

EAT in MAA (Microsoft Azure Attestation)

CCC Attestation SIG

(6/21/2022, 7/5/2022, 7/19/2022)

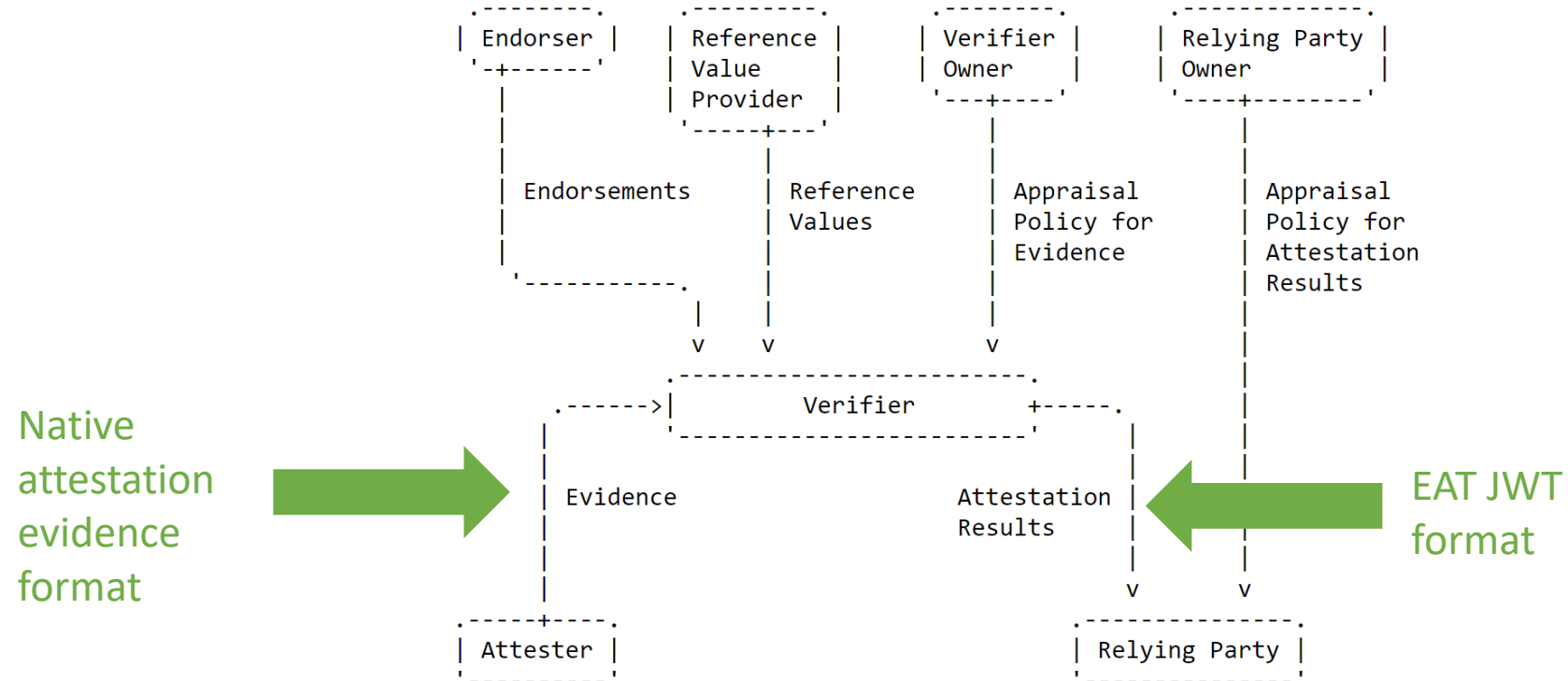
Agenda

- MAA Overview
- Registered and Unregistered Claim Names
- Claim Source
 - MAA service
 - Attestation evidence
 - TEE runtime
 - TEE initialization time
 - MAA policy
- Challenges
- What's Next?

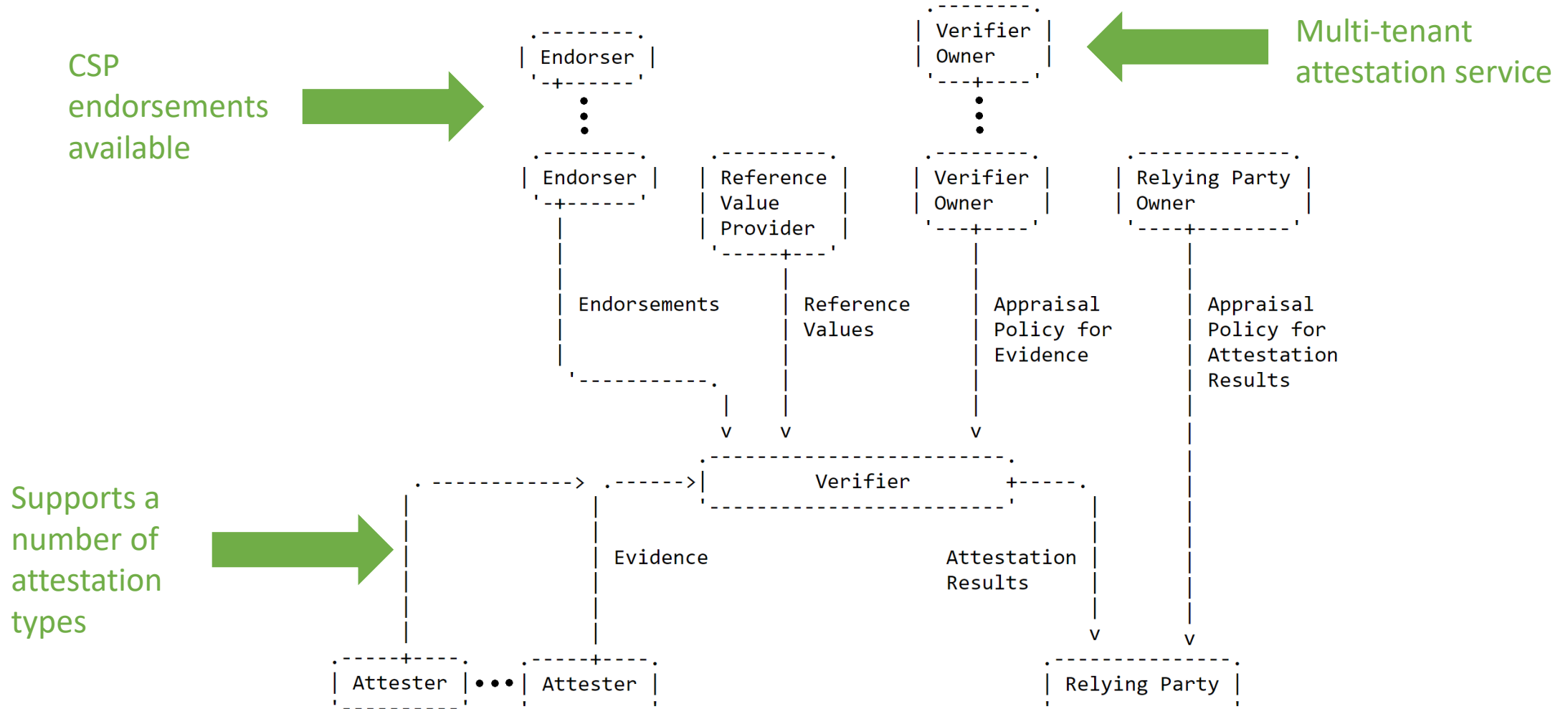
MAA Overview

- GA in January 2021
- Multi-tenant verification service
 - Microsoft operates a single “instance” with a simple policy
 - Any Azure customer can create their own “instance” to specific their own policy
- Supports a growing number of attestation types and scenarios
 - SGX
 - SEV-SNP
 - TPM
 - Virtualization Based Secure (VBS) Enclaves
 - “Trusted Launch” virtual machines (built on TPM)
 - “Confidential” virtual machines (built on SEV-SNP)
- Provably runs within a trusted execution environment (currently SGX)

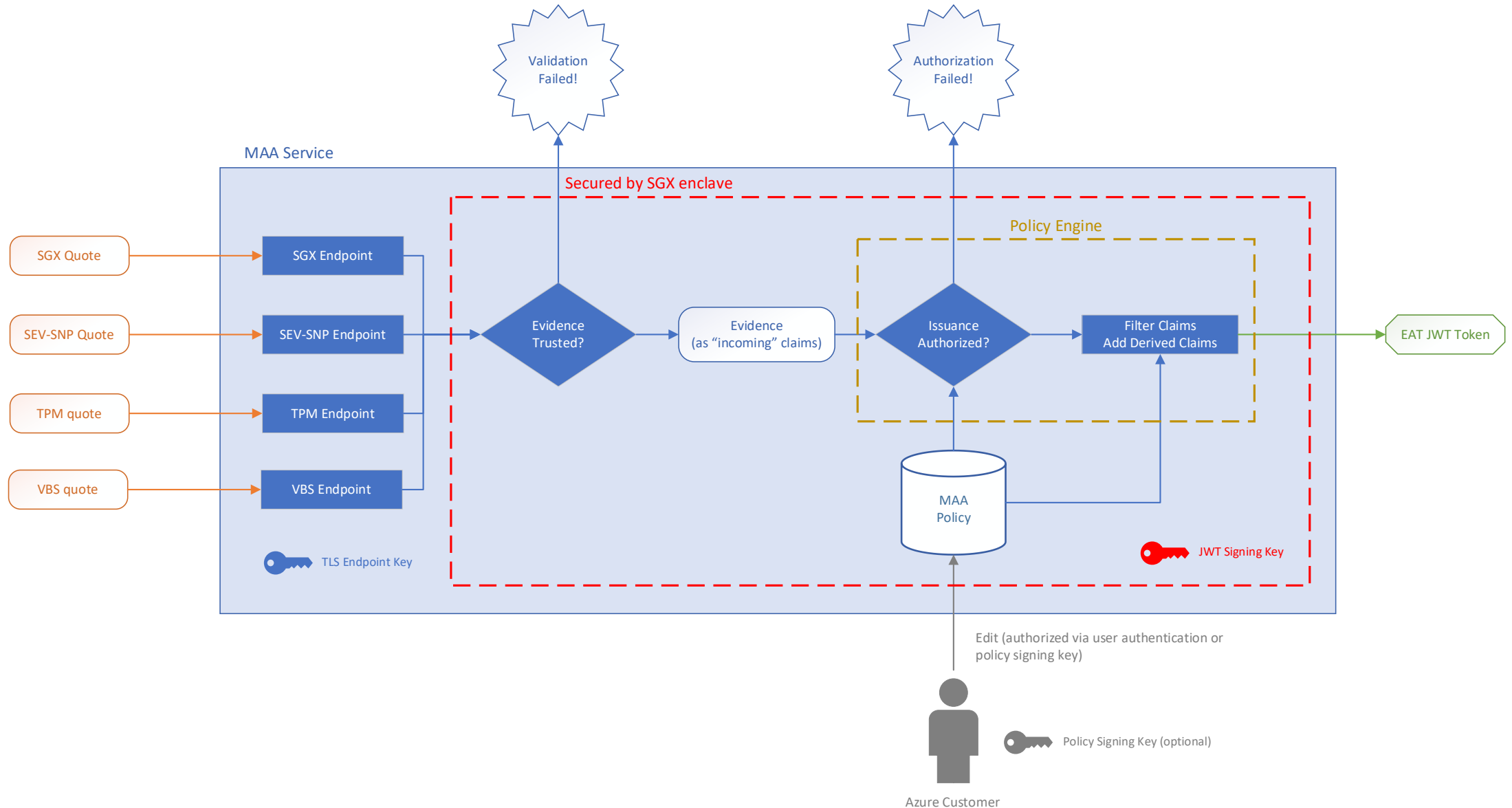
MAA and RATS Architecture



MAA and RATS Architecture



MAA Attestation Logic Flow



MAA Sample Policy

```
version= 1.0;

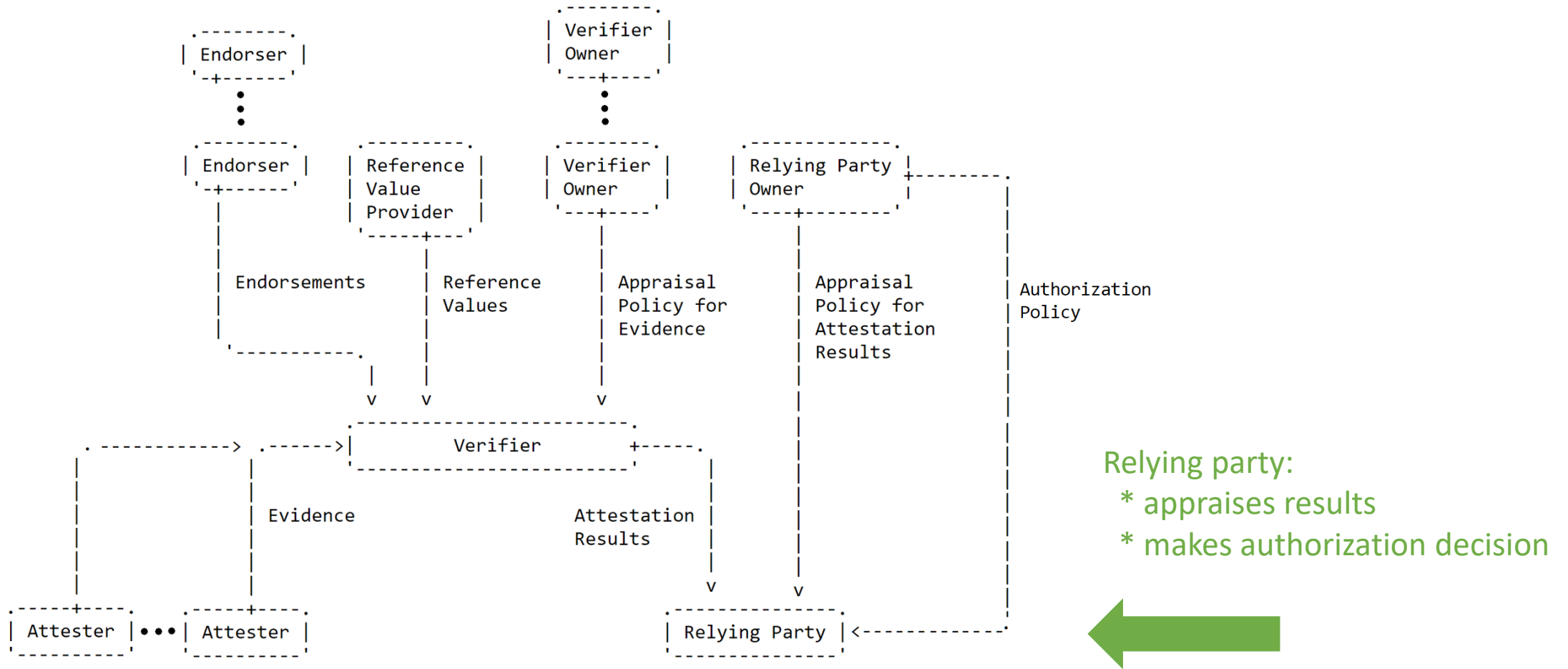
authorizationrules
{
    // Deny JWT token issuance if MRSIGNER does not match expectations
    [type=="x-ms-sgx-mrsigner", value!="61b8f05efb4e3b259f3b1bba6b89a1a16784bd4b3b172d00ba22a36c7131d2d5"]
    => deny();

    // Otherwise permit JWT token issuance
    => permit();
};

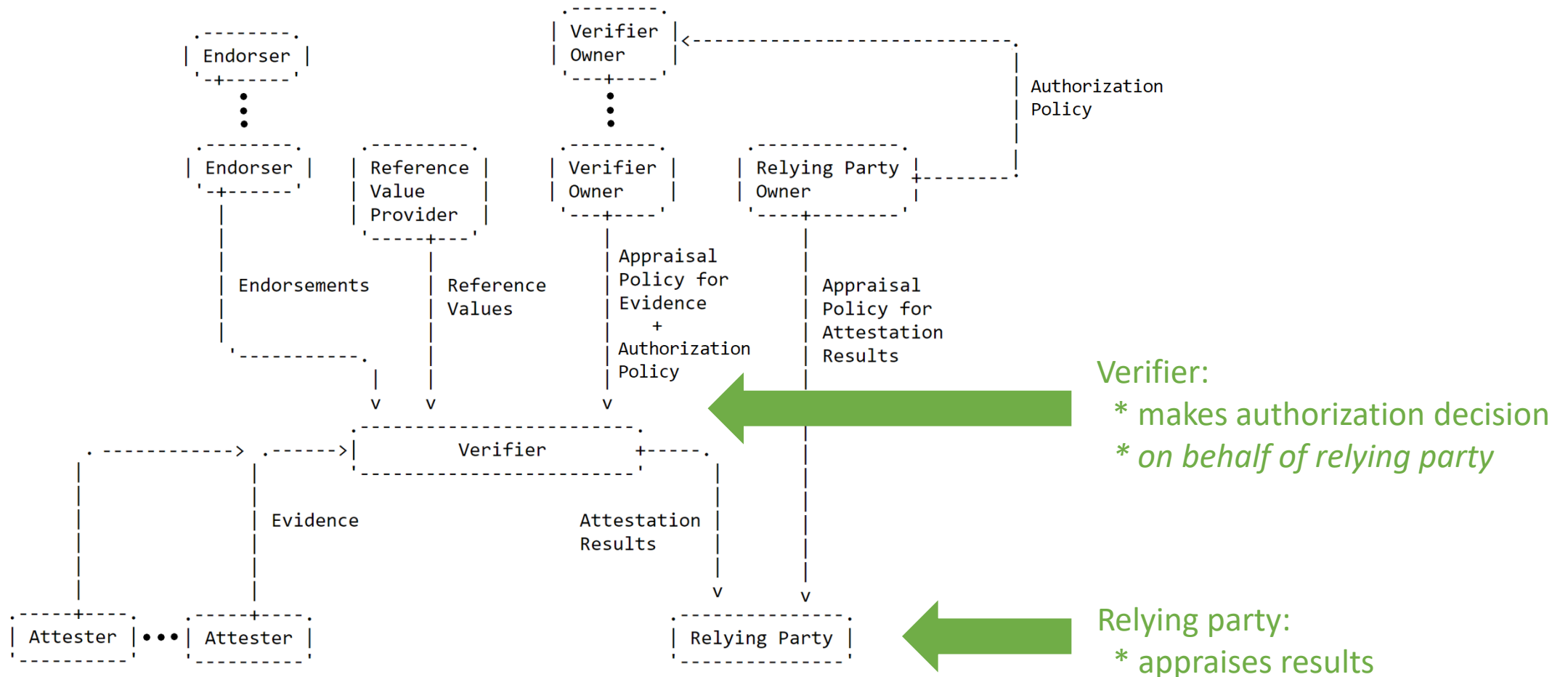
issuancerules
{
    // Set CurrentVersion claim value based on matching MRENCLAVE value for most recent build
    [type=="x-ms-sgx-mrenclave", value=="ece566969a914712f15e736cebfe3aa06cec28a1a75cf4701cf2270da2a93512"]
    => issue(type="CurrentVersion", value=true);

    [type=="x-ms-sgx-mrenclave", value!="ece566969a914712f15e736cebfe3aa06cec28a1a75cf4701cf2270da2a93512"]
    => issue(type="CurrentVersion", value=false);
};
```


MAA and RATS Architecture



MAA and RATS Architecture



MAA application of EAT

- Per Thomas' slides from 6/7/22
- EAT offers:
 - A number of pre-defined "claims"
 - readily reusable pieces of semantics
- Implementor choices for claim names:
 - Proprietary Claims only (plus maybe EAT.profile)
 - Standard Claims
 - A mix of proprietary and standard (a la PSA)  MAA choice

MAA Claim Name Principles

- EAT pre-defined claim names will be used where possible
- Claim names will be one of:
 - registered with IANA
 - defined in the EAT specification (destined to be registered in IANA)
 - prefixed by “x-ms”
- Claims can be categorized based on source (e.g. evidence, policy, etc.)
- Claim categories are expressed via one of:
 - Claim name prefix (e.g. “x-ms-sgx”)
 - Claim location (e.g. child of top level “x-ms-policy” claim)

MAA Use of Registered Claim Names

Name	Source	Meaning
jti	IETF JWT (RFC 7519)	Unique identifier for the JWT
iss	IETF JWT (RFC 7519)	Principal that issued the JWT
iat	IETF JWT (RFC 7519)	Issued at time for the JWT
exp	IETF JWT (RFC 7519)	Expiry time for the JWT
nbf	IETF JWT (RFC 7519)	Not before time for the JWT
cnf	IETF POP Key (RFC 7800)	JWK with public portion of enclave key
nonce (=> eat_nonce)	IETF EAT	Direct copy of nonce value provided by caller
secureboot (=> secboot)	IETF EAT	True if secure boot enabled, false otherwise

- This list is expected to grow as standardization of attestation results progresses (e.g. IETF EAT, AR4SI, etc.)

MAA Use of Unregistered Claim Names

Claim Name Prefix	Meaning
x-ms	Claims sourced by the MAA service
x-ms-<type>	Claims sourced from attestation evidence for the “type”
x-ms-sgx-<name>	Claims sourced from SGX evidence
x-ms-sevsnpvm-<name>	Claims sourced from SEV-SNP evidence
x-ms-tpm-<name>	Claims sourced from TPM evidence

Claim Name	Meaning
x-ms-runtime	Container for claims sourced from within the TEE at runtime
x-ms-inittime	Container for claims sourced from the host application at TEE creation time
x-ms-policy	Container for claims added by MAA policy


Source: MAA service

Name	Optional	Meaning
jti		Unique identifier for the JWT
iss		Principal that issued the JWT
iat		Issued at time for the JWT
exp		Expiry time for the JWT
nbf		Not before time for the JWT
x-ms-ver		JWT schema version
x-ms-attestation-type		String value representing attestation type
x-ms-policy-hash		Hash of MAA appraisal policy
x-ms-policy-signer	Yes	JWK defining key customer used to sign their policy
x-ms-compliance-status	Yes	Compliance status of attested environment (e.g. "azure-compliant-cvm")
nonce (=> eat_nonce)	Yes	Direct copy of nonce value provided by caller

- { x-ms-ver, x-ms-attestation-type } is logically equivalent to the eat_profile claim
- x-ms-compliance-status can be thought of as reflecting a CSP endorsement for the attested environment

Source: MAA service (example)

```
{  
  "exp": 1655359101,  
  "iat": 1655330301,  
  "iss": "https://maasandbox0001.wus.attest.azure.net",  
  "jti": "de0f1f6efd38f65d318f3acf7195f05d81e963008b08df9d0d64213c08a92d31",  
  "nbf": 1655330301,  
  "nonce": "12345678901234567890",  
  "x-ms-attestation-type": "sgx",  
  "x-ms-policy-hash": "rT7nv1FXdb2_MPe0M1zcm2LIB5xVfMy-Xp8awfYWfQU",  
  "x-ms-ver": "1.0",  
  ...  
}
```



DNS name for
tenant specific
service instance

Source: SGX attestation evidence

Name	Processor(s)	Meaning
x-ms-sgx-is-debuggable	Coffeelake + Icelake	SGX enclave is-debuggable flag value
x-ms-sgx-product-id	Coffeelake + Icelake	SGX enclave product id value
x-ms-sgx-mrenclave	Coffeelake + Icelake	SGX enclave mrenclave value
x-ms-sgx-mrsigner	Coffeelake + Icelake	SGX enclave msrigner value
x-ms-sgx-svn	Coffeelake + Icelake	SGX enclave SVN value
x-ms-sgx-report-data	Coffeelake + Icelake	SGX enclave report data field
x-ms-sgx-config-id	Icelake	SGX config id field
x-ms-sgx-config-svn	Icelake	SGX config SVN value
x-ms-sgx-isv-extended-product-id	Icelake	ISV extended product ID
x-ms-sgx-isv-family-id	Icelake	ISV family ID
x-ms-sgx-collateral	Coffeelake + Icelake	A JSON object describing the collateral used to perform attestation

Source: SGX attestation evidence (example)

[illegible]

Hashes of endorsement values

Icelake specific claims

Source: SEV-SNP attestation evidence

Name	Meaning
x-ms-sevsnpvm-authorkeydigest	The SHA384 hash of the author signing key
x-ms-sevsnpvm-bootloader-svn	The AMD boot loader security version number (SVN)
x-ms-sevsnpvm-familyld	The HCL family identification string
x-ms-sevsnpvm-guestsvn	The HCL security version number (SVN)
x-ms-sevsnpvm-hostdata	Arbitrary data defined by the host at VM launch time
x-ms-sevsnpvm-idkeydigest	The SHA384 hash of the identification signing key
x-ms-sevsnpvm-imageid	The HCL image identification
x-ms-sevsnpvm-is-debuggable	Boolean value indicating whether AMD SEV-SNP debugging is enabled
x-ms-sevsnpvm-launchmeasurement	Measurement of the launched guest image
x-ms-sevsnpvm-microcode-svn	AMD microcode security version number (SVN)
x-ms-sevsnpvm-migration-allowed	Boolean value indicating whether AMD SEV-SNP migration support is enabled
x-ms-sevsnpvm-reportdata	CVM: Data passed by HCL to include with report, to verify that transfer key and VM configuration are correct
x-ms-sevsnpvm-reportid	Report ID of the guest
x-ms-sevsnpvm-smt-allowed	Boolean value indicating whether SMT is enabled on the host
x-ms-sevsnpvm-snpfw-svn	AMD firmware security version number (SVN)
x-ms-sevsnpvm-tee-svn	AMD trusted execution environment (TEE) security version number (SVN)
x-ms-sevsnpvm-vmpl	VMPL that generated this report (0 for HCL)

- Note: Definitions based on Azure “confidential VM” use of SEV-SNP attestation

Source: SEV-SNP attestation evidence (example)

[illegible]

MAA service claim
based on CSP
endorsements

Source: TEE runtime (background)

- When attestation evidence is created within a TEE:
 - many TEE's allow inclusion of a small amount of arbitrary data
 - interpretation of the data is a contract between the TEE and the relying party
 - a common pattern is to store the hash of a larger amount of data
- Available in both SGX and SEV-SNP attestation evidence
 - The attribute is called “report data”
 - The attribute size is 64 bytes

Source: TEE runtime (background cont'd)

- MAA understands this usage pattern
 - Assumption: “report data” contains the SHA256 hash of the data
 - Can turn associated data into full formed claims in JWT
- Verification processing:
 - Wire protocol accepts optional “runtime data” parameter
 - MAA service verifies that hash of “runtime data” matches “report data” value
 - MAA includes the “runtime data” as a claim in the JWT
- JWT claim format:
 - “runtime data” input is tagged with a data type: “Binary” or “JSON”
 - If “Binary”, format is Base64URL encoded version of “runtime data”
 - If “JSON”, format is deserialized JSON object extracted from “runtime data”

Source: TEE runtime

Name	Meaning
x-ms-sgx-ehd	A copy of the runtime data formatted as BASE64URL(runtime data)
x-ms-runtime	A copy of the runtime data formatted as a JSON object

- x-ms-sgx-ehd is deprecated
- x-ms-runtime is the recommended format to share verified runtime information between a TEE and a relying party

Source: TEE runtime (example - Binary)

```
{
    ...
    "x-ms-sgx-ehd":
    "ewogICAgImtleXMiOiBbCiAgICAgICB7CiAgICAgICAgICAiZSI6ICJBUEUFCIiwKICAgICAgICAg
    ICJrZXlfY3BzIjogWwogICAgICAgICAgICAgICAgImVuY3J5cHQiCiAgICAgICAgICBdLAogICAgICAgI
    CAgImtpZCI6ICJTZWVDb21tdW5pY2F0aW9uQ2hhbm5lbEtleSIsCiAgICAgICAgICAgICAia3R5Ij
    ogIlJlTQSIsCiAgICAgICAgICAgICAibiI6ICJ1cjA4RGNjakdHelJvM09JcTQ0NW4wMFEzITRoTUliUjN
    TV0l6Q2NpY0lNXzduUGlwRjVOQklrbmsiemRIWk4xaW10aEl6SmV6clhTcVZUN1R5MURsNEFCNXhp
    QUFXeG83eEdqRnFsTDQ3TkE4V2JaUk14UXR3bHNPalpnrnhvc0ROWE10NmRNcTdPRGg0bmo2blYyS
    k1TY05mUkt5cjFYRklVSzBYa09Xd1ZsU2xOWmphQXhqOEg0cFMweU5mTndyMVE5NFZKU24zTFBSdV
    pCSEU3VnJvZkhSR1NISnJhRGxsZktUMC04b0tXOEVqcE13djFNRV9PZ1BxUHdMeW1SenI5OW1vQjd
    1eHpqRVZEZTU1RDJpMm1QcmNtVDdrU3NIId3A1TzJ4S2hNNjhYGE2Ri1JVDIxSmckaFE2bjRIV0Np
    Y3NsQm14NG9xa0kteDVsVnNSa1EiCiAgICAgICB9CiAgICBdCn0gICA",
    ...
}
```

Source: TEE runtime (example - JSON)

```
{  
  ...  
  "x-ms-runtime": {  
    "keys": [  
      {  
        "e": "AQAB",  
        "key_ops": [  
          "encrypt"  
        ],  
        "kid": "SecureCommunicationChannelKey",  
        "kty": "RSA",  
        "n": "ur08DccjGGzRo306n4HWCics1Bmx4oqkI-x5lVsRkQ"  
      }  
    ]  
  },  
  ...  
}
```


Source: TEE inittime (background)

- When a host process launches a TEE:
 - many TEE's allow the definition of a small amount of arbitrary “configuration” data
 - this “configuration” data is:
 - included in all attestation evidence produced for the running enclave
 - immutable after TEE creation
 - interpretation of the data is between the host, TEE and the relying party
 - a common pattern is to store the hash of a larger amount of configuration data
- Available in both SGX (Icelake) and SEV-SNP attestation evidence
 - On SGX (Icelake), the attribute is called “[config id](#)” and is 64 bytes long
 - On SEV-SNP, the attribute is called “[host data](#)” and is 32 bytes long

Source: TEE inittime (background cont'd)

- MAA understands this usage pattern
 - Assumption: “[config id](#)” or “[host data](#)” contains the SHA256 hash of the data
 - Can turn associated data into full formed claims in JWT
- Verification processing:
 - Wire protocol accepts optional “inittime data” parameter
 - MAA verifies hash of “inittime data” matches “[config id](#)” or “[host data](#)”
 - MAA includes the “inittime data” as a claim in the JWT
- JWT claim format:
 - Format is deserialized JSON object extracted from “inittime data”

Source: TEE inittime

Name	Meaning
x-ms-inittime	A copy of the inittime data formatted as a JSON object

Source: TEE inittime (example)

```
{
  ...
  "x-ms-inittime": {
    "mounts": {
      "disk1": {
        "destination": "/mnt/disk1",
        "diskhash": "2413FB3709B05939F04CF2E92F7D0897FC2596F9AD0B8A9EA855C7BFEBAAE892"
      },
      "disk2": {
        "destination": "/mnt/disk2",
        "diskhash": "47A6F6B0F734662F5C723ED8D1AB823BFC280D6FA14820C957906DB1417E882D"
      }
    }
  },
  ...
}
```

Source: MAA Customer Defined Policy

Name	Meaning
x-ms-policy	JSON object containing all claims added by customer defined MAA policy

- Azure customer defined MAA policy allows customers to add claims
- For example:

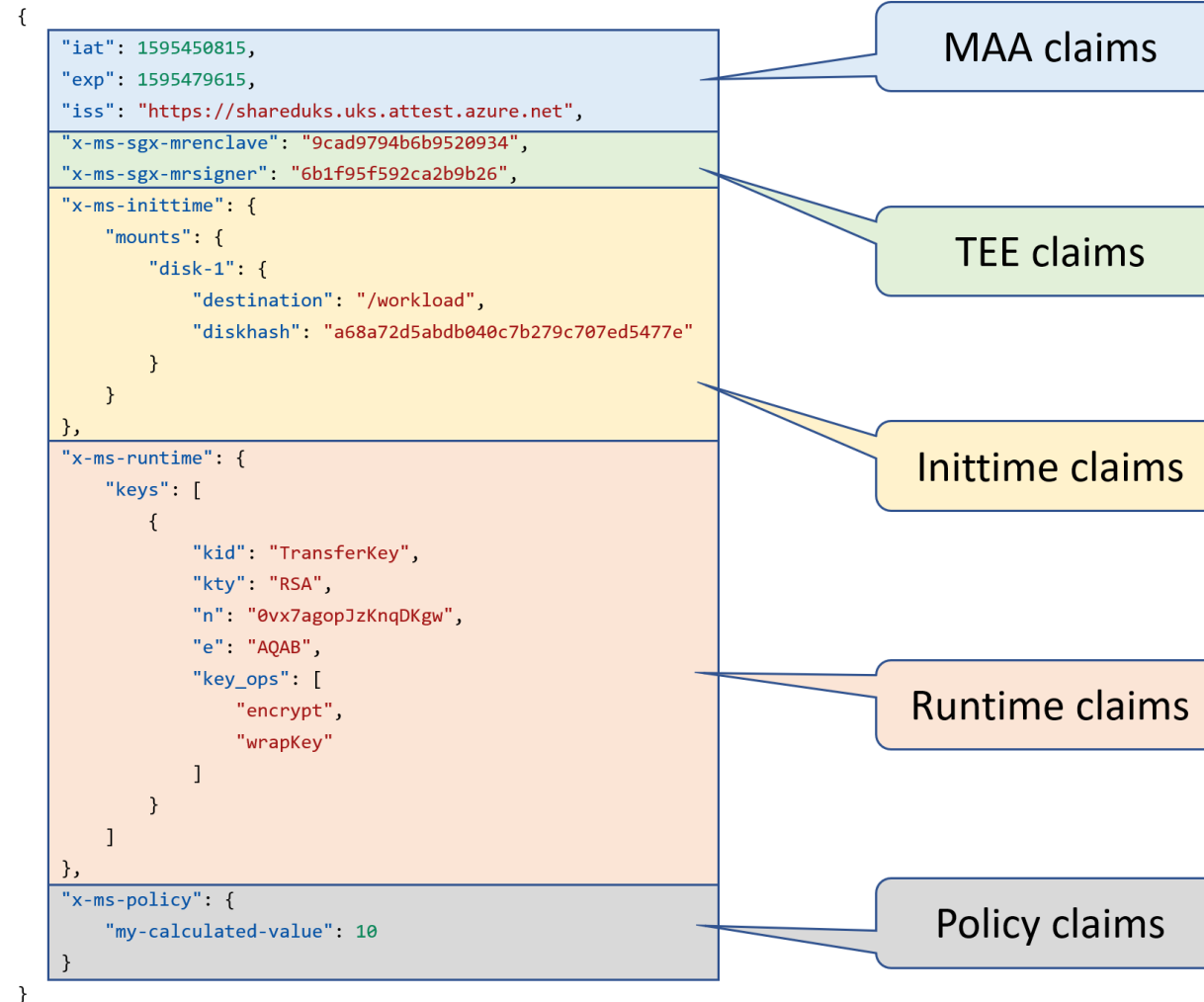
```
issuancerules
{
    // Set CurrentVersion claim value based on matching MRENCLAVE value for most recent build
    [type=="x-ms-sgx-mrenclave", value=="ece566969a914712f15e736cebfe3aa06cec28a1a75cf4701cf2270da2a93512"]
        => issue(type="CurrentVersion", value=true);

    [type=="x-ms-sgx-mrenclave", value!="ece566969a914712f15e736cebfe3aa06cec28a1a75cf4701cf2270da2a93512"]
        => issue(type="CurrentVersion", value=false);
};
```

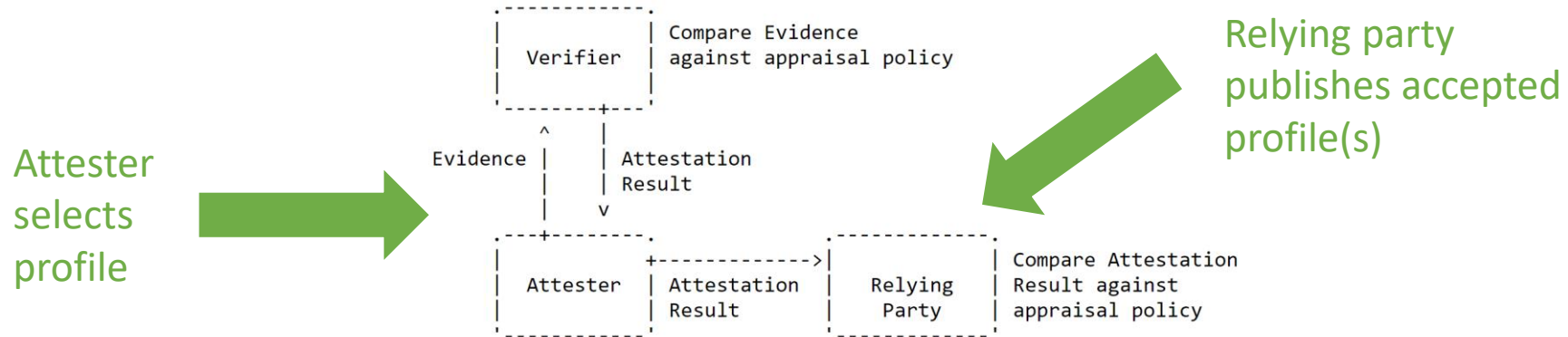
Source: Appraisal Policy (example)

```
{  
  ...  
  "x-ms-policy": {  
    "CurrentVersion": true,  
    "ExfiltrationRiskScore": 77,  
    "LegalDepartmentApproved": false  
  },  
  ...  
}
```

Example: All Sources



Challenge: Renaming claims in passport model



- Expected mechanism to rename a claim:
 - publish a new profile
 - verifier allows caller to select desired profile
- Feedback MAA has received from relying parties on this idea:
 - Simply put, “no!”
 - No desire to add support for new profiles that simply encompass a cosmetic change. If it ain’t broke, don’t fix it!
 - There is no way to add support for a new profile before an attester might start to use it.
 - Attester selection of an unsupported profile will lead to operational failures and higher support costs.
- MAA mitigation:
 - One profile (per attestation type) that we never break
 - Rename really means add new claim name and deprecate (but still support/include) the old claim name
 - Obvious downside is token bloat

Challenge: EAT submod is vague

[4.2.19.](#) Submodules (submods)

Some devices are complex, having many subsystems. A mobile phone is a good example. It may have several connectivity subsystems for communications (e.g., Wi-Fi and cellular). It may have subsystems for low-power audio and video playback. It may have multiple security-oriented subsystems like a TEE and a Secure Element.

The claims for a subsystem can be grouped together in a submodule or submod.

[4.2.19.1.1.](#) Submodule Claims-Set

This is a subordinate Claims-Set containing claims about a submodule, a subordinate entity.

[4.2.19.1.2.](#) Nested Token

This type of submodule is a fully formed complete token. It is typically produced by a separate Attester. It is typically used by a composite device as described in RATS Architecture

[4.2.19.1.3.](#) Detached Submodule Digest

This is type of submodule equivalent to a Claims-Set submodule, except the Claims-Set is conveyed separately outside of the token.

- What is a submod?
 - A physical device?
 - A “subsystem”?
 - A “subordinate entity”?
 - A part of a “composite device”?
- The definition is vague
- Readers will interpret differently
 - Dave – MAA runtime is clearly a submod
 - Greg – MAA runtime could be a submod, but:
 - no obvious suggestion it should be a submod
 - is simpler and EAT compliant without being a submod
- Confusion/friction: A submod
 - is not the only way to nest claimsets
 - does not model non-hierarchical relationships
- A submod seems directly applicable to:
 - Independently signed pieces of attestation evidence
 - A hierarchy of attestation evidence

What's Next?

- Participate in and track standards efforts (e.g. EAT, AR4SI)
- Adopt standard claim names as make sense
- Items on the radar for consideration include:

Name	Notes
eat_profile	Documentation of specific claims used in a MAA EAT JWT
eat_nonce	Rename “nonce” to “eat_nonce”; verify correct usage/data-type
dbgstat	Add support where possible based on attestation evidence
secboot	Rename secureboot to secboot; add additional support when known
manifest	Add support where possible based on attestation evidence
swevidence	Add support where possible based on attestation evidence
swresults	Add support where possible based on attestation evidence
AR4SI	Assess if any relying parties are asking for or could benefit from AR4SI

Questions?

Appendix

Bookmarks

- MAA overview
 - <https://docs.microsoft.com/en-us/azure/attestation/overview>
- MAA JWT claim names
 - <https://docs.microsoft.com/en-us/azure/attestation/claim-sets#outgoing-claims>
- MAA Appraisal Policy grammar
 - <https://docs.microsoft.com/en-us/azure/attestation/claim-rule-grammar>

<https://aka.ms/maasandbox>

Attestation Type <

MAA Sandbox

SGX Attestation

Mon Jun 20 2022 2:15 PM

SGX

SevSnpVm

TPM

AzureVm

My SGX quote

Help

MAA Policy

Sample: Default

Sample Policy:

SELECT ...

```

1  version= 1.0;
2
3  authorizationrules
4  {
5      => permit();
6  };
7
8  issuancerules
9  {
10     c:[type=="x-ms-sgx-is-debuggable"] => issue(type="is-debuggable", value=c.value);
11     c:[type=="x-ms-sgx-mrsigner"] => issue(type="sgx-mrsigner", value=c.value);
12     c:[type=="x-ms-sgx-mrenclave"] => issue(type="sgx-mrenclave", value=c.value);
13     c:[type=="x-ms-sgx-product-id"] => issue(type="product-id", value=c.value);
14     c:[type=="x-ms-sgx-svn"] => issue(type="svn", value=c.value);
15     c:[type=="x-ms-attestation-type"] => issue(type="tee", value=c.value);
16 };

```

MAA Claims

GENERATE

Use my quote

Binary runtime data

FOLD

DOWNLOAD JWT

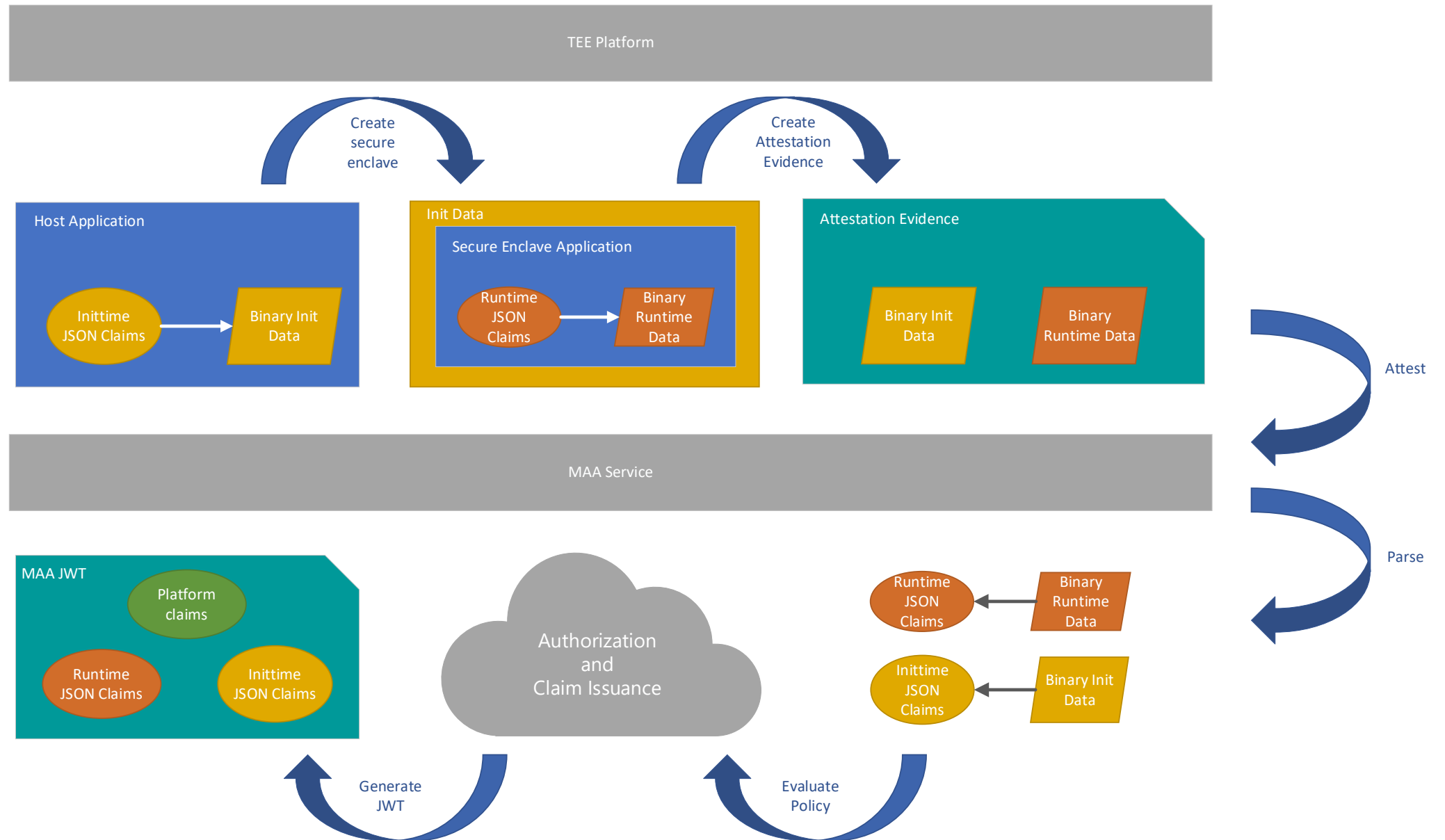
OPTIONS ...

```

1  {
2      "exp": 1655788500,
3      "iat": 1655759700,
4      "iss": "https://maasandbox0001.wus.attest.azure.net",
5      "jti": "4c7d4470c15001995f7a4be7014aa8a44aed3aea1ad2d7449fe944fe3abf7a6d",
6      "nbf": 1655759700,
7      "x-ms-attestation-type": "sgx",
8      "x-ms-policy": { ...
9  },
10     "x-ms-policy-hash": "rT7nv1FXdb2_MPe0M1zcm2LIB5xVfMy-Xp8awfYwFQU",
11     "x-ms-runtime": {
12         "keys": [
13             {
14                 "e": "AQAB",
15                 "key_ops": [
16                     "encrypt",
17                     "wrapKey"
18                 ],
19                 "kid": "SecureTransferKey",
20                 "kty": "RSA",
21                 "n": "t7hvgdk1Un_yrqmB1NyQZEaCtKycoZVH-0pDCoPaEAVuk6Ug1ALjRfEjvh3he1aixmSmioJR80L1wretC
22             }
23         ],
24     },
25     "x-ms-sgx-collateral": {
26         "qeidcertshash": "a64d649198507d8b57e33f63ab266838f43f327bd4aacc78510b6976ed046e10",
27         "qeidcrlhash": "3dbcd25597ba0548bf32240b3079d4310151756f17e5537d3015b16e399acad5",
28         "qeidhash": "7701f64700b7f505d7b4b7a93e45d5cde8cfc865b60f1dd49ecbee9790c3372e",
29         "quotehash": "b95fe2b6a1fae7201c36096fbd17beed2ce31f77b02fd45fc9c8ccfcb0ca5fb",
30         "tcbinfocertshash": "a64d649198507d8b57e33f63ab266838f43f327bd4aacc78510b6976ed046e10",
31         "tcbinfocrlhash": "3dbcd25597ba0548bf32240b3079d4310151756f17e5537d3015b16e399acad5",
32         "tcbinfohash": "82d109fb308f24a90e43936ea9e12b55b05250221fda2294f74ab5817e71bea4"
33     },
34     "x-ms-sgx-is-debuggable": true,
35     "x-ms-sgx-mrenclave": "ece566969a914712f15e736cebfe3aa06cec28a1a75cf4701cf2270da2a93512",
36     "x-ms-sgx-mrsigner": "61b8f05efb4e3b259f3b1bba6b89a1a16784bd4b3b172d00ba22a36c7131d2d5",
37     "x-ms-sgx-product-id": 1,
38     "x-ms-sgx-report-data": "abb086ff754a347a511a7d0955cefaba6d52c8e8da3a95c373f37d51f8a09416600000000",
39     "x-ms-sgx-svn": 1,
40     "x-ms-ver": "1.0"
41 }

```

Conceptual Flow for Runtime and Inittime Claims



Claim Sources (temporal ordering)

