FAPI-SIG Community 25th Meeting

@Web Conference

18 Aug 2021

PROPOSED DRAFT

Table of Contents

Major Topics

Proposal :

  Client Policies Revised

  FAPI 2.0 Baseline

Working Items Status

PROPOSED DRAFT

Major Topics

Major Topics

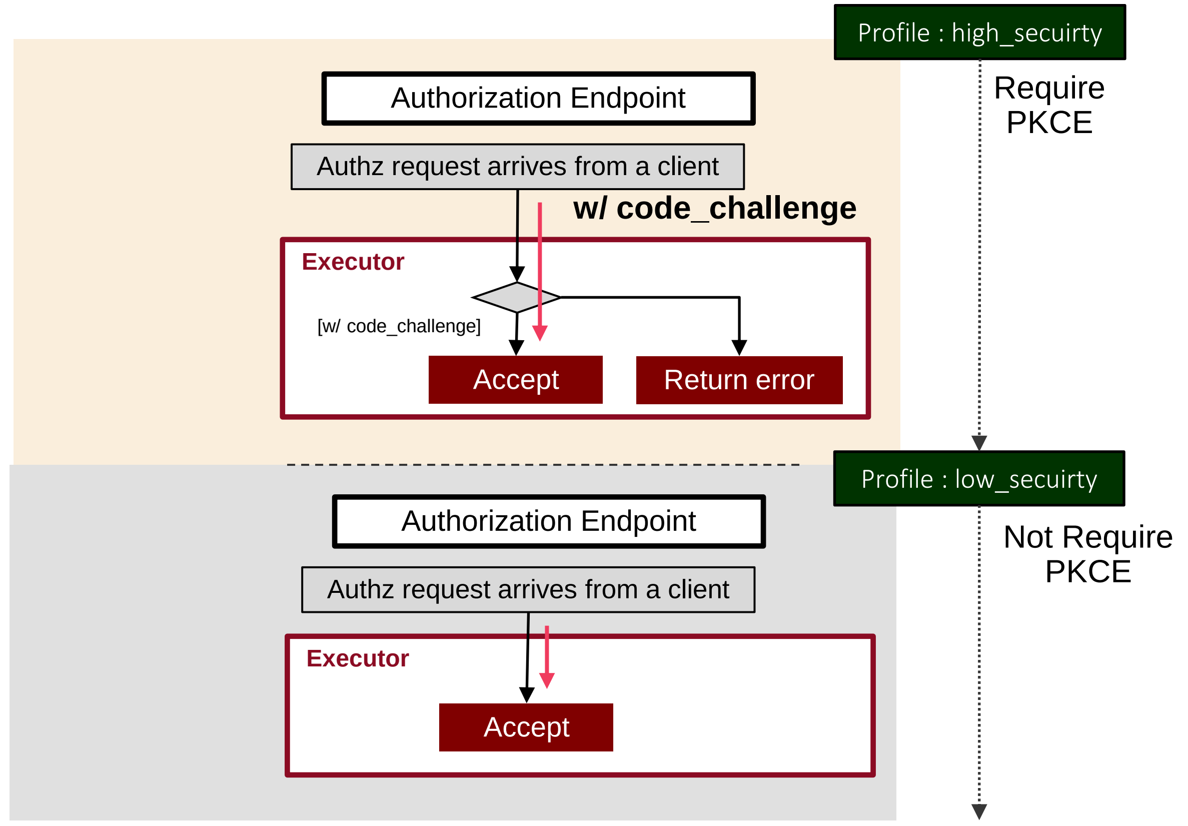
- Automated Conformance Test Run Environment in kc-fapi-sig repository

  It supported Open Banking Brasil FAPI 1.0 conformance tests.

  - Client Authentication Method : MTLS, private_key_jwt
  - Signature Algorithm : PS256
  - Request Object Method : plain, PAR
  - Response Mode : plain, JARM

Proposal : Client Policies Revised

Objectives



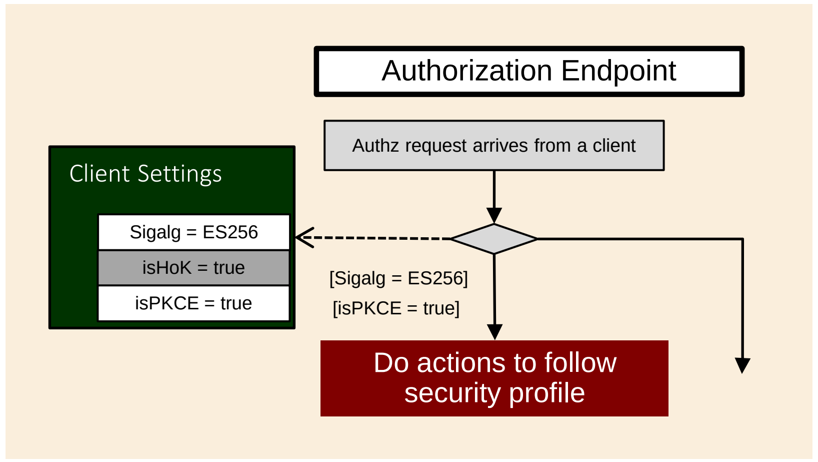
Current client policies have some limitation on dynamically changing security profiles per client request.

Client Policies Revised tries to get rid of this limitation.

To do so, client policies does not rely on client settings as much as possible.

PROPOSED DRAFT

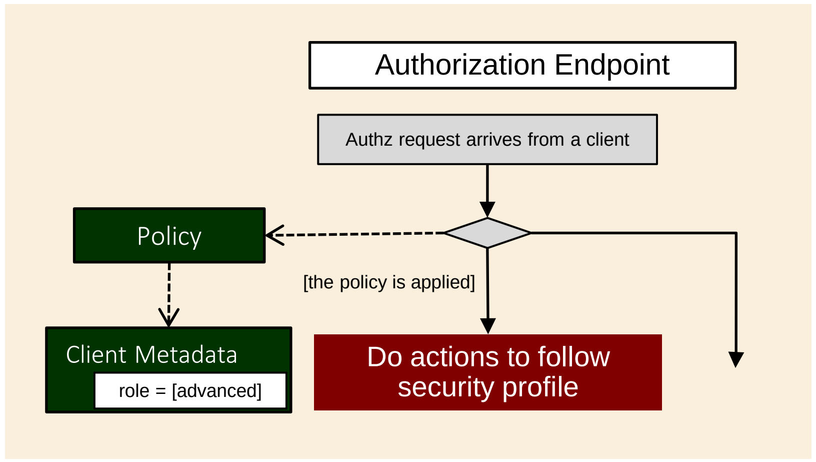
Settings vs Policies : Way of executing security profile related actions



[By Settings]

By referring values of client settings, determine whether security profile related actions and checks are executed.

- leverages existing logics.



[By Policies]

By evaluating a policy, determine whether security profile related actions and checks are executed.

- change security profiles dynamically and flexibly.

PROPOSED DRAFT

Proposal : FAPI 2.0 Baseline

PROPOSED DRAFT

FAPI 2.0 Baseline (Draft ver)

● The aim of supporting FAPI 2.0 Baseline that is still Draft ver

Make it ease to support it when it becomes Final version.

● Specification

FAPI 2.0 Baseline Profile :

https://bitbucket.org/openid/fapi/src/master/FAPI_2_0_Baseline_Profile.md

● Referred Commit

d65cc55574ae5de1cf4dd3517279a85d8b3cc14c

● Target

Requirements for Authorization Servers

FAPI 2.0 Baseline (Draft ver)

Supported #1 : Server Metadata Advertisement

Possible #2 : Accept Authorization Code Grant

Currently, it has already been achieved by client settings, not client policies.

Possible #3 : Reject Implicit Grant and Resource Owner Password Credentials Grant

Currently, it has already been achieved by client settings, not client policies.

Possible #4 : Accept PAR

Currently, it has already been achieved by client settings, not client policies.

Possible #5 : Reject authorization request except PAR

Currently, it has already been achieved by client settings, not client policies.

Supported #6 : Reject PAR without client authentication

Client is authenticated on PAR endpoint as default.

FAPI 2.0 Baseline (Draft ver)

In Progress	#7 : PAR's authorization_details support
Supported	#8 : Accept only confidential client
Supported	#9 : Holder-of-Key token support

Currently, it has already been achieved by mutual TLS. DPoP is under development.

Supported	#10 : Accept only specified secure client authentication methods
Supported	#11 : Require PKCE
Not Yet	#12 : Require PAR including redirect_uri
Not Yet	#13 : Require authorization response including iss
Not Yet	#14 : Make sure TLS for HTTP redirect with authorization response
Supported	#15 : Detect multiple use of authorization code

Supported NOTE section's description.

FAPI 2.0 Baseline (Draft ver)

Supported	#16 : Provide mechanism for verifying access token by resource server
Supported	#17 : Prohibit 307 redirect
Optional	#18 : Should use 303 redirect
Supported	#19 : Prohibit Open Redirector
Supported	#20 : Check aud claim's value of JWS client assertion in private_key_jwt

Currently, JWTClientAuthenticator supported it.

Difference between FAPI2 and FAPI1

#1 : Not require JAR and JARM

#2 : Require RAR

#3 : Not require s_hash

#4 : Require redirect_uri in PAR

#5 : Not require hybrid flow

#6 : Not require ID token as detached signature

#7 : Not require JWE for ID token

#8 : Add DPoP for Holder-of-Key token

Working Items Status

Working Items

[Security Features]

<Common>

In Progress OIDC Client’s Public Key Management — Hitachi

1st phase -> 2nd phase

New In Progress Client Policies Revised — Hitachi

[Security Features]

<SPA/Native App>

In Progress OAuth 2.0 Demonstration of Proof-of-Possession (DPoP) — Backbase

Working Items

[Security Features]

<High Level Security>

● FAPI 2.0 (baseline/advanced)

	In Progress	Rich Authorization Request (RAR)	Adorsys
	In Progress	Grant Management API	Adorsys
New	In Progress	Other requirements support	Hitachi

Working Items

[Market Specific Features]

<PSD2>

● Following eIDAS regulations

In Progress QWAC verification

Adorsys

● Consent Management

<UK OpenBanking>

● Onboarding

- Software Statement Support
- Software Statement Assertion (SSA) Verification

PROPOSED DRAFT

Roadmap

Security	2020 Sep	2021 Apr
Common Client Policies	• Implemented as Preview • New Admin REST API support	• Migrate Client Registration Policy • Move Preview to Official Support -> leads to FAPI 1.0 Official Support
High Level Security		
FAPI 1.0	• Supported ID2 ver • Passed ID2 conformance test	• Support Final ver • Pass Final ver conformance test
FAPI 2.0		• Start design and implementation PAR, RAR, Grant Management API
FAPI-CIBA	• Supported Pure CIBA (poll mode) as Preview	• Support FAPI-CIBA (poll mode) • Pass conformance test
SPA/Native App DPoP		• Start design and implementation

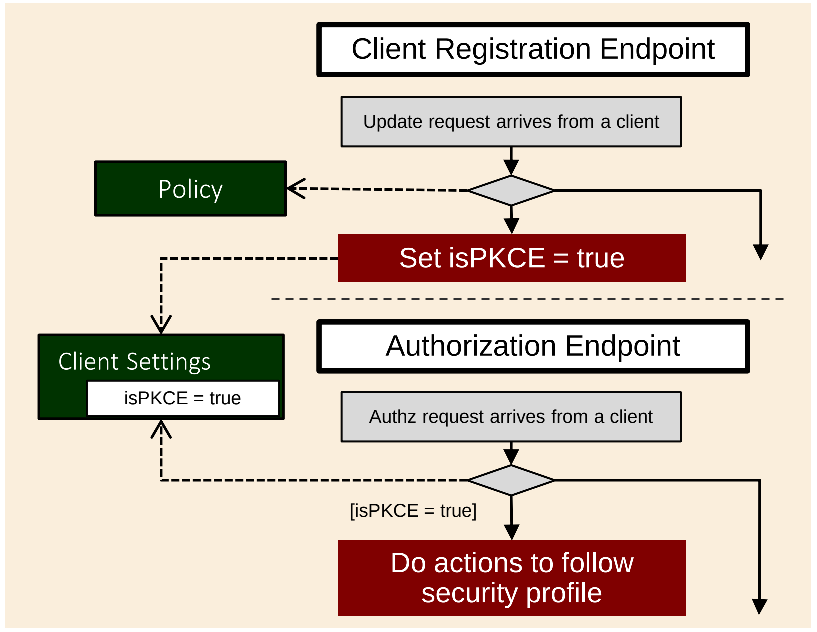
PROPOSED DRAFT

Roadmap

Market Specific	*2020* *Sep*	*2021* *Apr*
PSD2	• Discussed QWAC verification	• Start design and implementation QWAC verification
UK OpenBanking		• Start design and implementation (?) Onboarding
Open Banking Brasil		• Support OBB FAPI

Appendix : Client Policies Revised

Current Client Policies : Executor Type

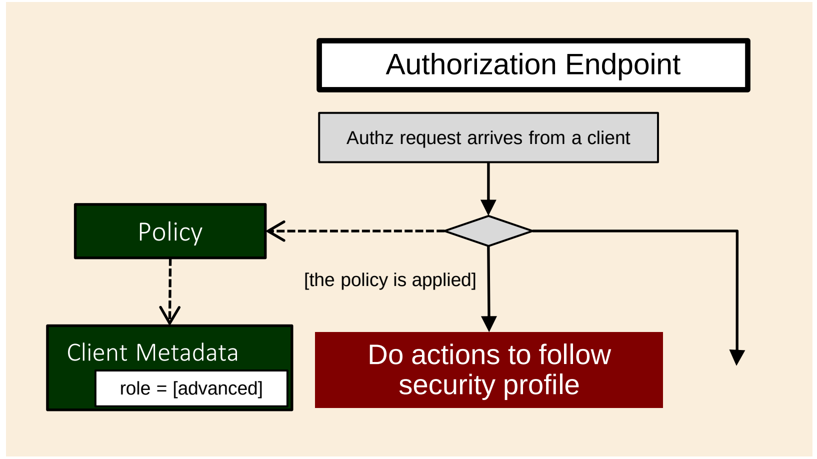

Client Registration Endpoint
Update request arrives from a client
Policy
Set isPKCE = true
Client Settings
isPKCE = true
Authorization Endpoint
Authz request arrives from a client
[isPKCE = true]
Do actions to follow security profile

[Type A : Enforce/Check Client Settings]

Enforce/check client settings on only client registration/update requests.

It does not work on other endpoints.

- leverages existing logics.



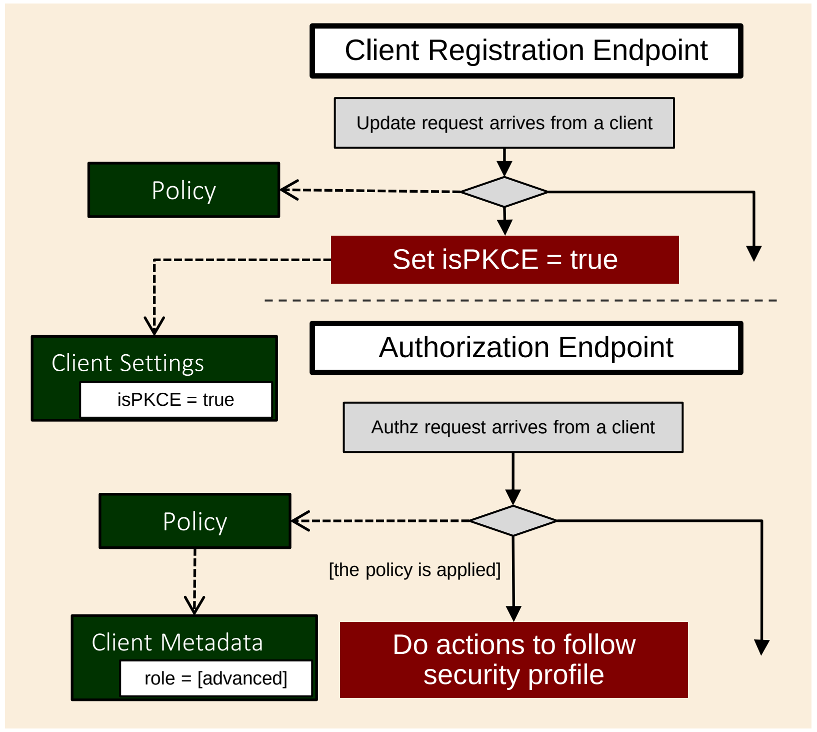
[Type B : Check on Runtime]

Do security profile related actions and checks on every requests from clients to endpoints.

- execute regardless of client settings.

PROPOSED DRAFT

Current Client Policies : Executor Type

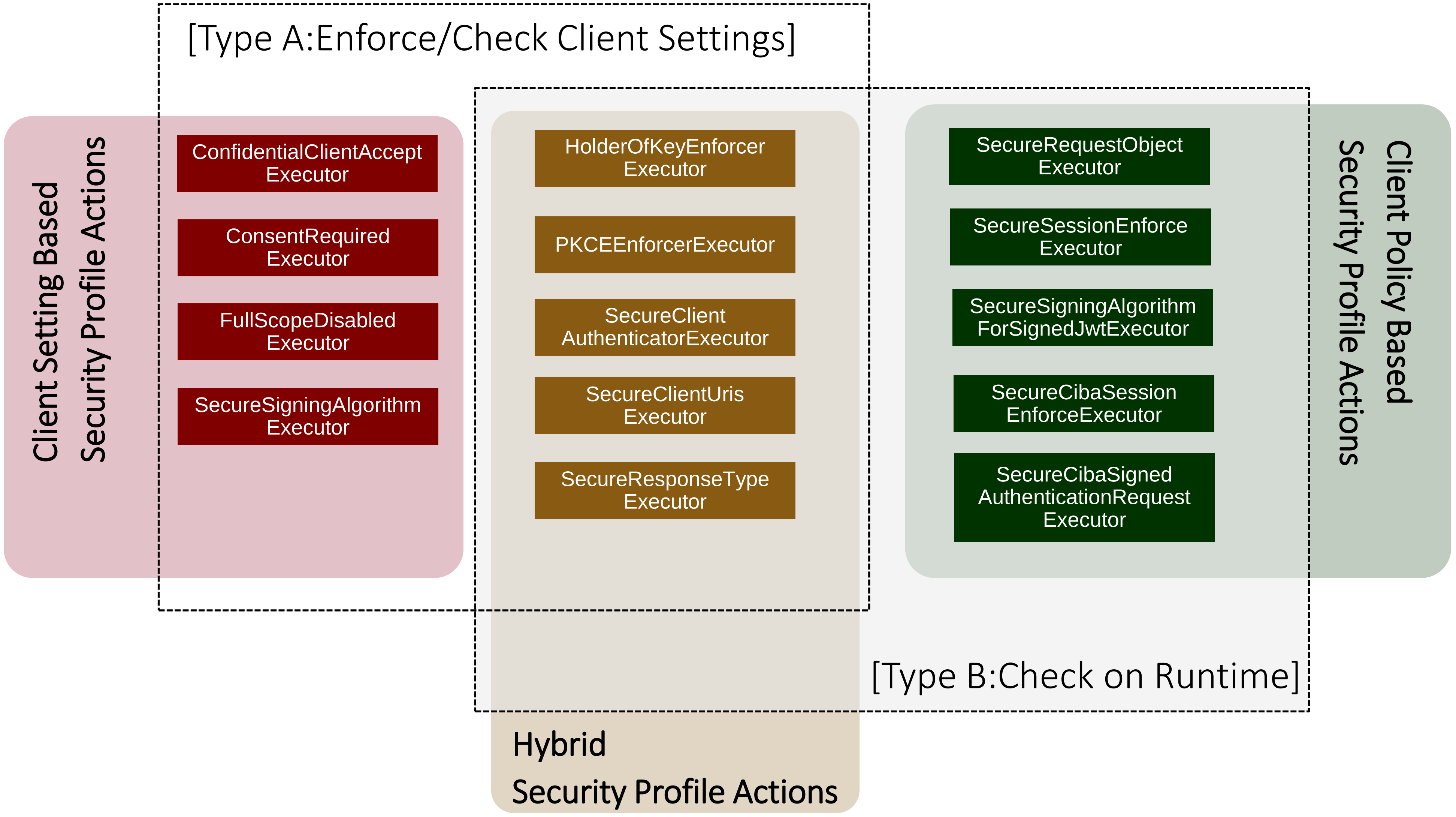

Client Registration Endpoint
Update request arrives from a client
Policy
Set isPKCE = true
Client Settings
isPKCE = true
Authorization Endpoint
Authz request arrives from a client
Policy
[the policy is applied]
Client Metadata
role = [advanced]
Do actions to follow security profile

[Type A & B : Hybrid]

Enforce/check client settings on only client registration/update requests.

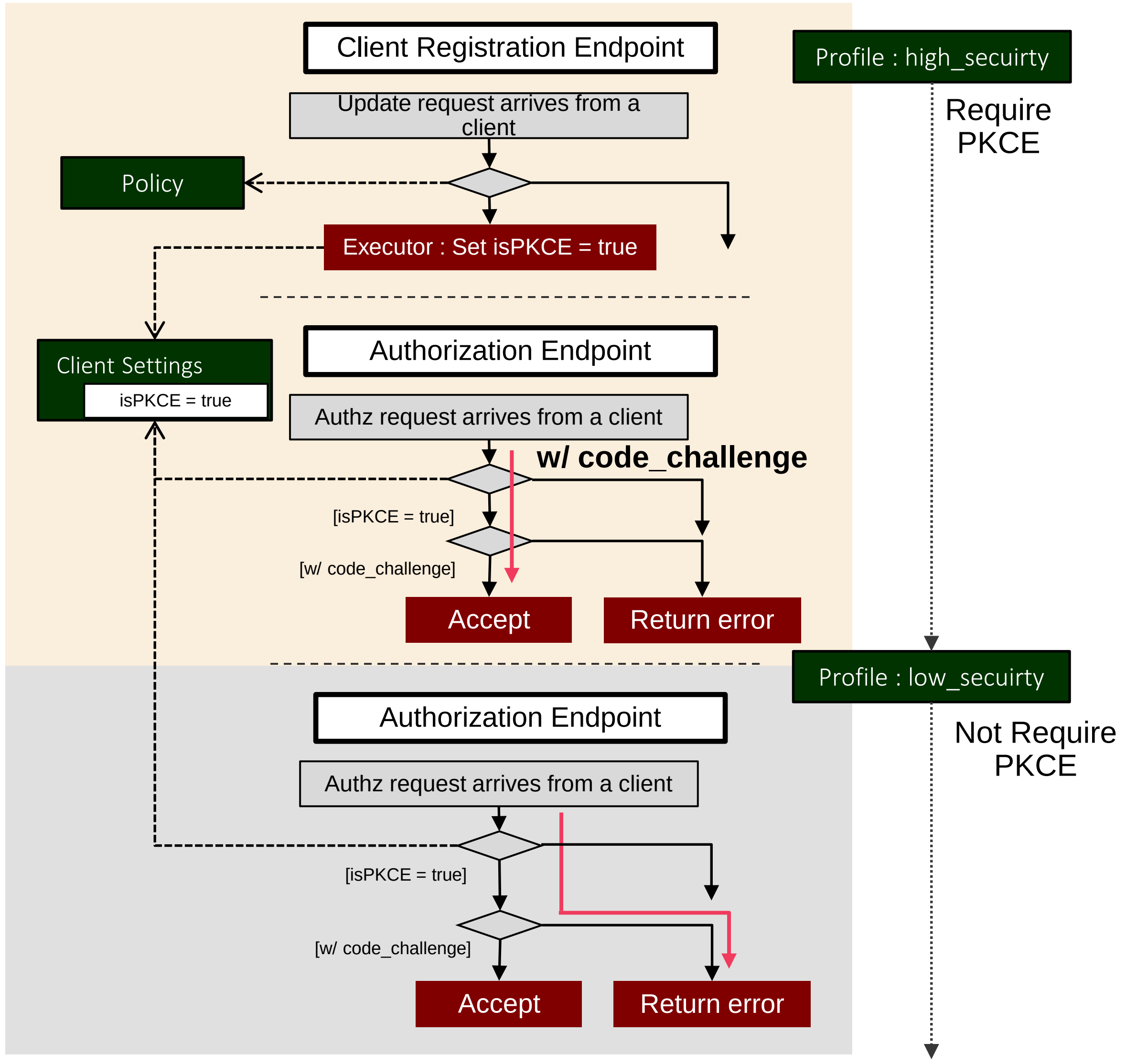
Do security profile related actions and checks on every requests from clients to endpoints.

Current Status : Ways of executing security profile


[Type A:Enforce/Check Client Settings]
Client Setting Based Security Profile Actions
ConfidentialClientAccept Executor
ConsentRequired Executor
FullScopeDisabled Executor
SecureSigningAlgorithm Executor
HolderOfKeyEnforcer Executor
PKCEEnforcerExecutor
SecureClient AuthenticatorExecutor
SecureClientUris Executor
SecureResponseType Executor
SecureRequestObject Executor
SecureSessionEnforce Executor
SecureSigningAlgorithm ForSignedJwtExecutor
SecureCibaSession EnforceExecutor
SecureCibaSigned AuthenticationRequest Executor
Client Policy Based Security Profile Actions
[Type B:Check on Runtime]
Hybrid Security Profile Actions

PROPOSED DRAFT

Limitation on dynamic change of security profile : Type A:Enforce/Check Client Settings



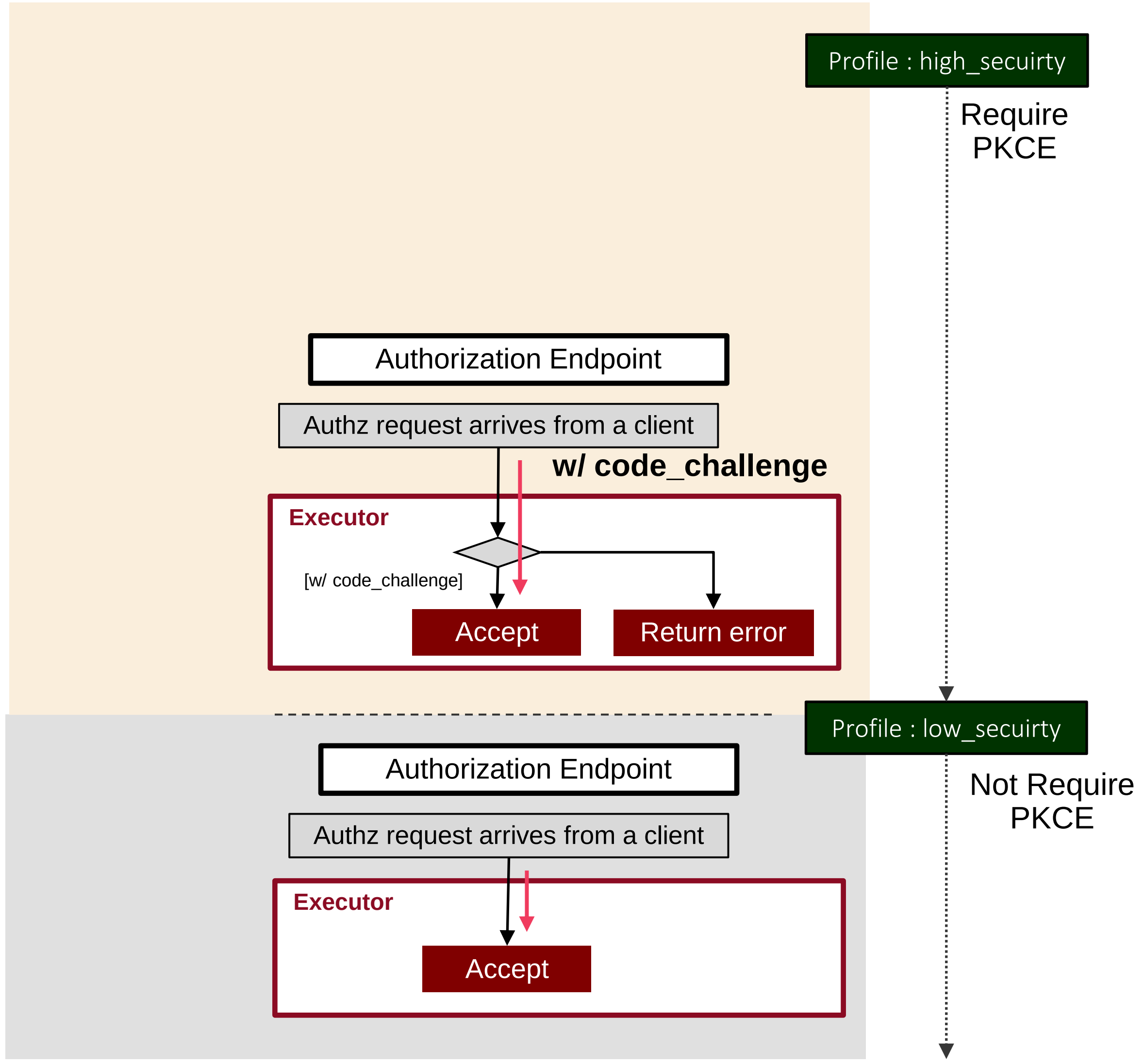
By employing the type A executor, some client setting need to be changed when changing security profiles depending on their nature.

It might be not good if clients want to change applied profiles per request because it enforces them to change their settings whenever sending requests that requires different profile.

E.g.

- Calling API for bank transfer, that required PKCE.
- Calling API for retrieving band account data, that does not require PKCE.

Un-Limitation on dynamic change of security profile : Type B:Check on Runtime



By employing the type B executor, some client setting need **NOT** to be changed when changing security profiles depending on their nature.

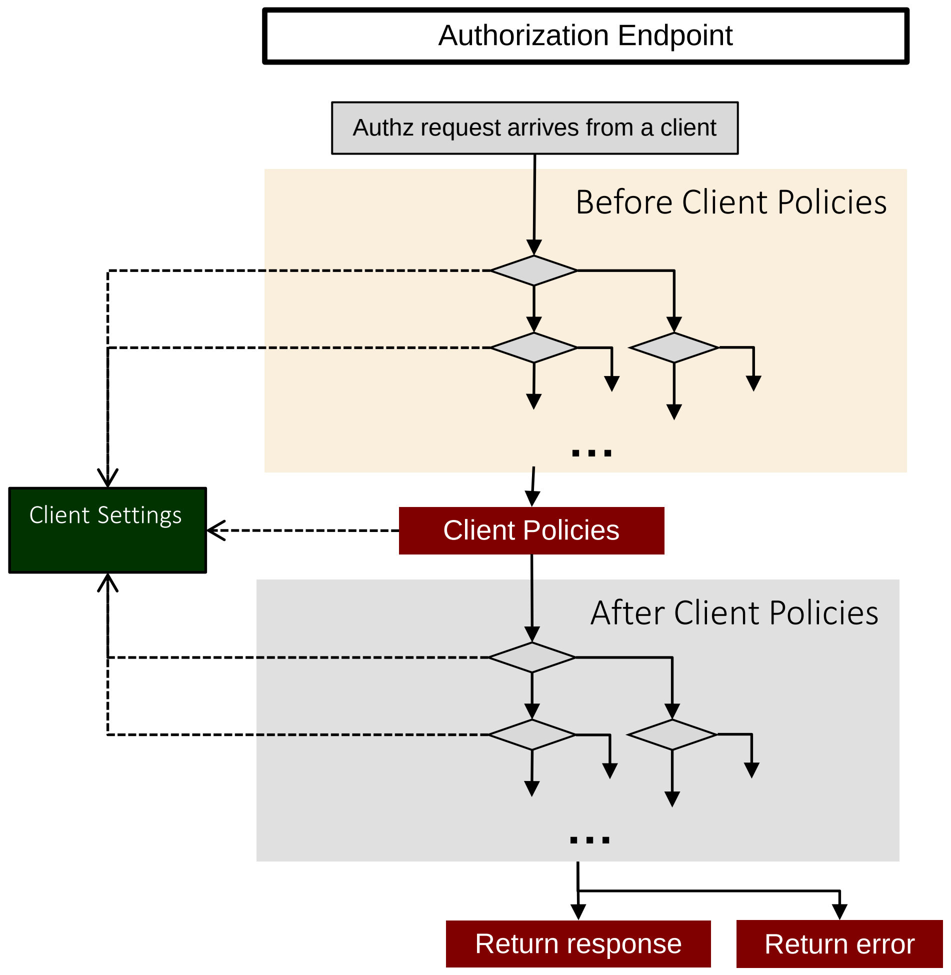
It might be good if clients want to change applied profiles per request because it **DOES NOT** enforce them to change their settings whenever sending requests that requires different profile.

E.g.

- Calling API for bank transfer, that required PKCE.
- Calling API for retrieving band account data, that does not require PKCE.

PROPOSED DRAFT

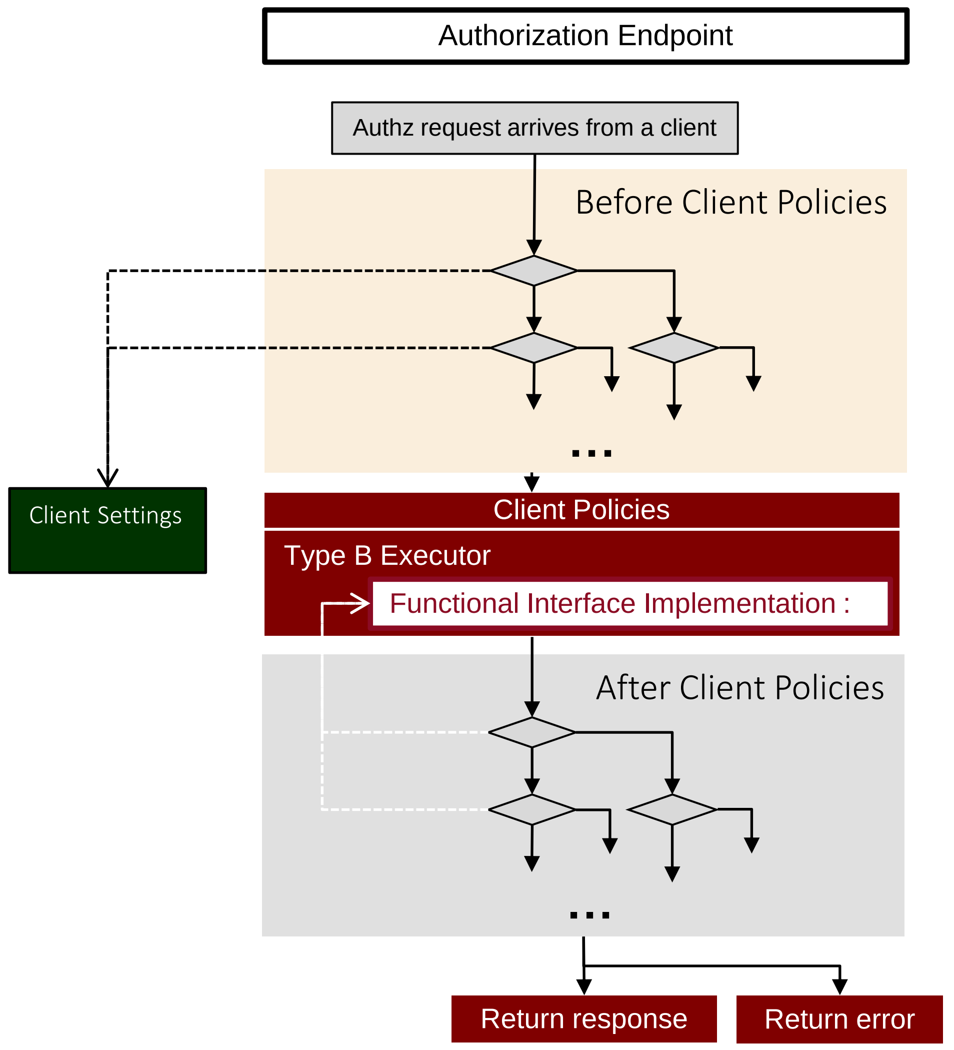
Difficulty : Migrating Type A to Type B Executor



In logics before executing client policies, some point refers client settings on the code path.

In logics after executing client policies, some point refers client settings on the code path.

Difficulty : Migrating Type A to Type B Executor



In logics before executing client policies, some point refers client settings on the code path.

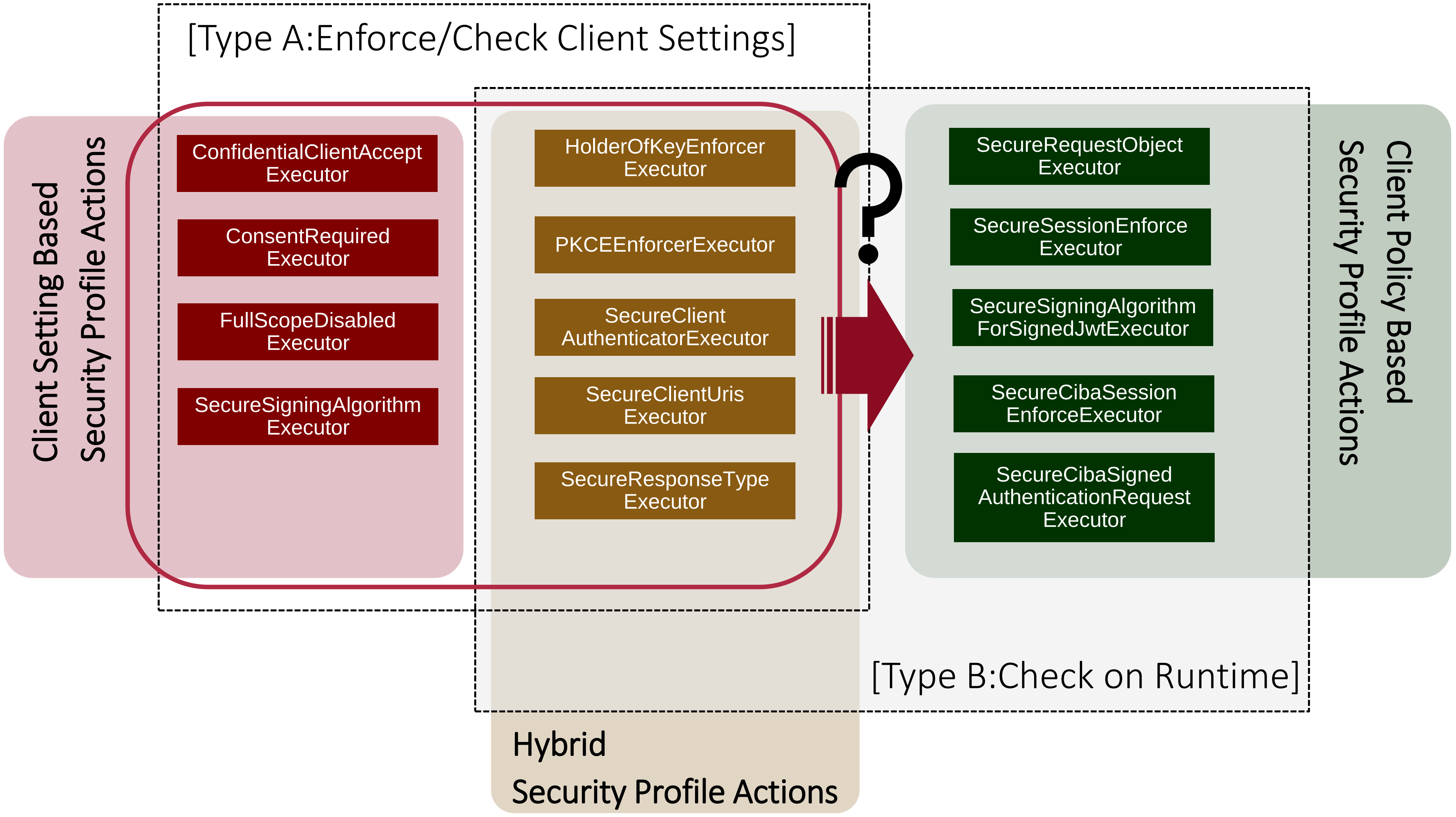
➔ Cannot be treated by type B executor.

In logics after executing client policies, some point refers client settings on the code path.

➔ Can be treated by type B executor :

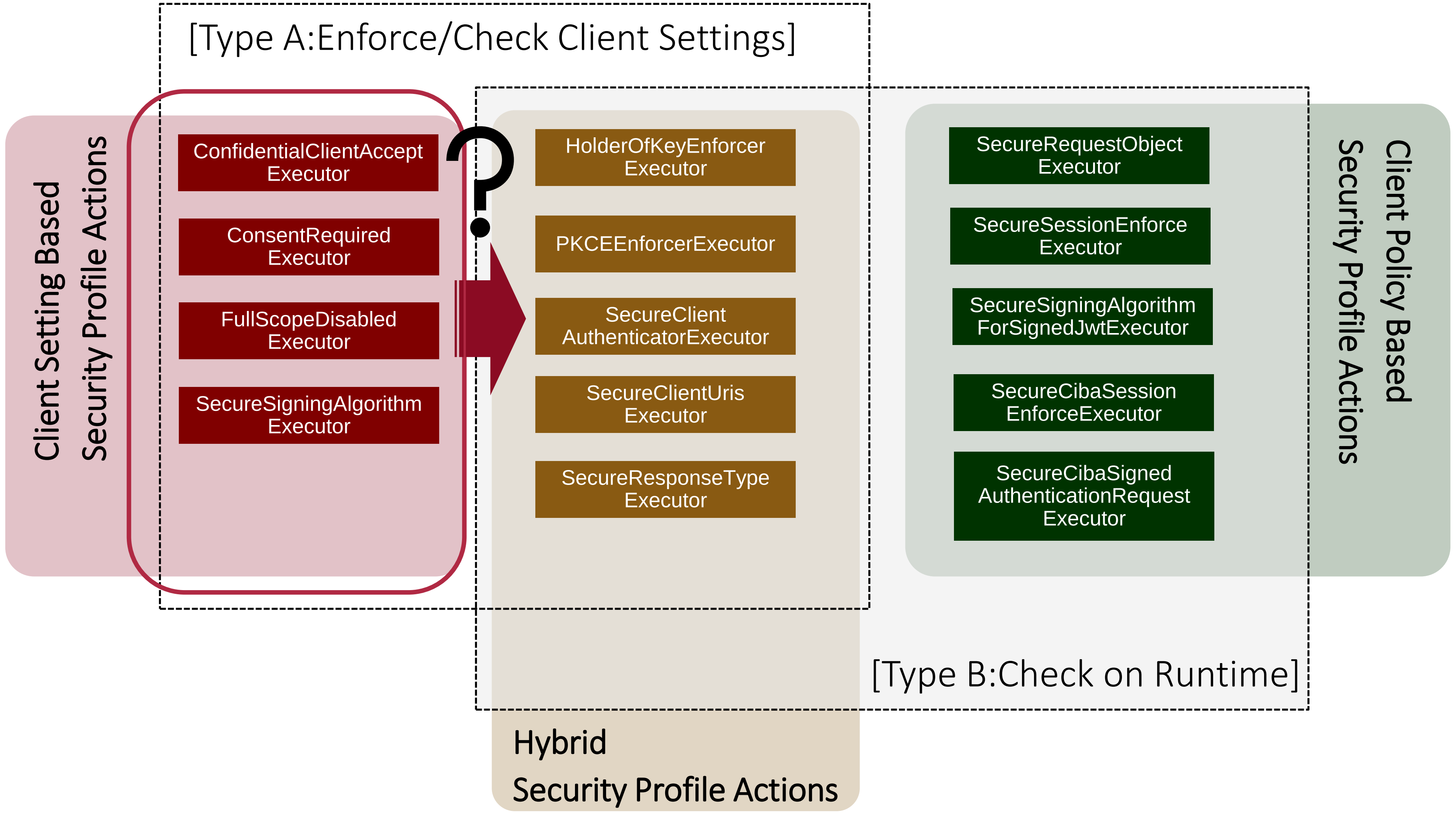
- Executor includes a functional interface implementation
- In logics after executing client policies, such points refer to this functional interface implementation instead of refer to client settings.

Plan 1 : Policy Oriented


[Type A:Enforce/Check Client Settings]
Client Setting Based
Security Profile Actions
ConfidentialClientAccept Executor
ConsentRequired Executor
FullScopeDisabled Executor
SecureSigningAlgorithm Executor
HolderOfKeyEnforcer Executor
PKCEEnforcerExecutor
SecureClient AuthenticatorExecutor
SecureClientUris Executor
SecureResponseType Executor
?
SecureRequestObject Executor
SecureSessionEnforce Executor
SecureSigningAlgorithm ForSignedJwtExecutor
SecureCibaSession EnforceExecutor
SecureCibaSigned AuthenticationRequest Executor
Client Policy Based
Security Profile Actions
[Type B:Check on Runtime]
Hybrid
Security Profile Actions

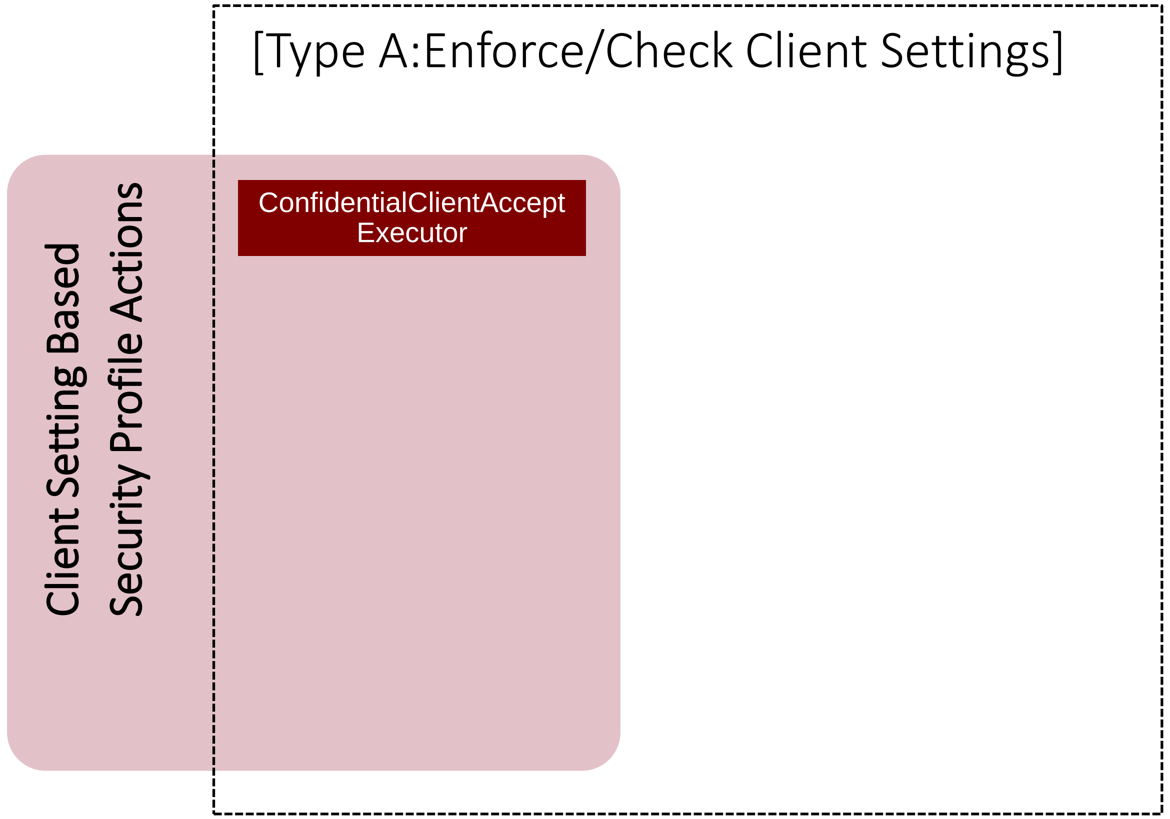
PROPOSED DRAFT

Plan 2 : Hybrid Oriented


[Type A:Enforce/Check Client Settings]
Client Setting Based
Security Profile Actions
ConfidentialClientAccept Executor
ConsentRequired Executor
FullScopeDisabled Executor
SecureSigningAlgorithm Executor
?
HolderOfKeyEnforcer Executor
PKCEEnforcerExecutor
SecureClient AuthenticatorExecutor
SecureClientUris Executor
SecureResponseType Executor
SecureRequestObject Executor
SecureSessionEnforce Executor
SecureSigningAlgorithm ForSignedJwtExecutor
SecureCibaSession EnforceExecutor
SecureCibaSigned AuthenticationRequest Executor
Client Policy Based
Security Profile Actions
[Type B:Check on Runtime]
Hybrid
Security Profile Actions

PROPOSED DRAFT

Strategy


[Type A:Enforce/Check Client Settings]
Client Setting Based Security Profile Actions
ConfidentialClientAccept Executor

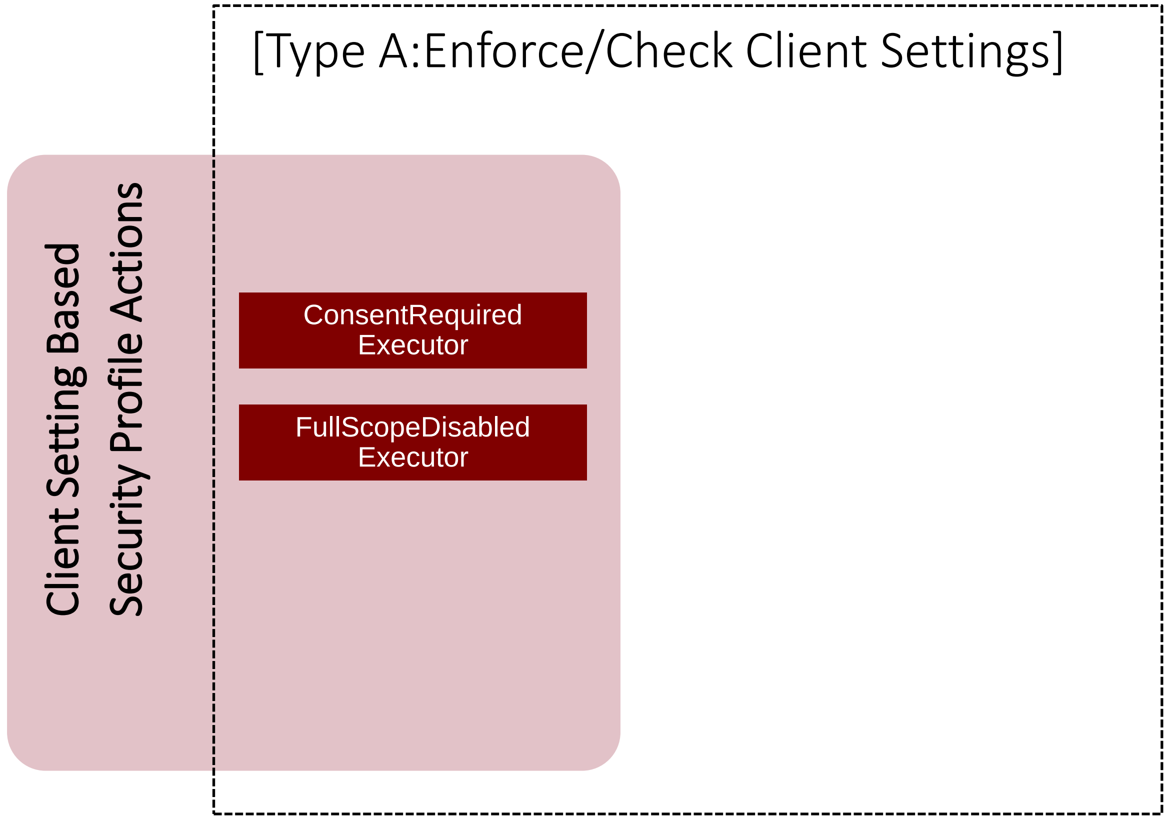
Keep the following executors be type A :

- By its nature, it requires to refer to a client setting because only the content of the request from the client on the endpoint is not adequate to execute its logics.

E.g.

ConfidentialClientAcceptExecutor

Strategy



Keep the following executors be type A :

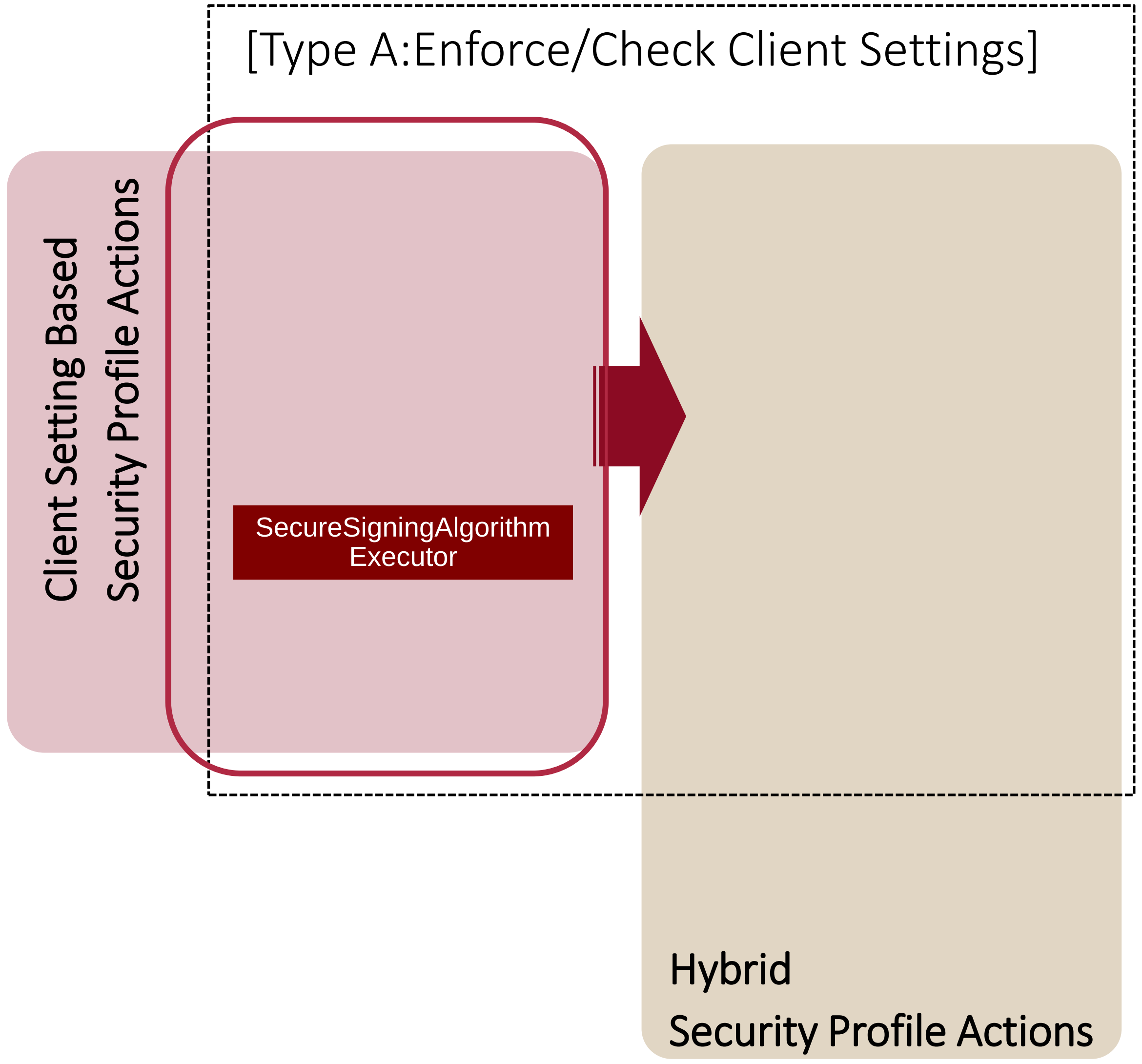
- By its nature, the client settings that the executor treats are referred from several point in the code path too much broad, especially points that are not relevant to endpoints' logic that receives client's requests.

E.g.

ConsentRequiredExecutor

FullScopeDisabledExecutor

Strategy

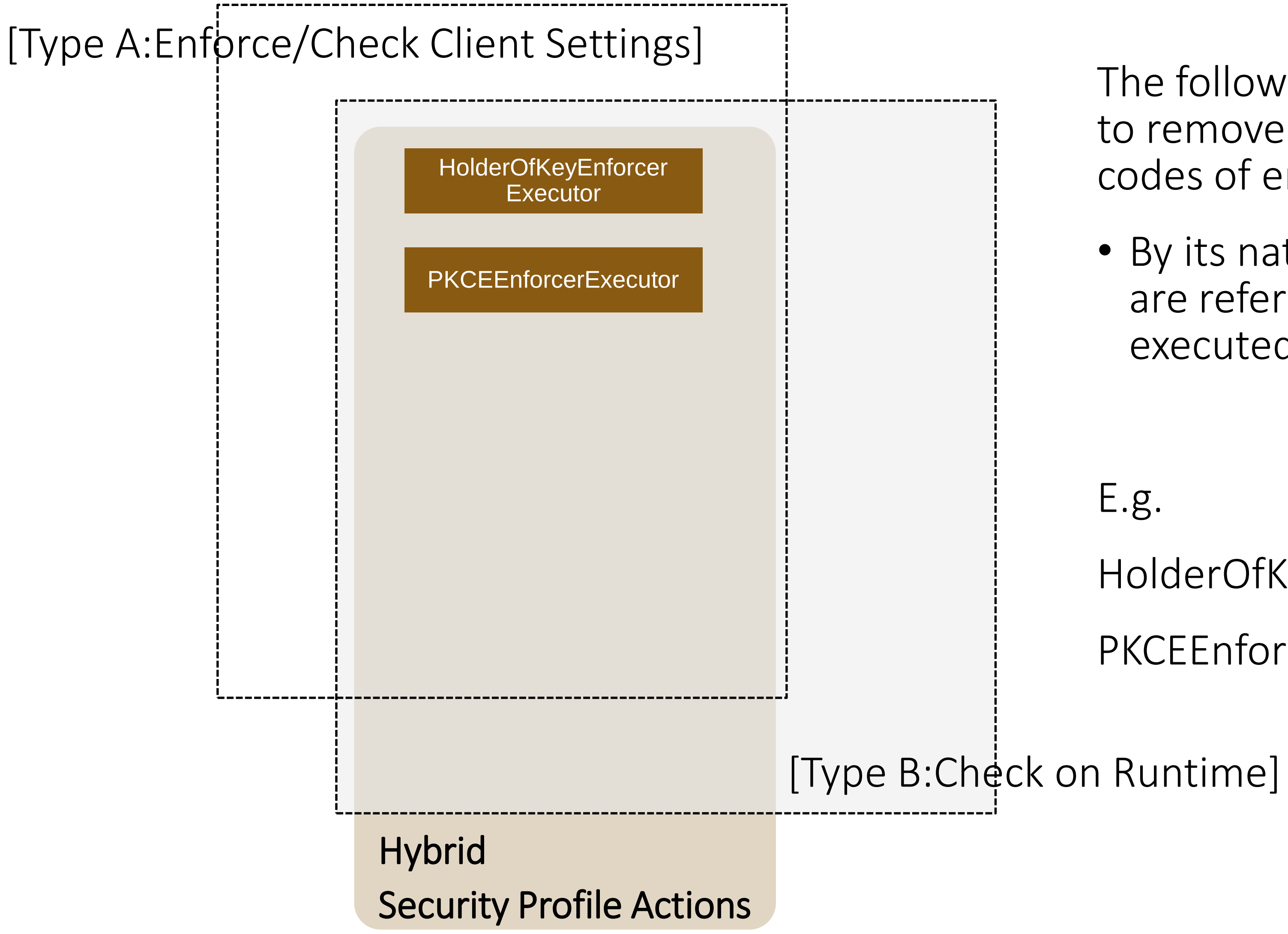


Other type A executors migrated to type A & B by considering backward compatibility.

E.g.

SecureSigningAlgorithmExecutor

Strategy


[Type A:Enforce/Check Client Settings]
HolderOfKeyEnforcer Executor
PKCEEnforcerExecutor
[Type B:Check on Runtime]
Hybrid
Security Profile Actions

The following type B executors needs to be revised to remove logics referring client settings from codes of endpoints:

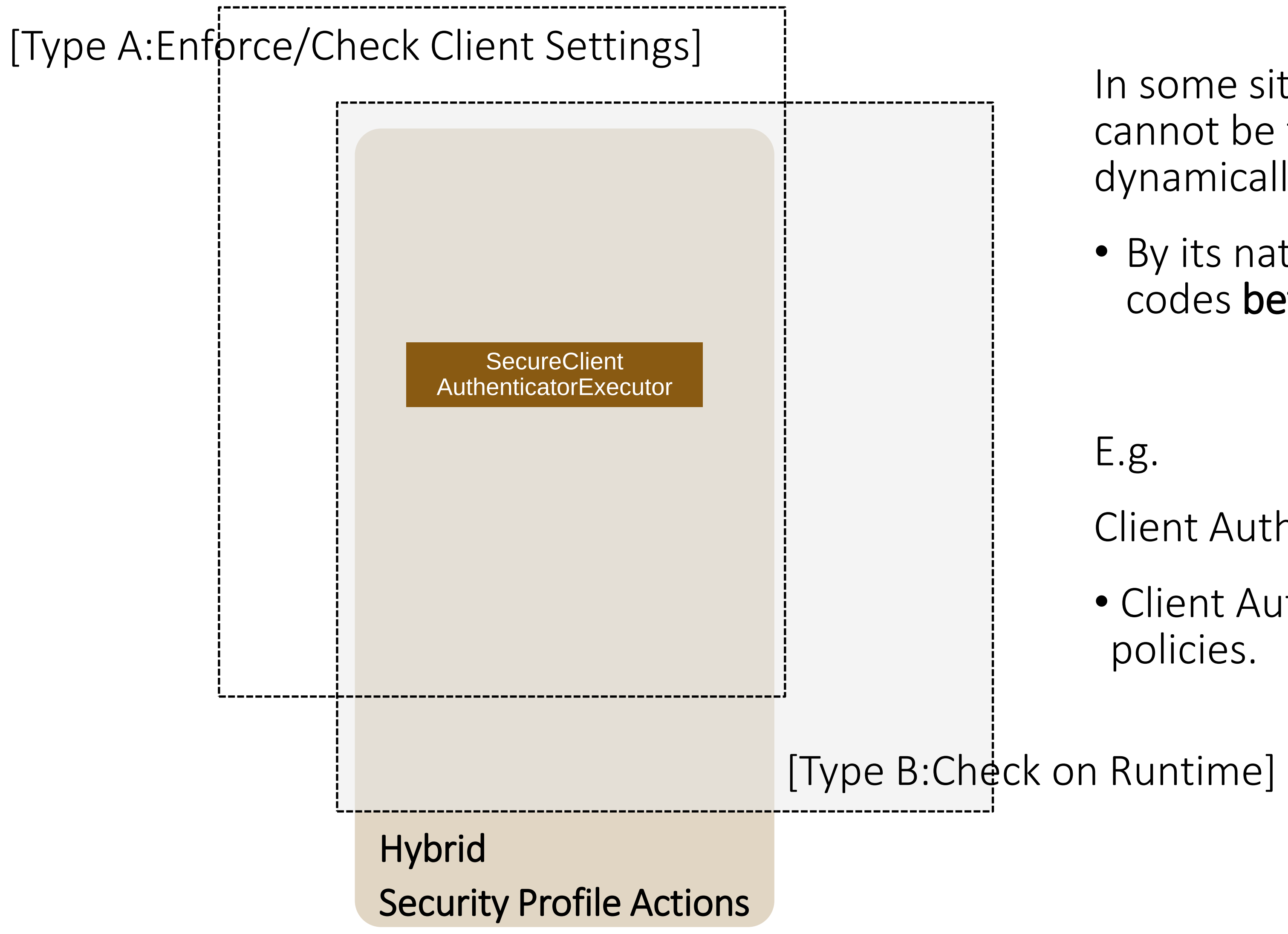
- By its nature, client settings this executor treats are referred from codes after client polices are executed.

E.g.

HolderOfKeyEnforcerExecutor

PKCEEnforcerExecutor

Strategy


[Type A:Enforce/Check Client Settings]
SecureClient AuthenticatorExecutor
[Type B:Check on Runtime]
Hybrid
Security Profile Actions

In some situation, the following client settings cannot be treated by changing profiles dynamically per client's request :

- By its nature, client settings are referred from codes **before** client polices are executed.

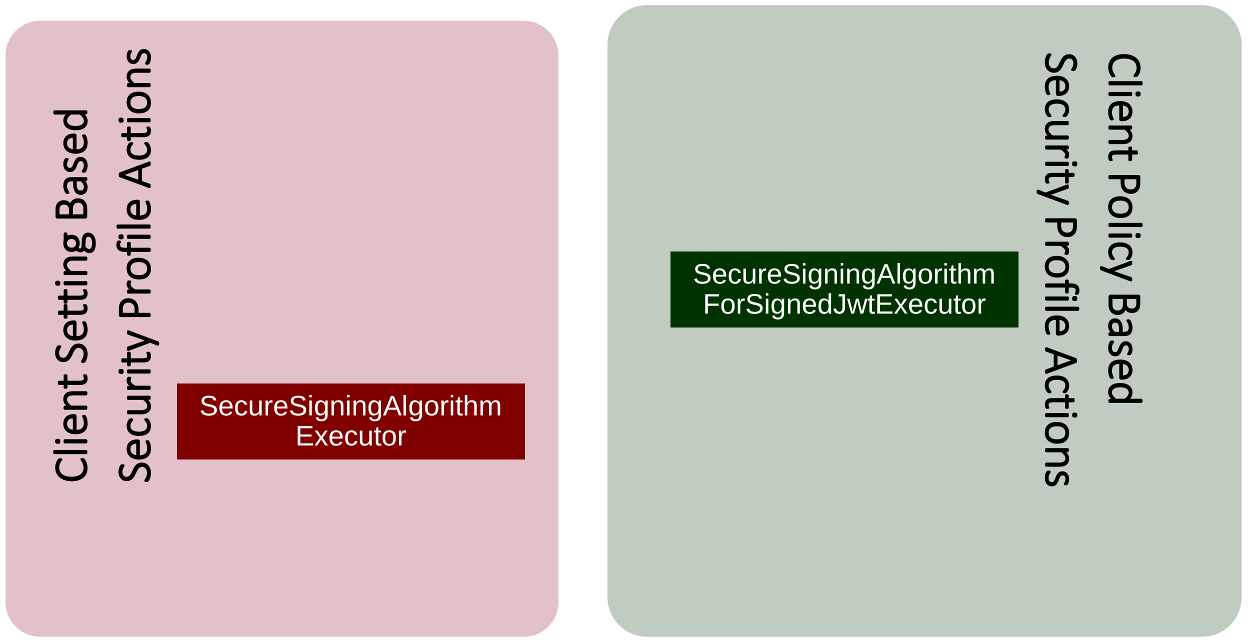
E.g.

Client Authenticator Type

- Client Authentication is executed before client policies.

Strategy

[Type A:Enforce/Check Client Settings]



[Type B:Check on Runtime]

In some situation, the following client settings cannot be treated by changing profiles dynamically per client's request :

- By its nature, client settings are referred from codes **before** client polices are executed.

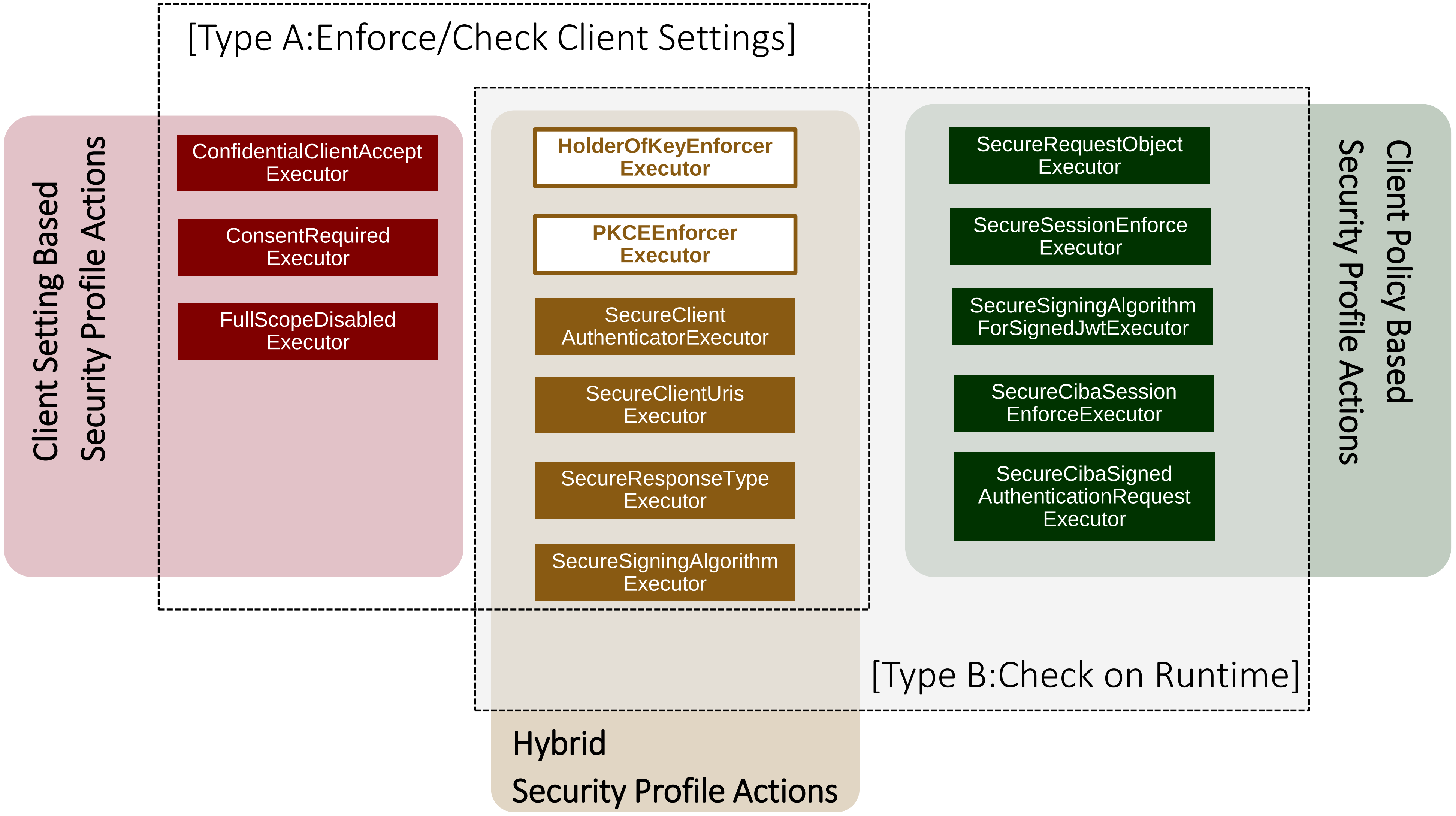
E.g.

JWS Signature Algorithm used for client authentication and request objects

- Client Authentication is executed before client policies.
- Parsing request object is executed before client policies.

PROPOSED DRAFT

Future : Nearly Policy Oriented


[Type A:Enforce/Check Client Settings]
Client Setting Based
Security Profile Actions
ConfidentialClientAccept Executor
ConsentRequired Executor
FullScopeDisabled Executor
HolderOfKeyEnforcer Executor
PKCEEnforcer Executor
SecureClient AuthenticatorExecutor
SecureClientUris Executor
SecureResponseType Executor
SecureSigningAlgorithm Executor
SecureRequestObject Executor
SecureSessionEnforce Executor
SecureSigningAlgorithm ForSignedJwtExecutor
SecureCibaSession EnforceExecutor
SecureCibaSigned AuthenticationRequest Executor
Client Policy Based
Security Profile Actions
[Type B:Check on Runtime]
Hybrid
Security Profile Actions

PROPOSED DRAFT

Future : Adding New Executor

- Avoid referring client settings

Tries to implement it as Type B : Check on Runtime executor.

- Get rid of logics referring client settings from codes of endpoints :

If using client settings, avoid such settings that are referred before client policies execution and introduce functional interface implementation to be referred after client policies execution.

- Recognizing limitation

If using such client settings, avoid referring such client settings when changing client profiles dynamically per client's request.

END

PROPOSED DRAFT