:$ $c_A := 0$
5. $B:$ $k^i_{A,B} := \mathrm{SESSIONKEY}(k_{A,B}, c_A)$
6. $B \rightarrow A:$ $n_B, \mathrm{MAC}(n_A)_{k^i_{A,B}}$
7. $A:$ {*Computes the pairwise key* $k^i_{A,B}$, *analogously*}
8. $A \rightarrow B:$ $\mathrm{MAC}(n_B \mid n_A)_{k^i_{A,B}}$
9. $B \rightarrow A:$ session_ack, $\mathrm{MAC}(n_A + 1)_{k^i_{A,B}}$

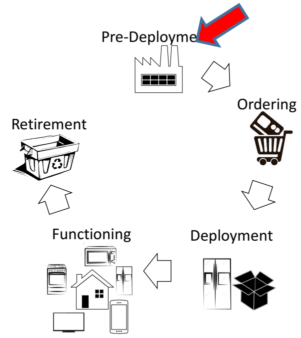
BINDING(*device* D, *user* U)

1. $D \rightarrow C:$ n_D, bind_req
2. $C \rightarrow D:$ $n_C, \mathrm{MAC}(n_D)_{k^i_{D,C}}$
3. $D \rightarrow C:$ bind, $id_{U,C}$, $\mathrm{SIG}(n_C \mid n_D)_{S^{I}_{D,C}}$
4. $C:$ binds U to D
5. $C \rightarrow D:$ bind_ack, $\mathrm{MAC}(n_D + 1)_{k^i_{D,C}}$

KEYISSUE(*device* D, *server* S, *domain* $\mathcal{Z}$, *cryptosystem* Y)

1. $D \rightarrow S:$ n_D, issue_req
2. $S \rightarrow D:$ $n_S, \mathrm{MAC}(n_D)_{k^i_{D,S}}$
3. $D \rightarrow S:$ $\mathrm{MAC}(n_S \mid n_D)_{k^i_{D,S}}$
4. $S \rightarrow D:$ $\mathrm{ENC}(S^{Y}_{D,\mathcal{Z}})_{k^i_{D,S}}$, issue_ack, $\mathrm{MAC}(n_D + 1)_{k^i_{D,S}}$

Main protocols



$\textsc{Pre-deployment}(Device\ D)$

$$
\begin{array}{rrl}
1. & C: & id_{D,C} := D's\ serial\# \\
2. & C: & S^{\mathrm{I}}_{D,C} := \textsc{Gen}^{\mathrm{I}}(\mathrm{secret}^{I}_{C}, id_{D,C}) \\
3. & C \rightarrow D: & \textsc{Phy}(id_{D,C} \mid S^{\mathrm{I}}_{D,C} \mid k_{D,C} \mid \mathrm{pin}_D \mid \mathrm{c}_D) \\
4. & C \Rrightarrow T_D: & D
\end{array}
$$

Description

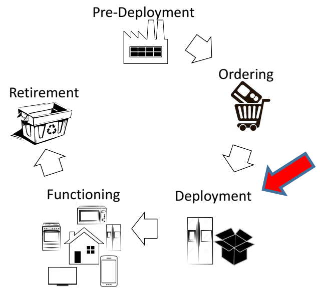
At the Factory

1. Cloud server C generates the identity of the device D
2. Server generates the device's private key
3. Server loads D's pvt key, pairwise key, PIN into the device
4. Server ships the device to its trader

Crypto Material is delivered over a physically secured channel

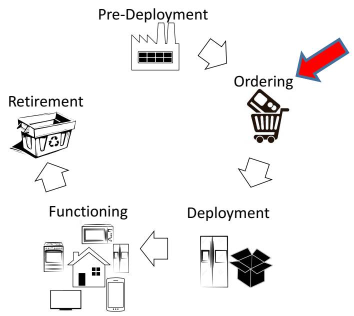
$\text{DEPLOYMENT}(\textit{device } D)$

1. $U_r \rightarrow D$: $\text{PHY}(\mathrm{pin}_D)$
2. $D_{U_r} \rightarrow D$: $\text{PHY}(id_{U_r,\mathcal{H}} \mid P^{\mathrm{I}}_{H,\mathcal{H}} \mid \mathrm{c}_{\mathbb{G}_H})$
3. $D \rightarrow D_{U_r}$: $\text{PHY}(id_{D,\mathcal{H}} \mid info_D \mid \text{ENC}(k_{D,H})_{P^{\mathrm{I}}_{H,\mathcal{H}}})$
4. $U_r \rightarrow D_{U_r}$: $\text{PHY}(\mathbb{A}_D \mid \mathbb{Y}_D)$
5. $D_{U_r} \rightarrow H$: $\mathrm{n}_{D_{U_r}}$, deploy_req
6. $H \rightarrow D_{U_r}$: $\mathrm{n}_H, \text{MAC}(\mathrm{n}_{D_{U_r}})_{k^i_{D_{U_r},H}}$
7. $D_{U_r} \rightarrow H$: deploy, $id_{D,\mathcal{H}}, \mathbb{A}_D, \mathbb{Y}_D, info_D,$ $\text{ENC}(k_{D,H})_{P^{\mathrm{I}}_{H,\mathcal{H}}}, \text{SIG}(\mathrm{n}_H \mid \mathrm{n}_{D_{U_r}})_{S^{\mathrm{I}}_{D_{U_r},\mathcal{H}}}$
8. H : $S^{\mathrm{I}}_{D_U,\mathcal{H}} := \text{GEN}^{\mathrm{I}}(\mathrm{secret}^I_{\mathcal{H}}, id_{D_U,\mathcal{H}})$
9. H : $S^{\mathrm{A}}_{D_U,\mathcal{H}} := \text{GEN}^{\mathrm{A}}(\mathrm{secret}^A_{\mathcal{H}}, \mathbb{A}_{D_U})$
10. D : $\text{KEYISSUE}(D,H,\mathcal{H},\mathrm{I})$
11. D : $\text{KEYISSUE}(D,H,\mathcal{H},\mathrm{A})$
12. $H \Rightarrow \mathbb{G}_H$: $\mathbb{Y}_{\mathbb{G}_H}, info_{\mathbb{G}_H}, \mathrm{c}_{\mathbb{G}_H}, \text{SIG}_{S^{\mathrm{I}}_{H,\mathcal{H}}}$
13. D : $\text{BINDING}(D, U_r)$
14. $H \rightarrow D_{U_r}$: deploy_ack, $\text{MAC}(\mathrm{n}_{D_{U_r}} + 1)_{k^i_{D_{U_r},H}}$



$\textsc{Ordering}(\textit{device } D, \textit{user } U)$

$$
\begin{array}{lrl}
1. & U \rightarrow T_D : & \text{TLS}(\$ \mid \text{order\_req}) \\
2. & T_D \rightarrow U : & \text{TLS}(\text{order\_ack}) \\
3. & T_D \rightarrow C : & \text{TLS}(\text{D} \mid \text{U}) \\
4. & C \rightarrow U : & \text{TLS}(\text{pin}_D) \\
5. & T_D \Rrightarrow U : & D
\end{array}
$$



Description

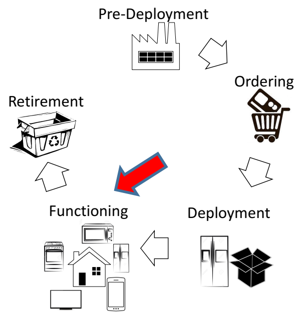
The user U orders the device D from the trader T

1. User U places the order and pays for the device
2. Trader confirms receipt of payment
3. Trader tells the Cloud server about the order
4. Cloud server sends the user U a PIN for access the device
5. Trader ships the device to user U's home

The whole process is protected via TLS

$$\textsc{Functioning}(\textit{user } U, \textit{device } A, \textit{device } B, \textit{operation } \mathsf{op})$$

$$
\begin{array}{llll}
1. & A \rightarrow B: & \mathrm{n}_A, \mathsf{op\_req} \\
2. & B \rightarrow A: & \mathrm{n}_B, \Upsilon_{op}, \textsc{Mac}(\mathrm{n}_A)_{k^i_{A,B}} \\
3. & A \rightarrow B: & \mathsf{op}, \textsc{Sig}(\mathrm{n}_B \mid \mathrm{n}_A)_{S^{\mathrm{A}}_{A,\mathcal{H}}} \\
4. & B: & \text{performs operation } \mathsf{op}
\end{array}
$$



Description

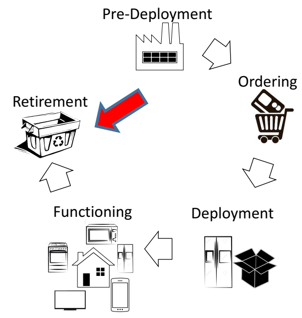
Normal network operation

1. Device A requests an operation over device B
2. B tells A the attributes required to perform the operation
3. A presents his signature as a proof it has the attributes
4. B performs the operation

A challenge-response protocol is going on behind the scenes

$$\textsc{Retirement}(\textit{user } U, \textit{device } A, \textit{device } B)$$

$$
\begin{array}{lll}
1. & A: & \{\textit{Requests retirement}\} \\
2. & B: & \textsc{unbinding}(B, U) \\
3. & B: & \text{deletes } S^{\mathrm{I}}_{B,\mathcal{C}}, S^{\mathrm{I}}_{B,\mathcal{H}}, S^{\mathrm{A}}_{B,\mathcal{H}} \text{ and } \mathbb{R}_B
\end{array}
$$



Description

Retirement is simply a special operation

1. A requests the retirement of the device B
2. B unbinds itself from its owner in the Cloud domain
3. B deletes its keys from both Cloud and Home domains

Extra Futures

Extra

- AoT enables seamlessly interdomain interactions
  - Devices from different Home domains may interoperate
- AoT supports device ownership reassignment
  - It allows a user to trade or give away one or more of his devices

$\textsc{InterDomainKeyAgreement}(\textit{device } A, \textit{device } B)$

1.	A :	$P^{\mathrm{I}}_{B,\mathcal{I}} := \textsc{Map}(id_{B,\mathcal{H}}) \cdot G^{\mathrm{I}}_{\mathcal{H}_B} + P^{\mathrm{I}}_{\mathcal{H}_B}$
2.	$A \rightarrow B$:	$\mathrm{n}_A, \mathrm{x}_A \cdot P^{\mathrm{I}}_{B,\mathcal{I}}$, inter_session_req
3.	B :	$k_{A,B} := \hat{e}(S^{\mathrm{I}}_{B,\mathcal{I}}, \mathrm{x}_A \cdot P^{\mathrm{I}}_{B,\mathcal{I}})^{\mathrm{x}_B}$
4.	B :	$\mathrm{c}_B := 0$
5.	B :	$k^i_{A,B} := \textsc{SessionKey}(k_{A,B}, \mathrm{c}_B)$
6.	B :	$P^{\mathrm{I}}_{A,\mathcal{I}} := \textsc{Map}(id_{A,\mathcal{H}}) \cdot G^{\mathrm{I}}_{\mathcal{H}_A} + P^{\mathrm{I}}_{\mathcal{H}_A}$
7.	$B \rightarrow A$:	$\mathrm{n}_B, \mathrm{x}_B \cdot P^{\mathrm{I}}_{A,\mathcal{I}}, \textsc{Mac}(\mathrm{n}_A)_{k^i_{A,B}}$
8.	A :	{*Computes the pairwise key, analogously*}
9.	$A \rightarrow B$:	$\textsc{Mac}(\mathrm{n}_B \mid \mathrm{n}_A)_{k^i_{A,B}}$
10.	$B \rightarrow A$:	inter_session_ack, $\textsc{Mac}(\mathrm{n}_A + 1)_{k^i_{A,B}}$

Device Ownership Reassignment

- Allows a user U to transfer the ownership of a device B he owns to another user V
- The protocol is similar to *Retirement* but the retired device does not delete its Cloud domain keys
- Besides, B is bound to a new user in the Cloud domain

REASSIGNMENT(*user* U, *device* A, *device* B, *user* V)

1.	$U \rightarrow C$:	TLS(B | V)
2.	$C \rightarrow V$:	TLS(pin'_B)
3.	A :	{*Requests reassignment*}
4.	B :	UNBINDING(B, U)
5.	B :	deletes $S^{\mathrm{I}}_{B,\mathcal{H}}$, $S^{\mathrm{A}}_{B,\mathcal{H}}$ and $\mathbb{R}_B$
6.	B :	displays "ownership reassigned"
7.	$U \Rrightarrow V$:	B

Evaluation

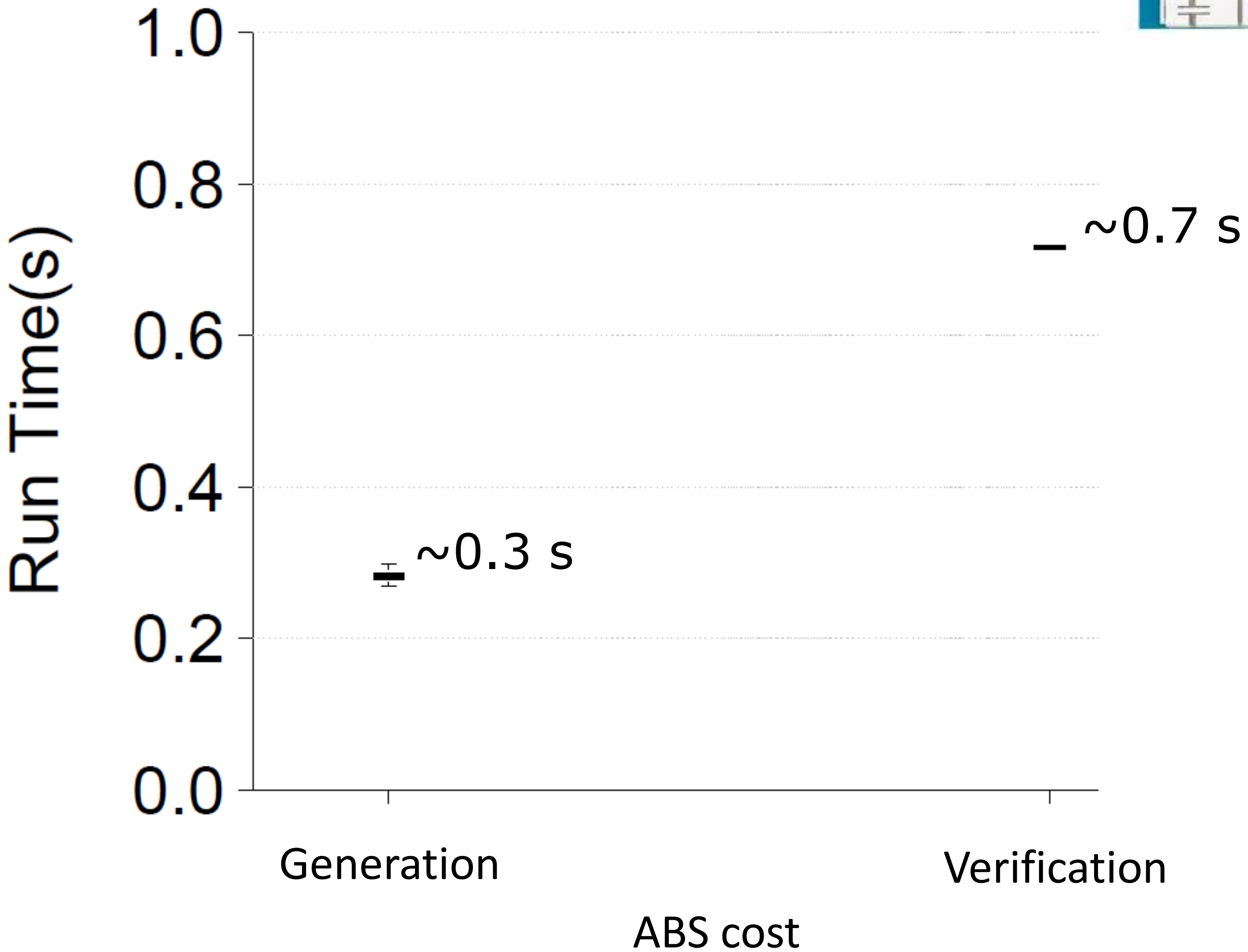
Devices

Class	Upper	Middle	Working
Device	(LG) G4	(Intel) Edison	(Arduino) Due
CPU	ARM A57	Atom	ARM M3
OS	Android 5.1	Raspbian	None
Word Size	64 bits	32 bits	32 bits
Clock	2 GHz	500 MHz	84 MHz
RAM	3 GB	1 GB	96 KB

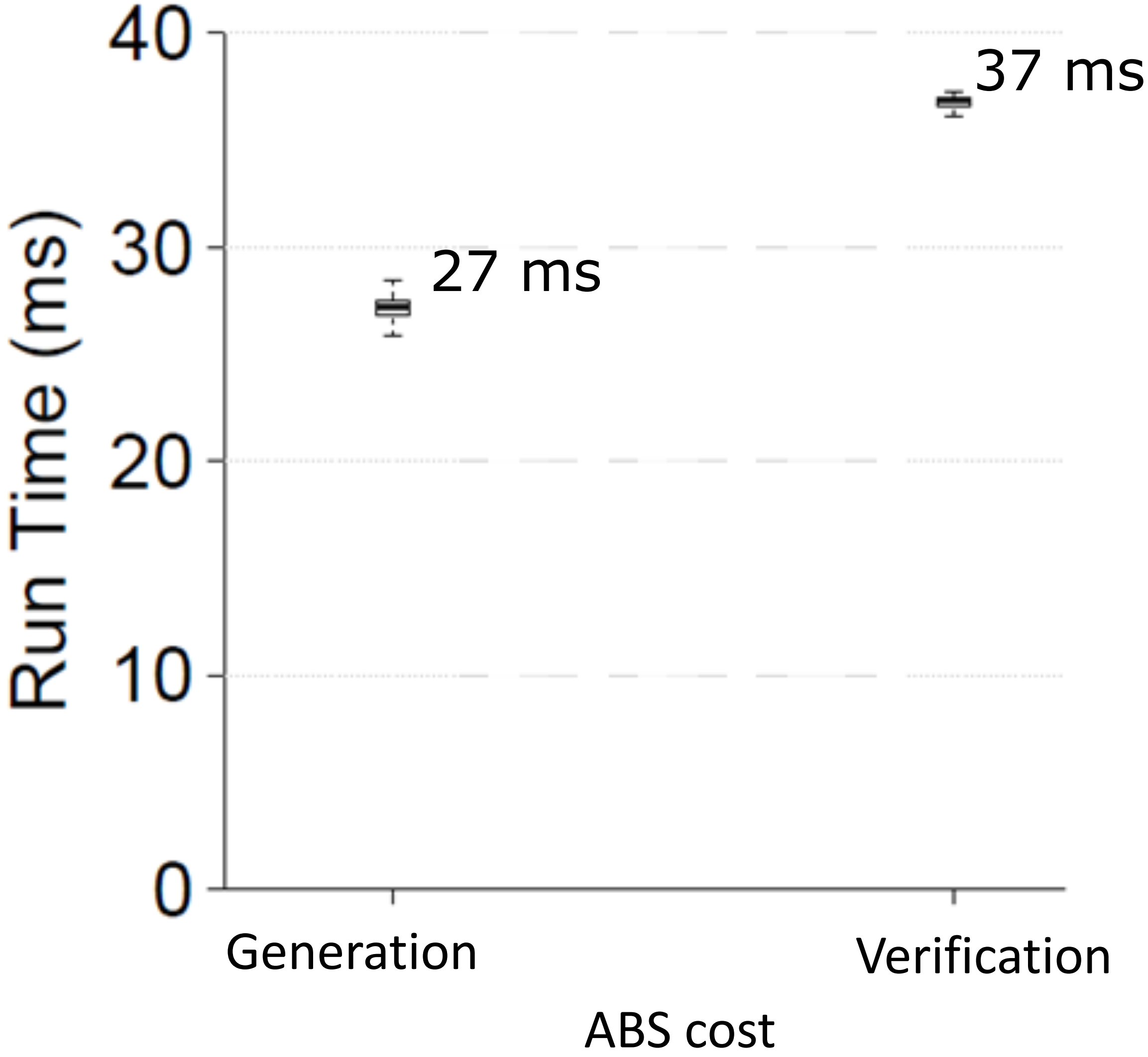
Experiments

- Predicate size of two attributes
  - X AND Y
- 128-bits of security
- Results are averages for 100 runs of each algorithm

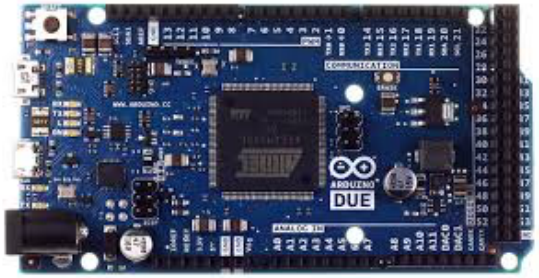
Middle Class

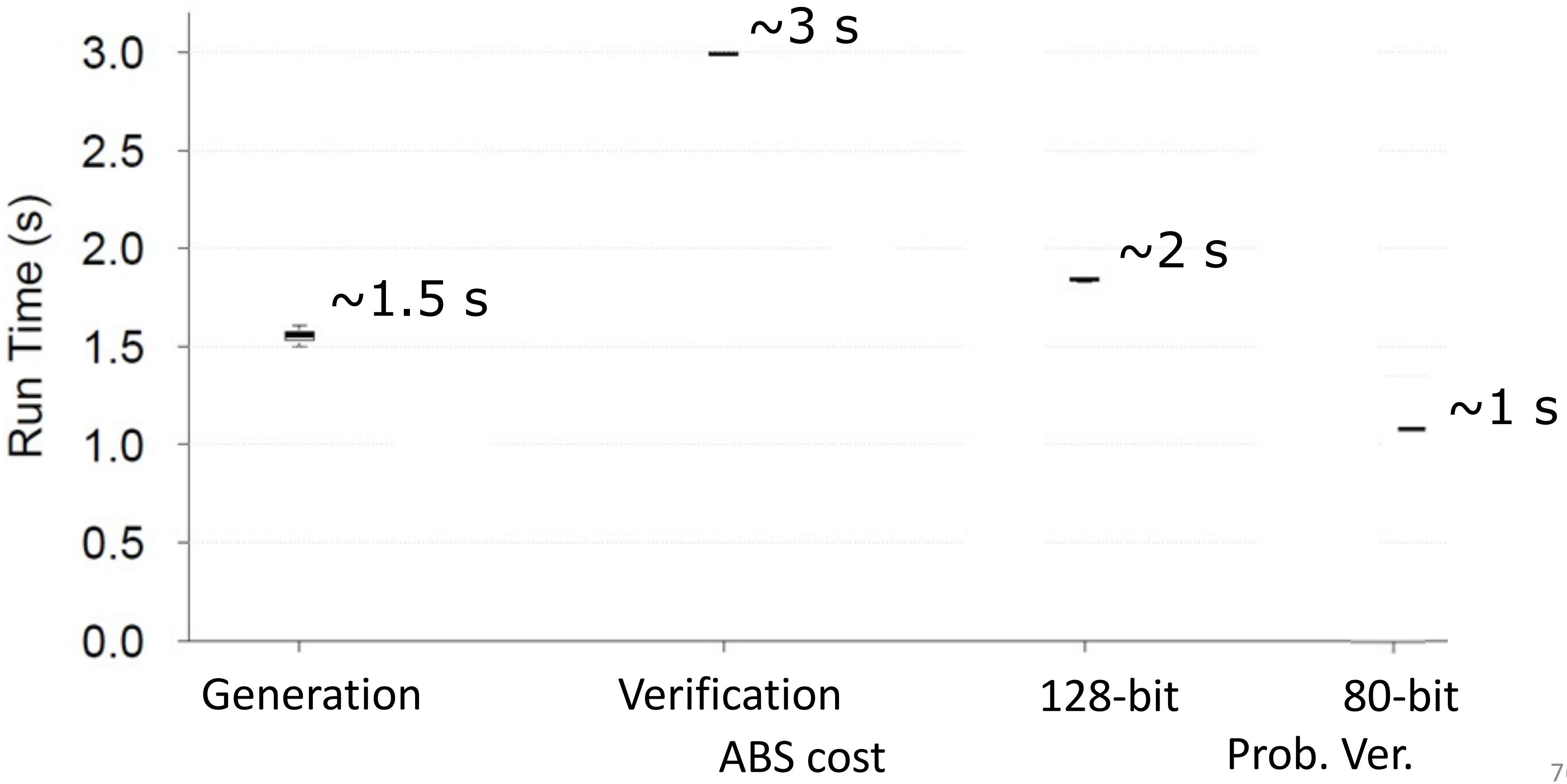


Resourceful

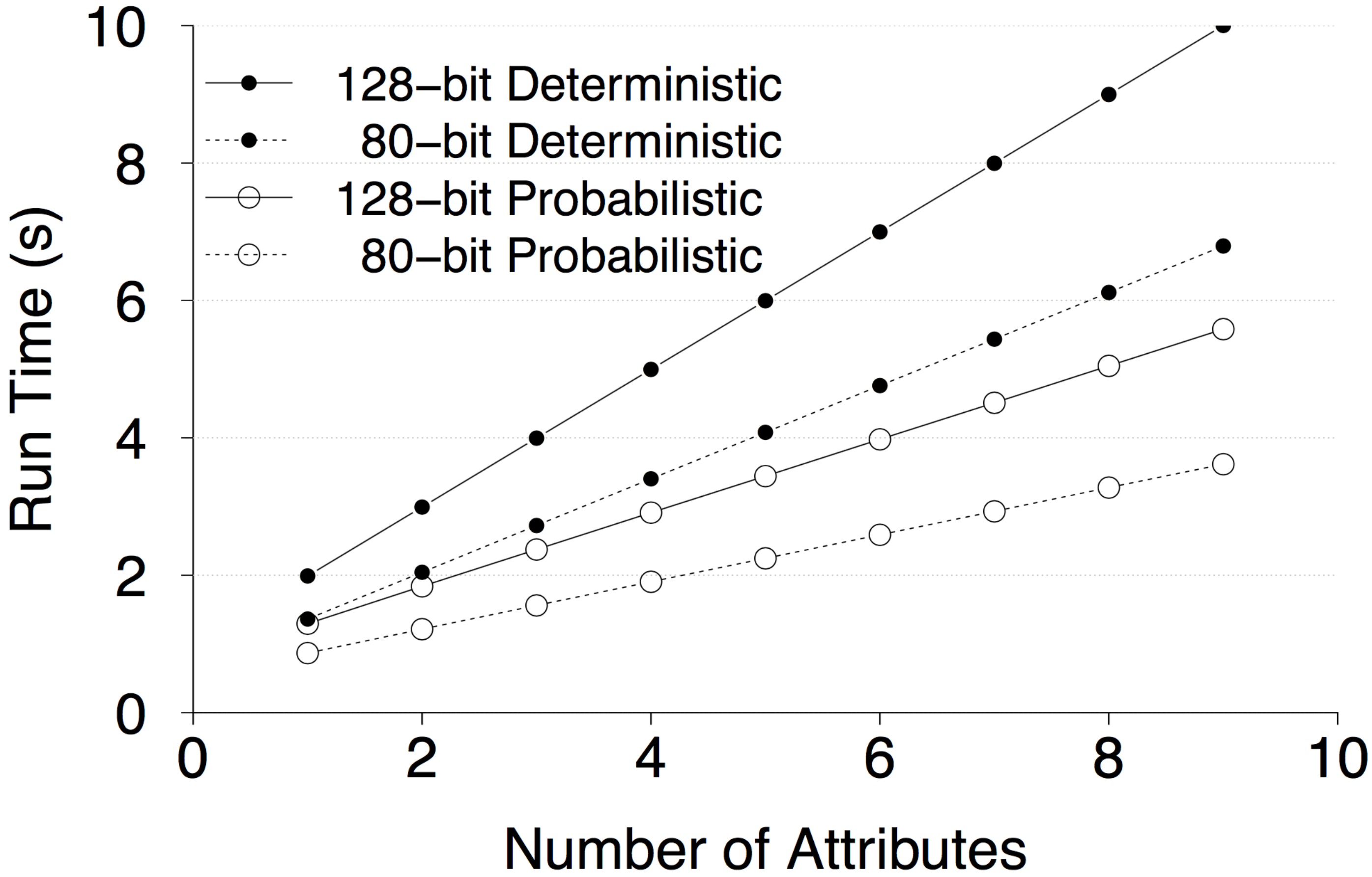


Resource-constrained





Scalability - Due

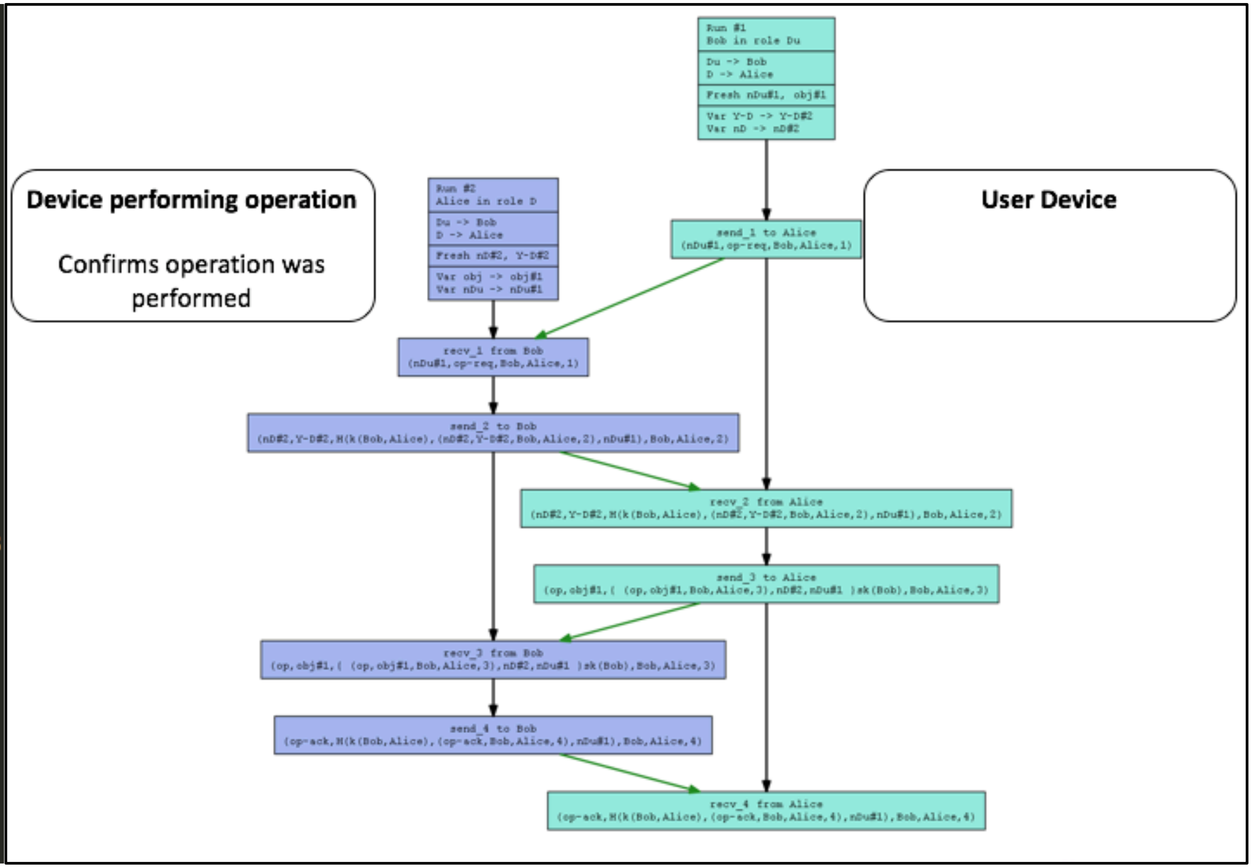


Security evaluation

- AoT provides the following security properties
  - Authentication, confidentiality, freshness, integrity, and non-repudiation
- We use a tool called Scypher to formally verify that AoT indeed provides the aforementioned properties
- It takes as input a protocol specified using the Security Protocol Description Language (SPDL)
- And then outputs the protocol flow which is used to check if the protocol follows the specification

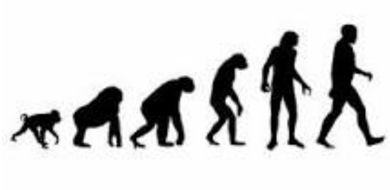
Scypher Illustration

```
//Protocol
protocol Functioning(Du,D)
{
    role Du
    {
        fresh nDu: Nonce;
        var nD   : Nonce;
        const obj: Object;
        var Y-D  : SigningPolicy;

        macro msg1 = (nD, Y-D, Du, D, 2);
        macro msg2 = (op, obj, Du, D, 3);
        macro msg3 = (op-ack, Du, D, 4);

        macro MAC-nDu    = H(k(Du,D), msg1, nDu);
        macro Sig-nD-nDu = {msg2, nD, nDu}sk(Du);
        macro MAC-nDu2   = H(k(Du,D), msg3, nDu);

        send_1(Du, D, (nDu, op-req, Du, D, 1));
        recv_2(D, Du, (nD, Y-D, MAC-nDu, Du, D, 2));
        send_3(Du, D, (op, obj, Sig-nD-nDu, Du, D, 3));
        recv_4(D, Du, (op-ack, MAC-nDu2, Du, D, 4));

        claim(Du, Alive);
        claim(Du, Niagree);
        claim(Du, Nisynch);
        claim(Du, Weakagree);
        claim(Du, Reachable);
    }

    role D{ ... }
}
```



Others

Pairing-Based Cryptography



- PBC was the missing link to enable Identity-Based Encryption and thus to enable complete IBC schemes
  - The bilinear pairing is the major crypto primitive here and its crucial property is the bilinearity

$$\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab} \ \ \forall\, P \in \mathbb{G}_1, Q \in \mathbb{G}_2 \text{ and} \, \forall\, a, b \in \mathbb{Z}_n$$

P and *Q* are points on an elliptic curve and *a* and *b* are scalars

- In AoT, PBC is used to implement both Identity-Based Cryptography and Attribute-Based Cryptography

Attribute-Based Access Control

- Traditional access control models does not scale well to IoT realistic scenarios
- This is because MAC, DAC, RBAC are all user centric and thus somewhat oblivious to context and content
- ABAC simplifies access by using policies based on attributes and therefore it scales smoother
  - ABAC is based on the observation that permissions is often attribute-related rather than identity-related

Thanks

leonardo.barbosa@dcc.ufmg.br