ALLSEEN ALLIANCE

Core Working Group: Security 2.0

July 20, 2015

The contents of this document, and the interfaces described, and all the information herein, are the result of collaborative discussions by the Core Working Group. This summary documents the final consensus of the team.

Reminder:

This call is being recorded

Antitrust Compliance Notice

- AllSeen Alliance meetings involve participation by industry competitors, and it is the intention of AllSeen Alliance to conduct all of its activities in accordance with applicable antitrust and competition laws. It is therefore extremely important that attendees adhere to meeting agendas, and be aware of and not participate in any activities that are prohibited under applicable US state, federal or foreign antitrust and competition laws.

- Examples of types of actions that are prohibited at AllSeen Alliance meetings and in connection with AllSeen Alliance activities are described in the AllSeen Alliance Antitrust Policy. If you have questions about these matters, please contact your company counsel, or if you are a member of AllSeen Alliance, feel free to contact Lee Gesmer or Andrew Updegrove, of the firm of Gesmer Updegrove LLP, which provides legal counsel to AllSeen Alliance.

Agenda

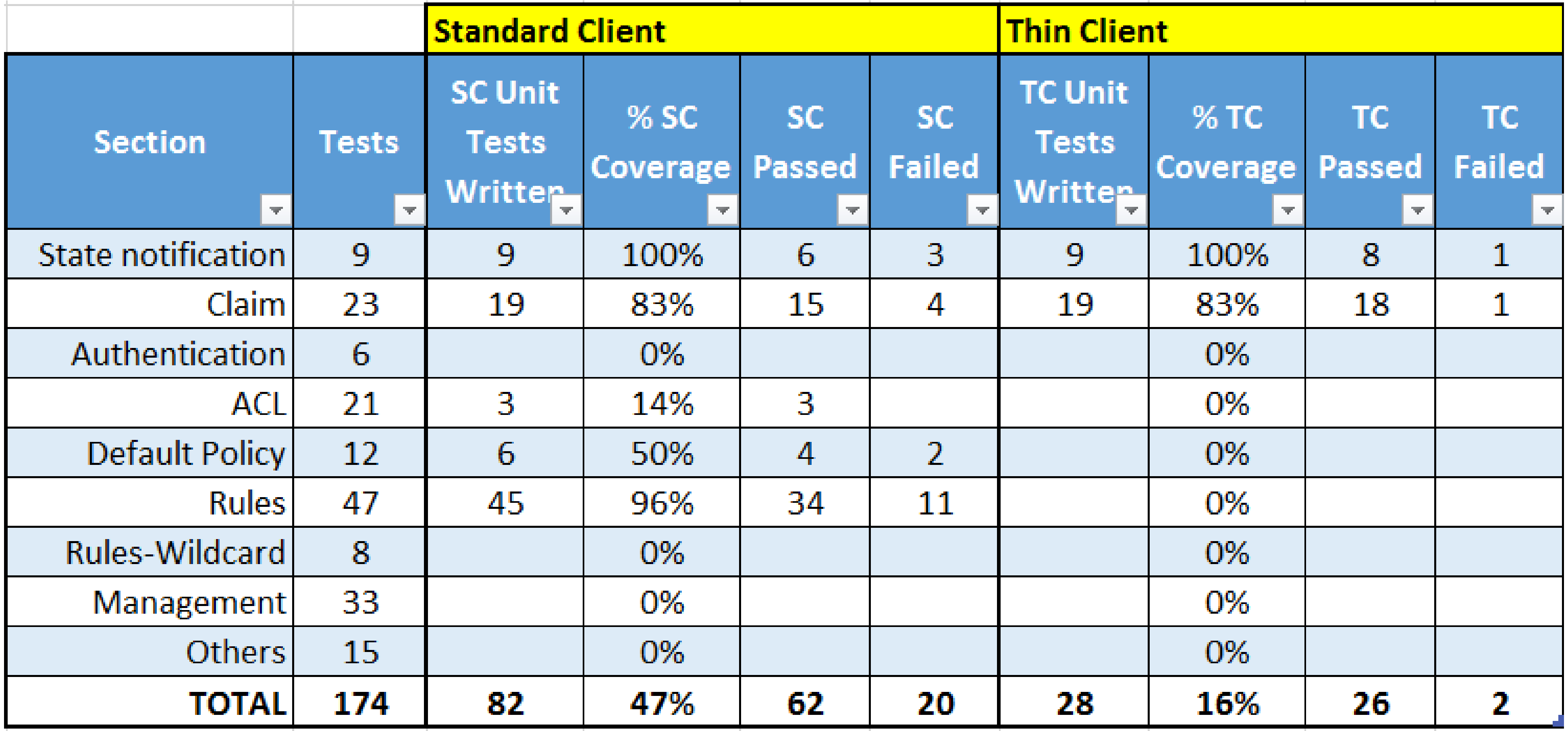
1. Schedule
2. Bindings
3. Triage
4. HLD updates
5. Discussion
6. Action items

Schedule

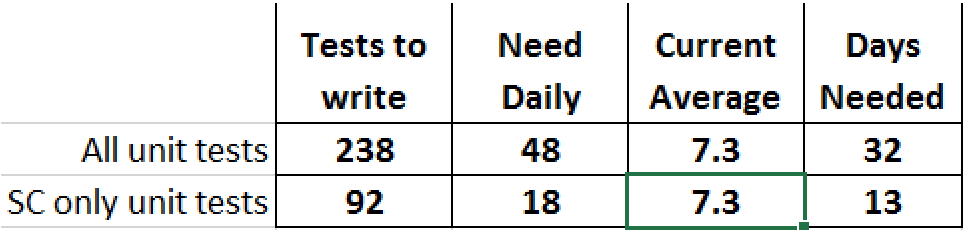
Schedule

- Dashboard: https://jira.allseenalliance.org/secure/Dashboard.jspa?selectPageId=10903
- Milestones
  - Merged with master – July 24 (This week)
    - All feature testing complete
    - Only regression testing & bug fixing remains after this milestone
    - Will merge this week
  - 15.09 release – September 30

Test Case Coverage



		Standard Client				Thin Client			
Section	Tests	SC Unit Tests Written	% SC Coverage	SC Passed	SC Failed	TC Unit Tests Written	% TC Coverage	TC Passed	TC Failed
State notification	9	9	100%	6	3	9	100%	8	1
Claim	23	19	83%	15	4	19	83%	18	1
Authentication	6		0%				0%		
ACL	21	3	14%	3			0%		
Default Policy	12	6	50%	4	2		0%		
Rules	47	45	96%	34	11		0%		
Rules-Wildcard	8		0%				0%		
Management	33		0%				0%		
Others	15		0%				0%		
TOTAL	**174**	**82**	**47%**	**62**	**20**	**28**	**16%**	**26**	**2**



	Tests to write	Need Daily	Current Average	Days Needed
All unit tests	**238**	**48**	**7.3**	**32**
SC only unit tests	**92**	**18**	**7.3**	**13**

Bindings

Bindings

- Core implementation
  - SC: C++
    - Gavin (MSFT) to verify internally if they will handle the C binding for Sec 2
      - Will have an update by Thursday’s Core WG call
  - TC: C

Triage

Triage

- Review unassigned tickets
- https://jira.allseenalliance.org/issues/?filter=11142

HLD updates

HLD Updates

- MSFT
  - Update for the membership privacy leakage change
- QCE
  - https://git.allseenalliance.org/gerrit/#/c/4612/ “Simplified ACL rule matching”
    - Default policy updated and it is in the alliance gerrit review.
  - Identity delegation & certificate chain validation & EKU validation
    - Status: Change for this has a -1 from Kevin Kane with several comments
  - ASACORE-2194: Membership certificate definition is not correct
    - Included with “Identity delegation & certificate chain validation & EKU validation”
    - Status: Completing QCE internal reviews
  - ASACORE-2192: Certificate authority installed during claim is not manageable.
    - Status: Pending – requires changes to 22 diagrams.

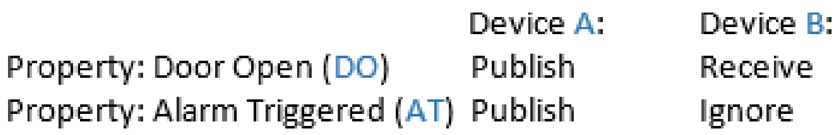
Discussion

Discussion

- Application Manifests
  - Should an application manifest include rules for Security 2.0 management interfaces?
  - There are pros and cons on both sides of the argument.
  - Yes..
    - There are no hard-coded rules for any interface, everything acts the same.
    - It is most flexible and security managers can allow/deny specific parts of the management interfaces.
    - Although, if used incorrectly, this could make an application unusable and require reset.
  - No..
    - Memory saving, avoid adding the same rules to every manifest.
  - Also, should manifest and policy include the version of each interface rule?
  - How should manifest and policy be interpreted if an interface gets upgraded/modified?

Discussion

- Interaction of GetAllProperties() and local properties
  - In the current implementation there is no easy way to enforce local trust policies with respect to individual properties that are fetched using GetAllProperties(). Our proposal is to document the GetAllProperties() API to indicate that the contents that are returned may not all be trusted, and if an application cares about specific properties it should check them individually. In a future release we should investigate the possibility of adding a new API that allows getting a list of properties: this will allow the caller to check each individually, and only fetch the values that are trusted.
  - An example to illustrate (I know these are not the terms used in the HLD but I find them confusing and will likely get them wrong – should be clear what the intent is – if not let me know):



	Device A:	Device B:
Property: Door Open (DO)	Publish	Receive
Property: Alarm Triggered (AT)	Publish	Ignore

  - When device B calls GetProperty(A, AT) the call will fail as on the calling side (B) it sees that it's local policy indicates that it isn't allowed to read property DO from A (presumably because it doesn't trust A to indicate that an alarm has been triggered). GetProperty(A, DO) will succeed because B is allowed to get that information from A.
  - What to do in the case of GetAllProperties(A)? Caller has a mix of permissions for properties on A, and unless B indicated in its manifest to A that AT is not readable (for lack of a better term) B will simply return both AT and DO. Unfortunately because of the nature of the APIs it is not possible at the caller side (B) to filter out the contents to remove AT.

Action Items

Action Items

- …

Thank You

Follow Us On

- For more information on AllSeen Alliance, visit us at: allseenalliance.org & allseenalliance.org/news/blogs