From:	Nguyen, Phil
To:	Ruelens Gerrit; Gerrit Ruelens (QEO LLC); Dave Thaler; Burns, Greg; Rob Smith (WINDOWS); Swinson, Ken; allseen-core@lists.allseenalliance.org
Subject:	RE: [Security 2.0] guild equivalence
Date:	Wednesday, March 18, 2015 8:46:54 AM

Hi all,

I'd like to consolidate the idea of guild equivalence and Dave's analogy of guild equivalence to the installation of a membership certificate for the guest at the service application.

How about the concept of "auto cert" for a guest?

1. The owner tells the service app that any client app with identity certificate chained up to this particular guest CA will get an automatic local cert for this security group for a valid period of time. This local cert is only valid on this service application.
2. The ACL of the security group is defined in the policy at the service app like just like any other security group.

Benefits:

1. Remove the restriction of guild equivalence to just one security group.
2. Similar to guild equivalence, the guest app does not need to install any additional certificate
3. Maybe easier to explain to a non-technical person ☺

Phil

From: Ruelens Gerrit [mailto:Gerrit.Ruelens@technicolor.com]
Sent: Tuesday, March 17, 2015 8:03 AM
To: Gerrit Ruelens (QEO LLC); Dave Thaler; Burns, Greg; Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; allseen-core@lists.allseenalliance.org
Subject: RE: [Security 2.0] guild equivalence

Hi all,

I updated the wiki page: https://wiki.allseenalliance.org/core/security_manager_description

Comments are welcome.

Kr,

Gerrit

From: Gerrit Ruelens (QEO LLC)
Sent: dinsdag 10 maart 2015 22:01
To: 'Dave Thaler'; Gerrit Ruelens (QEO LLC); Burns, Greg; Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; allseen-core@lists.allseenalliance.org
Subject: RE: [Security 2.0] guild equivalence

Hi Dave,

That is indeed what we have in mind. The starting point we have is: I have a smart phone and I install an AllSeen application on it from an app store. To what extend can I trust this app?
When I just downloaded a TV remote control application, I wouldn't mind giving it access to my TV's. But that's it. I don't want it to unlock my front door or to play with my oven.
So that why we find application manifests so important. This should help us to protect against malicious apps. Still it remains a judgment call. It you give trust to malicious app, it can and most likely will abuse it.
If you end up in such a situation, you can take away all rights from that app.

I'll update the doc and make sure this more explicit.

Kr,

Gerrit

From: Dave Thaler [mailto:dthaler@microsoft.com]
Sent: dinsdag 10 maart 2015 21:07
To: Dave Thaler; Gerrit Ruelens (QEO LLC); Burns, Greg; Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; allseen-core@lists.allseenalliance.org
Subject: RE: [Security 2.0] guild equivalence

I would find it helpful to see a description of the trust model (what entities inherently trust what other entities)
and the threat model (what you're trying to protect from whom). Currently, I'm trying to infer it, but usually
in security protocol design one tries to have written descriptions to reference.

One thing occurred to me after the email that probably you had in mind already but I didn't get it until now.
I think one threat you want to mitigate is that a malicious app cannot impersonate a well-written app.
You cannot preclude collusion between two malicious apps, but if a well-written app uses a non-shared keystore,

then you want a malicious app to not have access to the well-written app's keystore.

That is something that is easier to accomplish on sandbox-only systems like mobile phones and nigh impossible on desktop systems.   Of course a security manager may or may not know which is the case for a given app it gets a manifest about.

Dave

From: allseen-core-bounces@lists.allseenalliance.org [mailto:allseen-core-bounces@lists.allseenalliance.org] **On Behalf Of** Dave Thaler
Sent: Tuesday, March 10, 2015 11:55 AM
To: Gerrit Ruelens (QEO LLC); Burns, Greg; Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; allseen-core@lists.allseenalliance.org
Subject: Re: [Allseen-core] [Security 2.0] guild equivalence

Below

From: Gerrit Ruelens (QEO LLC) [mailto:Gerrit.Ruelens@qeo.org]
Sent: Tuesday, March 10, 2015 1:17 AM
To: Dave Thaler; Burns, Greg; Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; allseen-core@lists.allseenalliance.org
Subject: RE: [Security 2.0] guild equivalence

Hi Dave,

Thanks for reviewing and fixing typos/grammar. I replaced the references to guild by group. I'm glad you added the Windows desktop as well in the keystore example.

I'm still struggling with writing that we manage keystores.

[DT] Am I'm still struggling with any sentence saying applications are managed, since that doesn't make sense to me (and below you also say it is impossible).

The security manager does not talk directly to a keystore and technically keystores can contain multiple keypairs.   I hope that new section explains that bindi key to applications is not that obvious and that it is not always straight forward to determine who can access the key. The limitations we currently have on desktops make it impossible to link a key to an application.

[DT] I agree, and hence I don't understand how the design can possibly work.   The new text says

OS'es on smartphone do a better job at sandboxing applications. The link between key and application is stronger there.

Stronger yes, but the fundamental fact is unchanged… an app can still have a per-app keystore, or a shared keystore (unless you propose to remove this feature all together, but I've seen no such proposal), and no AllJoyn-specific enforcement is done outside of the sandbox (i.e. the app still has full ability to do whatever it wants protocol-wise).

However I still feel this is something we need to do.

[DT] I don't agree we need to do something that might be impossible.   Once there exists an explanation of how treating apps themselves as security principals could work in a secure fashion, we could discuss whether we need to do it. Until then, I maintain it is not needed.

We should be able to manage application/device/user triplets.

[DT] Just because we might like to do something doesn't mean it is technically feasible. I do believe that managing security principals (e.g. public keys stored in keystores) is absolutely feasible. But I do not believe it is possible in the AllJoyn architecture to manage "applications".

So I give rights to an application running on a well-known device under a well-known users. The same application running as different user on the same device is a different application from security manager perspective.

[DT] It uses a different keystore, which is what's important.

The same application on a different device, again different app.

[DT] If the user is the same, then I disagree it's ***necessarily*** a different identity.   That's up to whether the keystore for that user+app is shared/replicated across the devices or not.   Both are valid implementations.   (On Windows, these two concepts are that appdata can be Local or Roaming.)

This is very understandable for real users. Granting rights to a user on a desktop is asking for attacks. I browse to a website. The website runs an alljoyn enabled app in my browser and that app can do everything my desktop user can.

[DT] If an untrusted app in a browser can do everything your desktop user can, then I agree you have an insecure system.   But don't assume that all systems are as insecure as that.
Specifically, if an app in your browser can access the desktop user's shared keystore, then I would say you have a security vulnerability that must be fixed. I'm not aware of any such vulnerability on any device I use.

Kr,

Gerrit

From: Dave Thaler [mailto:dthaler@microsoft.com]
Sent: dinsdag 10 maart 2015 1:35
To: Gerrit Ruelens (QEO LLC); Burns, Greg; Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; allseen-core@lists.allseenalliance.org
Subject: RE: [Security 2.0] guild equivalence

I fixed a bunch of trivial typos/grammar issues on the wiki page, but it's still consistent in use of "guild" vs "security group".
And it still refers to apps instead of keystores, and so different sections use "application" with contradicting meanings.

From: Gerrit Ruelens (QEO LLC) [mailto:Gerrit.Ruelens@qeo.org]
Sent: Monday, March 9, 2015 4:18 AM
To: Burns, Greg; Dave Thaler; Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; Gerrit Ruelens (QEO LLC); allseen-core@lists.allseenalliance.org
Subject: RE: [Security 2.0] guild equivalence

Hi All,

After Dave's feedback, comments in this mail thread and last Fridays discussion I updated the text from the original mail and put it on the wiki:
https://wiki.allseenalliance.org/core/security_manager_description.
I marked the page as draft version – to be approved. I hope we can reach consensus on it and make it part of the HLD.

Main conclusion on the thread for now: we drop support for User equivalence and replace it by an Admin security group.
I updated ASACORE-1409: I added a snippet from this mail thread.

I added a use case for guild equivalence to explain the need and benefit compared to delegation.

Kr,

Gerrit

Guild Equivalence

Guild equivalence tries to address the same problem as delegation, but takes another approach to solve it. With delegation you give a certificate to an application. With certificate the application can proof if is allowed access to a guild. With guild equivalence we do the other we around. We define a policy on our managed applications allowing applications from a different security manager to get access. This would be as if we would pre-install the delegated membership certificate on all our managed applications. So when the peer comes around he doesn't need to send the proof, the application already has it. Since policy comes from a trusted source, we don't need to distribute certificates, we can define a more compact ACL.

In practice the security manager defines a guild equivalence ACL for all applications that need to interact with the applications of the peer security manager. This ACL maps applications of that security manager into a local guild. Those applications don't need a membership of that guild, but they need to prove that they are owned with an identity or membership certificate. If they can the policy for the local guild will be applied on them.

The guild equivalence rule allows one-way authentication. For a mutual authenticated session the peer security manager should install guild equivalence for your security managers' public key. The main advantage of guild equivalence is that these rules do not require any knowledge of the guilds owned by the peer security manager. In case of delegation the receiver of membership certs needs to know what the scope is of that guild. Guild equivalence is not transitive. If A declares guild equivalence rules for B and C, that does not mean that a guild equivalence exists between B and C. They have to explicitly install it themselves.

Guild equivalence works very nice in a setup where every security manager represents a single user. Every person takes up the role of end-user as well and administrator. Then guild equivalence easily allows you to express ACL to towards others user. Users with similar right can be mapped into the same guild.

As example use case: Suppose we have a real-estate agent. When showing a house to clients he'd like to show-off the AllSeen enabled home automation system. This can be achieved with both delegation and guild equivalence. The advantage for guild equivalence here is that if he potentially needs to show 100 homes, he can do it based on 1 certificate instead of 100 for the delegation scenario. There is less risk for information disclosure. If someone could get hold of the 100 certificates, then he can learn who the home sellers are. In the guild equivalence case, the sellers public keys are in the policy of the agents app and policy is never shared.

From: Burns, Greg [mailto:gburns@qce.qualcomm.com]
Sent: vrijdag 6 maart 2015 21:46
To: Dave Thaler; Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; Gerrit Ruelens (QEO LLC); allseen-core@lists.allseenalliance.org
Subject: RE: [Security 2.0] guild equivalence

I agree, but usability is going to be a big concern. If we can't make it lot easier than administration of OS groups we are in big trouble.

From: Dave Thaler [mailto:dthaler@microsoft.com]
Sent: Friday, March 06, 2015 12:11 PM
To: Burns, Greg; Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; Gerrit Ruelens (QEO LLC); allseen-core@lists.allseenalliance.org

Subject: RE: [Security 2.0] guild equivalence

I would argue that introducing new concepts that have no usability analysis showing they're more understandable by regular people than existing concepts,
is by default even less likely to be understandable by regular people.

From: Burns, Greg [mailto:gburns@qce.qualcomm.com]
Sent: Friday, March 6, 2015 11:38 AM
To: Dave Thaler; Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; Gerrit Ruelens (QEO LLC); allseen-core@lists.allseenalliance.org
Subject: RE: [Security 2.0] guild equivalence

Dave,
Should this scare me a little?

"in ways that are already well known and understood and expected by people who administer OS groups. "

We are designing this system so it can be used by regular people not system administrators.

Greg.

From: allseen-core-bounces@lists.allseenalliance.org [mailto:allseen-core-bounces@lists.allseenalliance.org] **On Behalf Of** Dave Thaler
Sent: Friday, March 06, 2015 11:05 AM
To: Nguyen, Phil; Rob Smith (WINDOWS); Swinson, Ken; Gerrit Ruelens (QEO LLC); allseen-core@lists.allseenalliance.org
Subject: Re: [Allseen-core] [Security 2.0] guild equivalence

Completely agree with the notion of an admin group and removing the concept of user equivalence (but not ownership).

More comments inline below with [DT]

From: Phil T. Nguyen [mailto:philn@qce.qualcomm.com]
Sent: Wednesday, March 4, 2015 3:53 PM
To: Rob Smith (WINDOWS); Swinson, Ken; Gerrit Ruelens (QEO LLC); Dave Thaler; Nguyen, Phil; allseen-core@lists.allseenalliance.org
Subject: RE: [Security 2.0] guild equivalence

Using an admin group we can remove the concept ownership. During claim, the claimer provides the CA and the admin group (guild authority public key and guid).

Phil
At 03:26 PM 3/4/2015, Rob Smith (WINDOWS) wrote:

I agree with the problem user equivalence is trying to solve, but what about a different solution: instead of allowing devices to be owned by a specific user, why not have an administrators group/guild like most other platforms? This way you can grant additional people administrator privileges just by added them to the admin guild, but you don't have to give them full rights to appear as you. This also means we wouldn't need to implement user equivalence.

From: allseen-core-bounces@lists.allseenalliance.org [mailto:allseen-core-bounces@lists.allseenalliance.org] **On Behalf Of** Swinson, Ken
Sent: Wednesday, March 4, 2015 1:28 PM
To: Gerrit Ruelens (QEO LLC); Dave Thaler; Nguyen, Phil; allseen-core@lists.allseenalliance.org
Subject: Re: [Allseen-core] [Security 2.0] guild equivalence

User equivalence came about from a desire to address a few issues that may arise as devices enter a proximal network:
- That ownership and administration is not tied to a single person.
- That an owner/administrator does not use a single device (I use both a tablet and phone and expect similar capabilities regardless of which is in hand)
- That permissions issued by one owner should as respected as permissions issued by another owner.

The current concept of user equivalence attempts to address the first two.

Assuming that an owner and a security manager will each be represented by unique key pairs, then providing access to a shared security manager would not address the user equivalence use case. As I understand it, administration of the device (managing owners, policies, etc..) requires that the peer be the device administrator which cannot be granted through an ACL expressed in a policy. The work for JIRA ticket 1409, "Support for signing keys and communication keys" will probably change this though.

Regarding guild equivalence:

If the current thought is to express Guild Equivalence through policy then the description below seems to be expanding a guild to be identified by only the public key of the security manager if no match is found for the GUID; effectively a wild card match on the guild's GUID.

Ken

From: allseen-core-bounces@lists.allseenalliance.org [mailto:allseen-core-bounces@lists.allseenalliance.org] **On Behalf Of** Gerrit Ruelens (QEO LLC)
Sent: Wednesday, March 04, 2015 8:21 AM
To: Dave Thaler; Nguyen, Phil; allseen-core@lists.allseenalliance.org
Subject: [Allseen-core] [Security 2.0] guild equivalence

Hi Phil, Dave,

I put a comparison on delegation, guild equivalence and user equivalence as well with some assumptions we have on the security manager.
I hope this clarifies things a bit more.

Kr,

Gerrit

Delegation vs Guild Equivalence vs User equivalence
The security manager
A security manager is an entity in charge of managing AlllSeen security 2.0 applications. Managing means providing certificates and ACL's to the applications it manages. The security manager is operated by an Administrator. The security manager has following tasks:

- It offers certificate authority (CA) functionality: It provides certificates, so it must be able to generate and sign them

- UI: The administrator needs to interact with the security manager in order to make configuration changes

[DT] Since there was some discussion of an always-on security manager, I think it's important to decouple the implementation of the app with the UI (which may or may not be on all the time, and which might even be a web form interface in a browser) from the security manager. The security manager should be able to either have a UI or expose interfaces/APIs/protocols for a UI app to use to talk to the security manager.
Below you say there's four functional blocks which can run in different applications or on different hosts, but defining the security manager as "an entity" at the start makes it sound too much like a single unit, which it's not (necessarily).

- Configuration storage: As managed applications are not online all the time, the security manager should keep track of the state and best effort basis update application when they come on-line.

[DT] As discussed during the Tuesday meeting, the concept of "application" is hard to understand here. Someone suggested to discuss in terms of keystore, which would make far more sense to me, and then one can discuss the relationship between applications and keystores.

- AllJoyn Agent (security manager agent): Configuration updates are sent using AllJoyn as communication protocol. As AllJoyn is proximal protocol. An Agent local to the applications is needed.

Following assumptions are made:

- The four functional blocks of the security manager can be combined into a single application, but it should be possible to run them in different applications or even on different hosts.
- A security manager can have multiple security manager agents
- The security manager topology is transparent for AllJoyn applications
- A security manager is identified by the public key of its CA. We call this THE key of the security manager.

[DT] The last bullet above is confusing, especially in light of my previous comment. The "key" of the security manager is whatever public key the code is currently running as. The same "app" (executable on a host) might be run by different users at different times, and it would use a per-user keystore and so the public key of the "app" is based on who is running it.
The Alliance envisions multiple implementations of security managers and does not provide implementation specifications. The alliance does specify how security managers need to interact with AllJoyn security 2.0 applications.

- org.allseen.Security.PermissionMgmt.Notification: a sessionless signal send by the applications. This can be used to discover applications.
- org.allseen.Security.PermissionMgmt : For changing the configuration of applications

[DT] The IRB provided feedback on the problems with the above, which I need not repeat here.
What does the security manage?

- Applications: more in detail the configuration of the application, identity, guild memberships and ACLs

    - As a side note: a single-function device (e.g. a temperature sensor) is considered an Application.. Every app on a smartphone is considered as an app on its own. The build-in firmware of a smart TV is also considered as a single app. Applications installed on top of the firmware of the TV are separates app and should be claimed separately.

[DT] Already discussed problems with the above description.

- Guilds/Security groups: guild allow grouping of applications. A guild is unique defined by GUID and the public key of Security manager. Applications can become member of guild when they are issued a membership certificate for that guild

[DT] It allows grouping of public keys, not applications per se. Using the term application here is incorrect, from my understanding of the Tuesday discussion.

- Identities: Identities are used to define the users of application. Users can map to physical persons, but they could also a more conceptual meaning. An identity is represented by a GUID and the public key of the defining security manager.

Applications can act on behalf of a user when they receive an identity certificate for that user. Applications should only have 1 identity certificate

Inter security manager interaction
When applications interact with each other, they check if the interaction is allowed by their security manager.

[DT] "is allowed by their security manager" sounds like a run-time requirement that the security manager has to be online at the time of access, which isn't the case (as I understand the doc anyway). It would be more correct, if I understand right, to say something like "is allowed by its policies, as previously set by their security manager".

In practice its peer must present a certificate (chain) signed by its security manager public key. Meaning that with the basic features we created silos, you can only talk to applications owned by your own security manager. In practice application managed by different security manager need to interact with each other. The current HLD provides 3 ways to do this: Delegation, Guild equivalence and user equivalence.

Delegation
Use case: I'm the administrator of my home eco system. I claim appliances in the home and provide them with configuration. I as administrator am the only person having access to the security manager. When my kids want to get access to an appliance, then they have to ask me to get approval for each application they want to use. This is not workable. With delegation my security manager gives a membership certificate with delegation rights to the security manager of each my kids. With this certificate they can delegate these rights to their applications. They can make their applications part of my guild. Even though my kids did not interact directly with each other, with these delegated certificates they interact with each other in the scope of this guild.
What are the limitations of delegation?

- You can only authenticate members of the guild. Mutual authenticated request can only be done between members of the guild.
    - My kids get Remote-control rights for the TV by giving them a membership certificate with delegation rights for my TV Guild. Their remote control applications become member of the TV guild. If a give my TV a policy for the TV guild, then the TV will allow the request from the RC apps of my kids. This requires my kids to define an any-peer policy for TV operations for their apps. This is ok for TVs RC operations. If mutual authentication is required, the TV must become member of the TV guild as well.
    - For a chat use case you need to know who is sending a message and to whom you're sending messages to. So mutual auth is required and all participants have to be in that guild.
- As policy is defined on guild level, it is impossible to differentiate between kids and parents. It would require separate guilds to achieve this.
    - If we have a TV with remote control and Parental control features and chat apps, then we probably need 3 guilds to solve this. A Chat guild, a TV RC guild and TV parental control guild.

[DT] I find the wording of the 2nd bullet ("it is impossible to…") confusing. As I understand it, you can have as many guilds as you want, they're just security groups. So it's not impossible at all, it's normal/typical and the last sentence above explains how.

This works fine for small set-ups but becomes quite complex for larger configurations. It might involve a large number of guilds to get the configuration as expected.

[DT] I don't follow why it would involve a large number of guilds. In a household with parents and children, there might only be two guilds: one for parents, and one for kids. I don't see any explanation of why one would need more for some common scenario.

When you remove or even change the meaning of a guild with delegated members, then you need to notify all involved security managers so they can update their applications as well

Guild Equivalence
Guild equivalence tries to address the same problem as delegation, but takes another approach to solve it. Rather than giving a membership certificate for a guild, the security manager defines a guild equivalence ACL for all applications that need to interact with the applications of the peer security manager. This ACL maps applications of that security manager into a local guild. Those applications don't need a membership of that guild, but they need to prove that they are owned with an identity or membership certificate.
The guild equivalence rule allows one-way authentication. For a mutual authenticated session the peer security manager should install guild equivalence for your security managers' public key.
The main advantage of guild equivalence is that these rules do not require any knowledge of the guilds owned by the peer security manager. In case of delegation the receiver of membership certs needs to know what the scope is of that guild.
Guild equivalence is not transitive. If A declares guild equivalence rules for B and C, that does not mean that a guild equivalence exists between B and C. They have to explicitly install it themselves.

[DT] And it's not symmetric either.

Guild equivalence works very nice in a setup where every security manager represents a single user. Every person takes up the role of end-user as well and administrator. Then guild equivalence easily allows you to express ACL to towards others user. Users with similar right can be mapped into the same guild.
Note: the HLD still speaks of Guild certificates, while latest consensus is to express guild equivalence inside the policy

[DT] I'm not yet convinced that guild equivalence is important, given that you can do the same thing with delegation and other guilds, in ways that are already well known and understood and expected by people who administer OS groups. Expecting people to understand a new foreign concept means it'll be far less useful, I'd predict, just extra complexity that's rarely used.

User Equivalence

The use case for user equivalence is husband and wife co-managing their home with equal rights. A specific type of certificate would express these rights. This would imply that both husband and wife each have a separate security manager and that both give the other a User equivalence certificate. Such a certificate grants them full access to the applications of the other.

This is certainly a valid use case, but can't we cover it by providing them access to the same security manager? This would remove complexity from the client software.

[DT] As others have noted, this sounds like unnecessary complexity. Just using security groups (guilds) should be sufficient, just like OS's have an administrators security group for the same reason. That concept is well understood, and much less confusing than expecting people to understand a new foreign concept.

Summary

User equivalence is for having to equal managers with full access to all applications. Guild equivalence also has equal 2 administrators, but each administrator has full control over what the other can do. Both user and guild equivalence are not transitive. Delegation has one Administrator who has more rights he can include and exclude others. The others can choose not join. Delegation allows a form of transitivity (a friend of the boss is my friend as well).