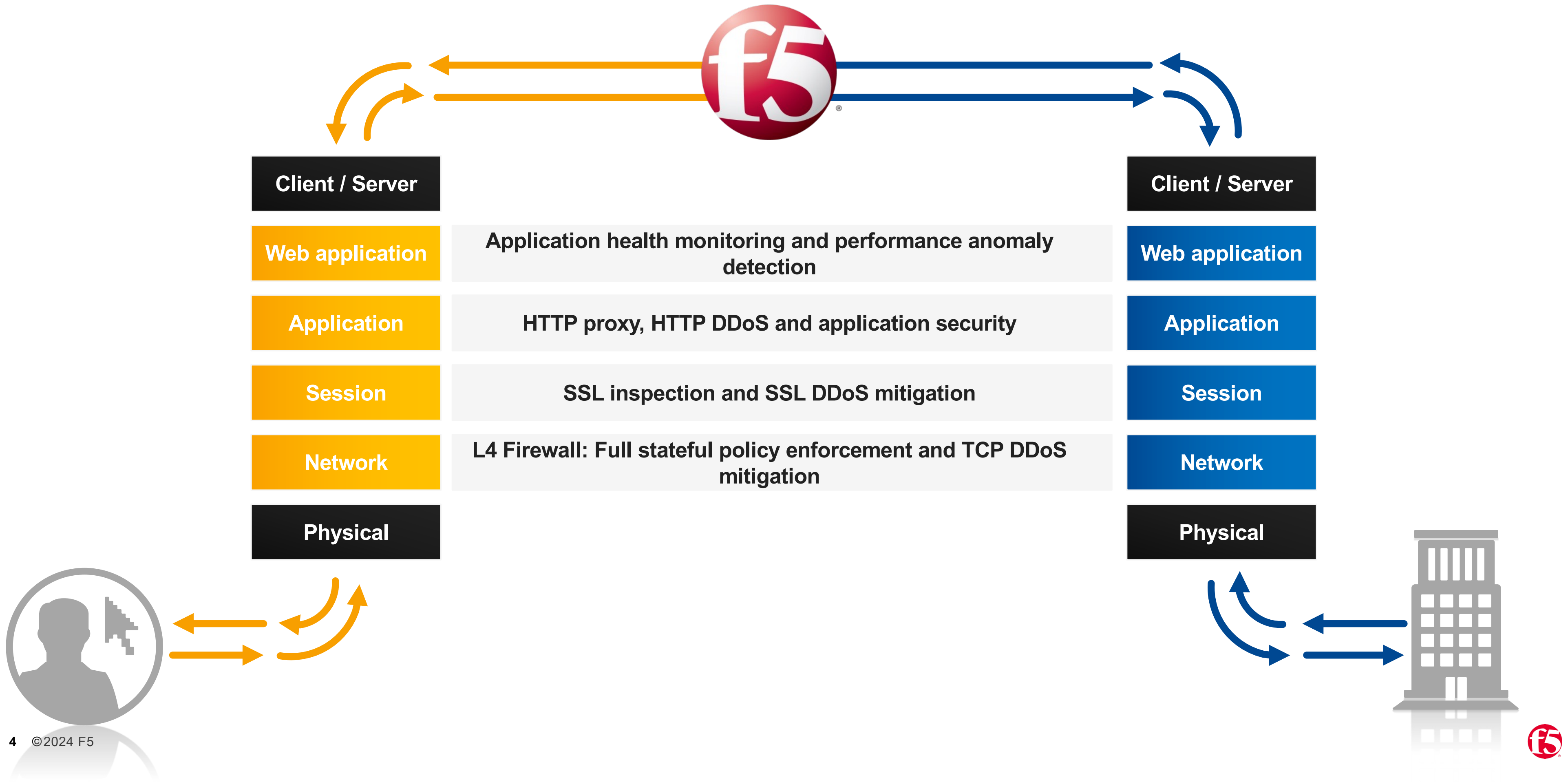
BIG-IP APM Best Practices

Anthony Graber – Solutions Engineer, DISA

Agenda

BIG-IP APM Overview

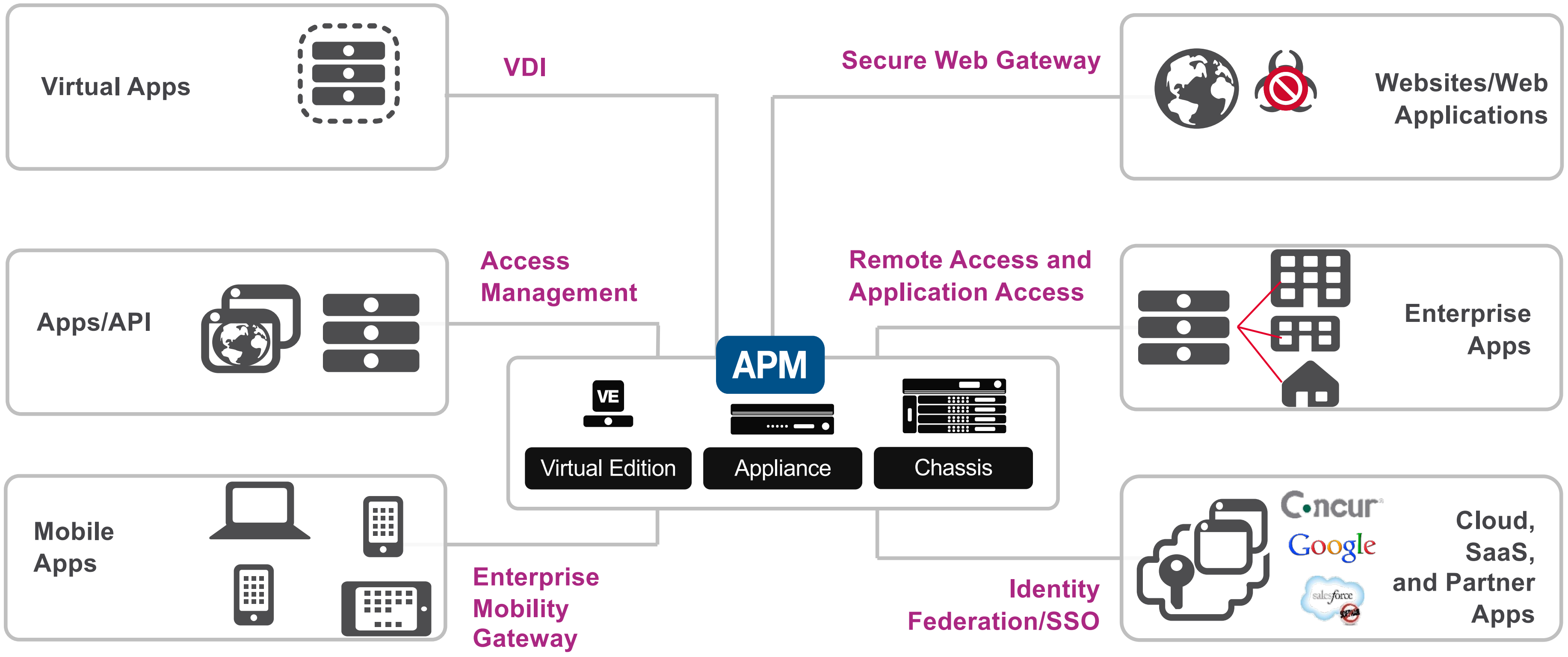
Full Proxy Architecture



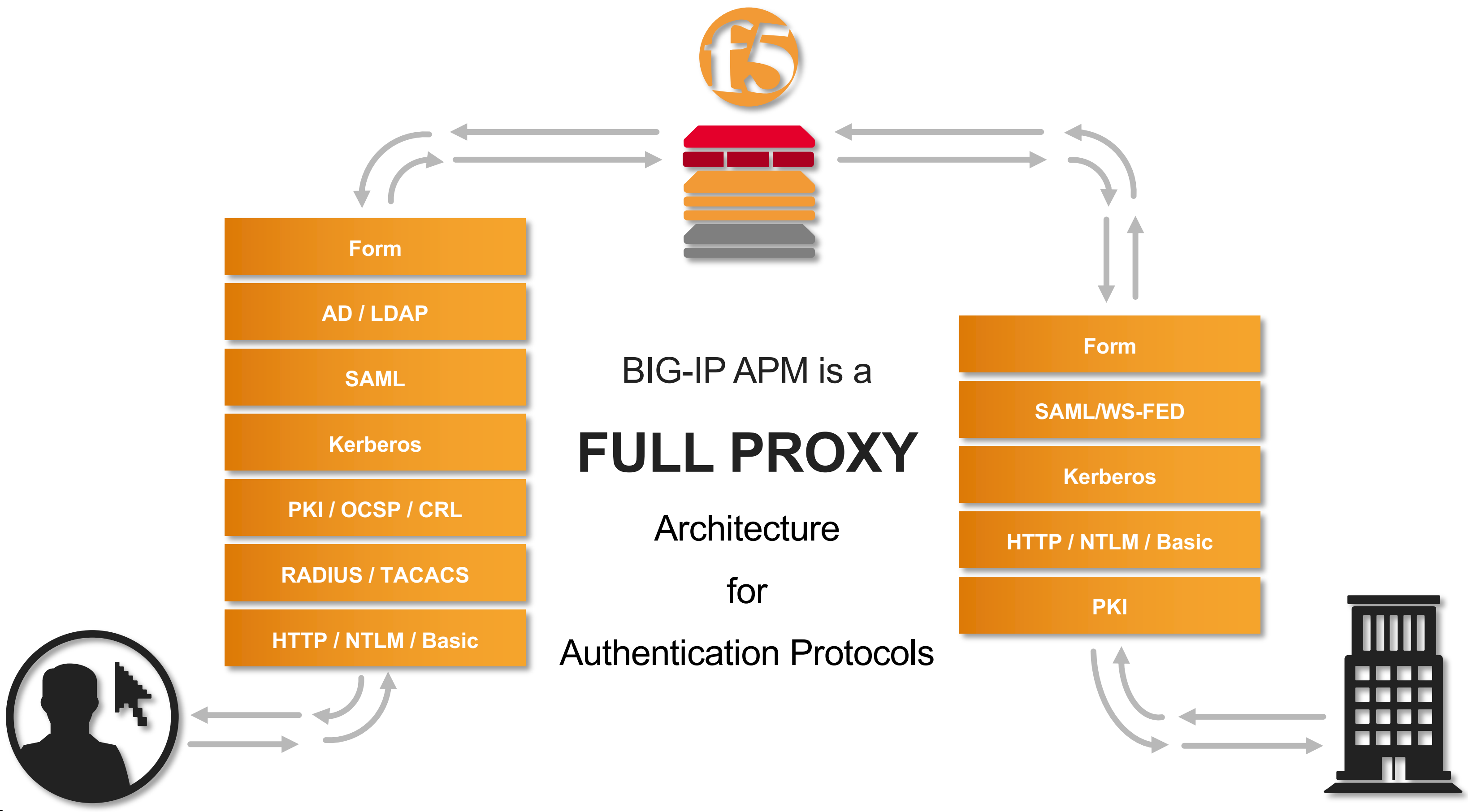
Client / Server
Web application
Application
Session
Network
Physical
Application health monitoring and performance anomaly detection
HTTP proxy, HTTP DDoS and application security
SSL inspection and SSL DDoS mitigation
L4 Firewall: Full stateful policy enforcement and TCP DDoS mitigation
Client / Server
Web application
Application
Session
Network
Physical

F5's Remote/Application Access Solutions

AUTHENTICATION, AUTHORIZATION, REMOTE ACCESS AND SSO TO ALL APPLICATIONS WITH CENTRALIZED ACCESS POLICY ENFORCEMENT USING ACCESS POLICY MANAGER (APM)



Full Proxy Architecture for Authentication



BIG-IP APM is a

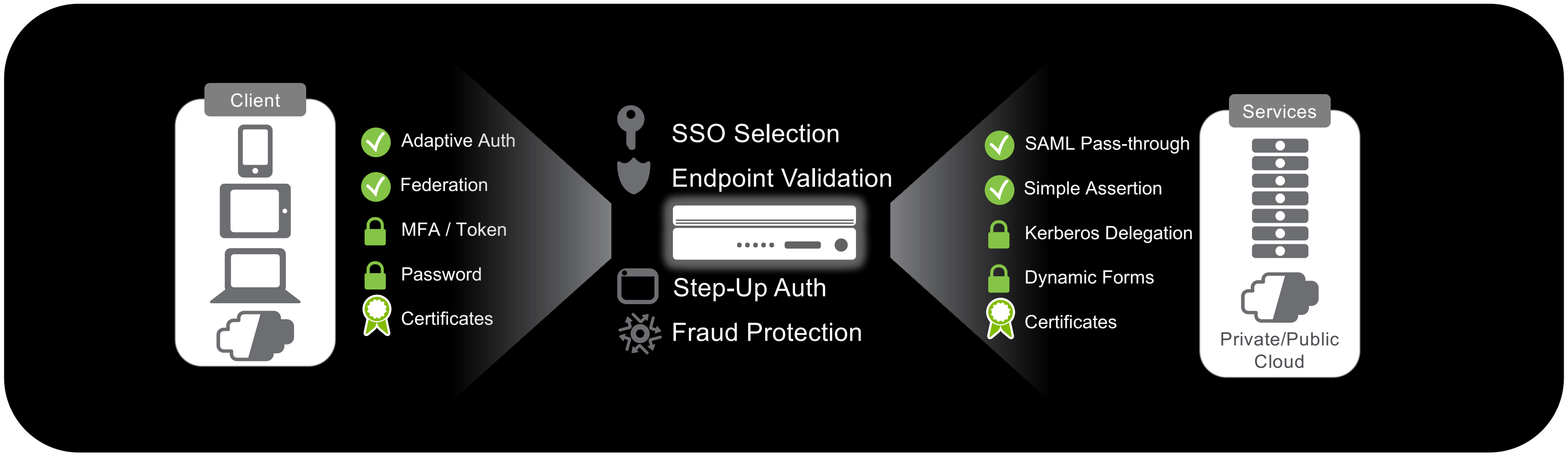
FULL PROXY

Architecture

for

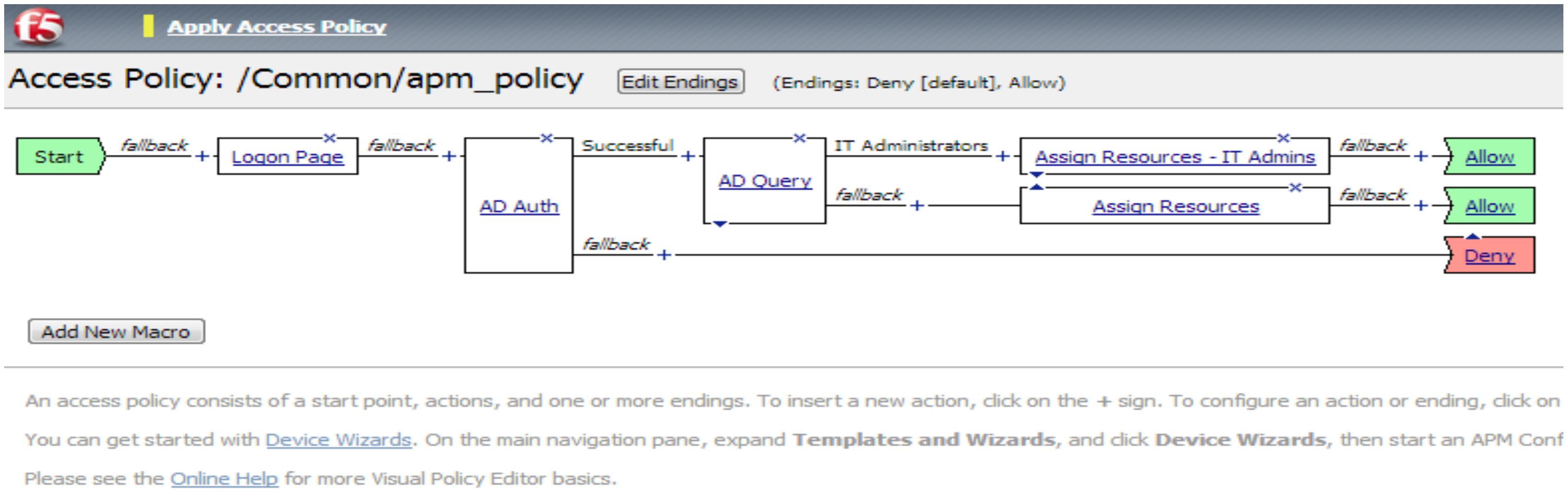
Authentication Protocols

F5 Access Policy Manager (APM)



- Transform one type of authentication into another so an app may understand and use it without installing additional agents
- Allow flexible selection of SSO technique appropriate to the application
- Allow for centralized session control of all applications

Access Policy Design



Access Policy: /Common/apm_policy

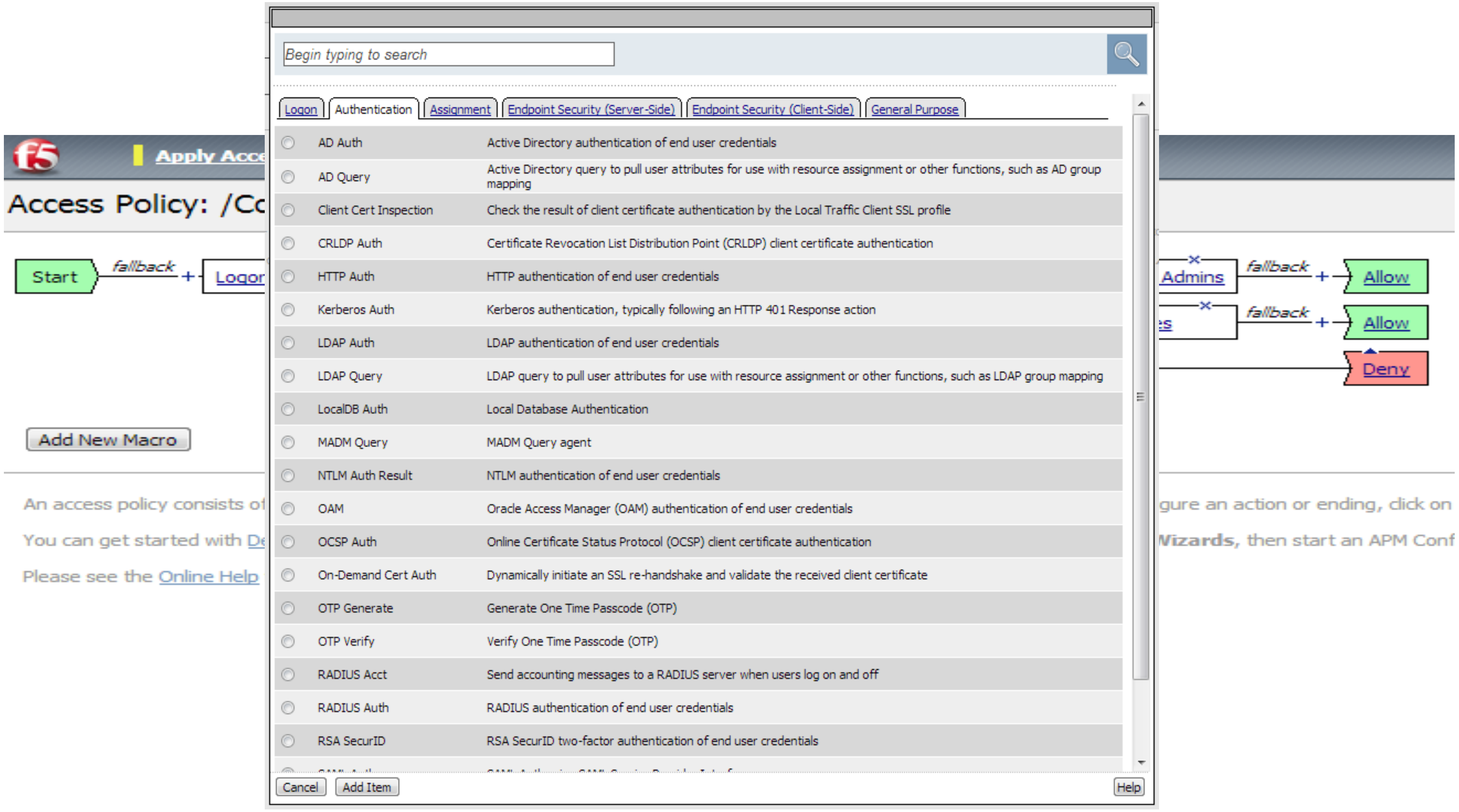
(Endings: Deny [default], Allow)

An access policy consists of a start point, actions, and one or more endings. To insert a new action, click on the + sign. To configure an action or ending, click on

You can get started with Device Wizards. On the main navigation pane, expand **Templates and Wizards**, and click **Device Wizards**, then start an APM Conf

Please see the Online Help for more Visual Policy Editor basics.

Access Policy Design



Begin typing to search

Logon | Authentication | Assignment | Endpoint Security (Server-Side) | Endpoint Security (Client-Side) | General Purpose

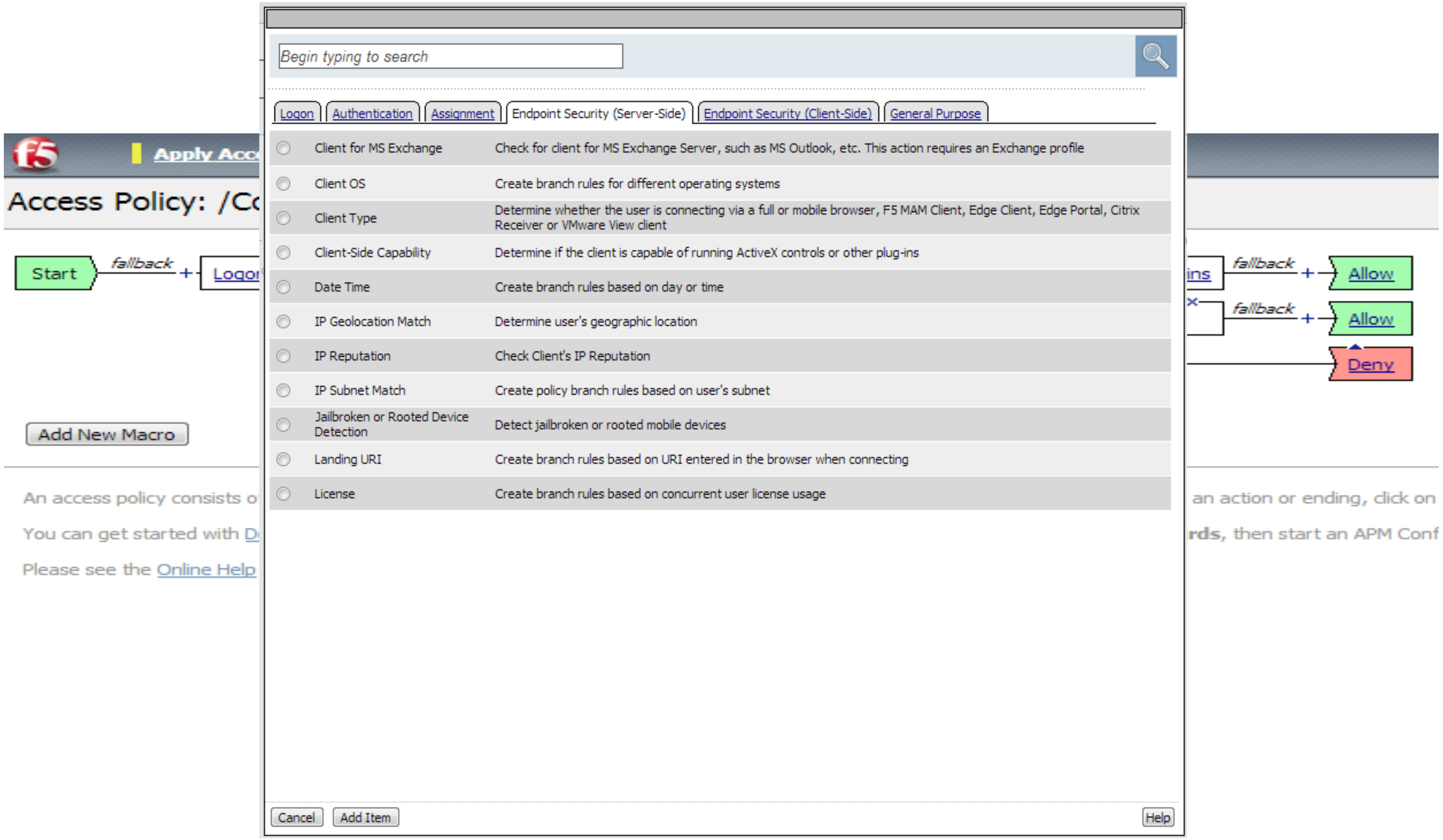
AD Auth	Active Directory authentication of end user credentials
AD Query	Active Directory query to pull user attributes for use with resource assignment or other functions, such as AD group mapping
Client Cert Inspection	Check the result of client certificate authentication by the Local Traffic Client SSL profile
CRLDP Auth	Certificate Revocation List Distribution Point (CRLDP) client certificate authentication
HTTP Auth	HTTP authentication of end user credentials
Kerberos Auth	Kerberos authentication, typically following an HTTP 401 Response action
LDAP Auth	LDAP authentication of end user credentials
LDAP Query	LDAP query to pull user attributes for use with resource assignment or other functions, such as LDAP group mapping
LocalDB Auth	Local Database Authentication
MADM Query	MADM Query agent
NTLM Auth Result	NTLM authentication of end user credentials
OAM	Oracle Access Manager (OAM) authentication of end user credentials
OCSP Auth	Online Certificate Status Protocol (OCSP) client certificate authentication
On-Demand Cert Auth	Dynamically initiate an SSL re-handshake and validate the received client certificate
OTP Generate	Generate One Time Passcode (OTP)
OTP Verify	Verify One Time Passcode (OTP)
RADIUS Acct	Send accounting messages to a RADIUS server when users log on and off
RADIUS Auth	RADIUS authentication of end user credentials
RSA SecurID	RSA SecurID two-factor authentication of end user credentials

An access policy consists of … gure an action or ending, click on

You can get started with De … Wizards, then start an APM Conf

Please see the Online Help

Access Policy Design



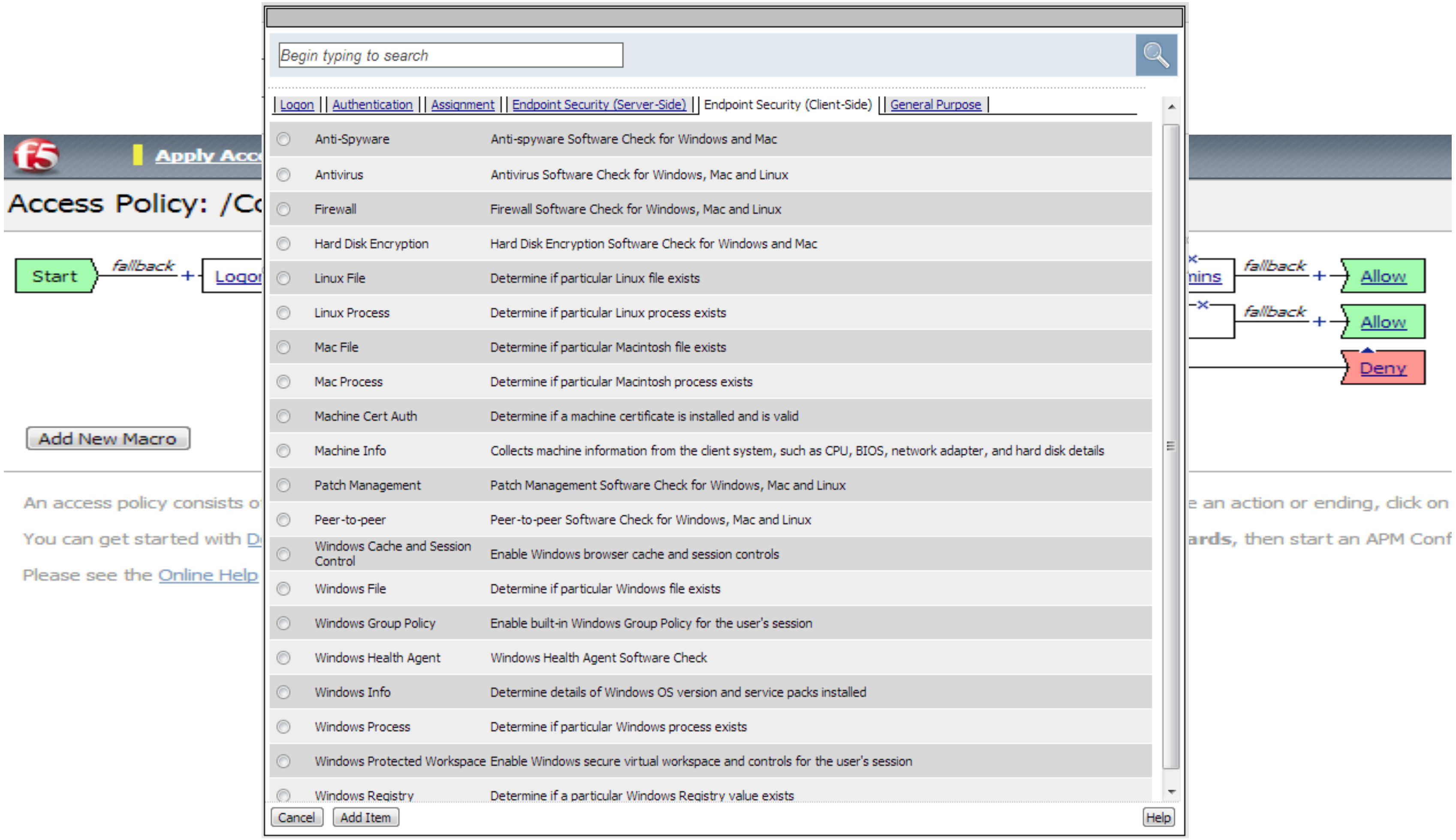
An access policy consists o … an action or ending, click on

You can get started with D … rds, then start an APM Conf

Please see the Online Help

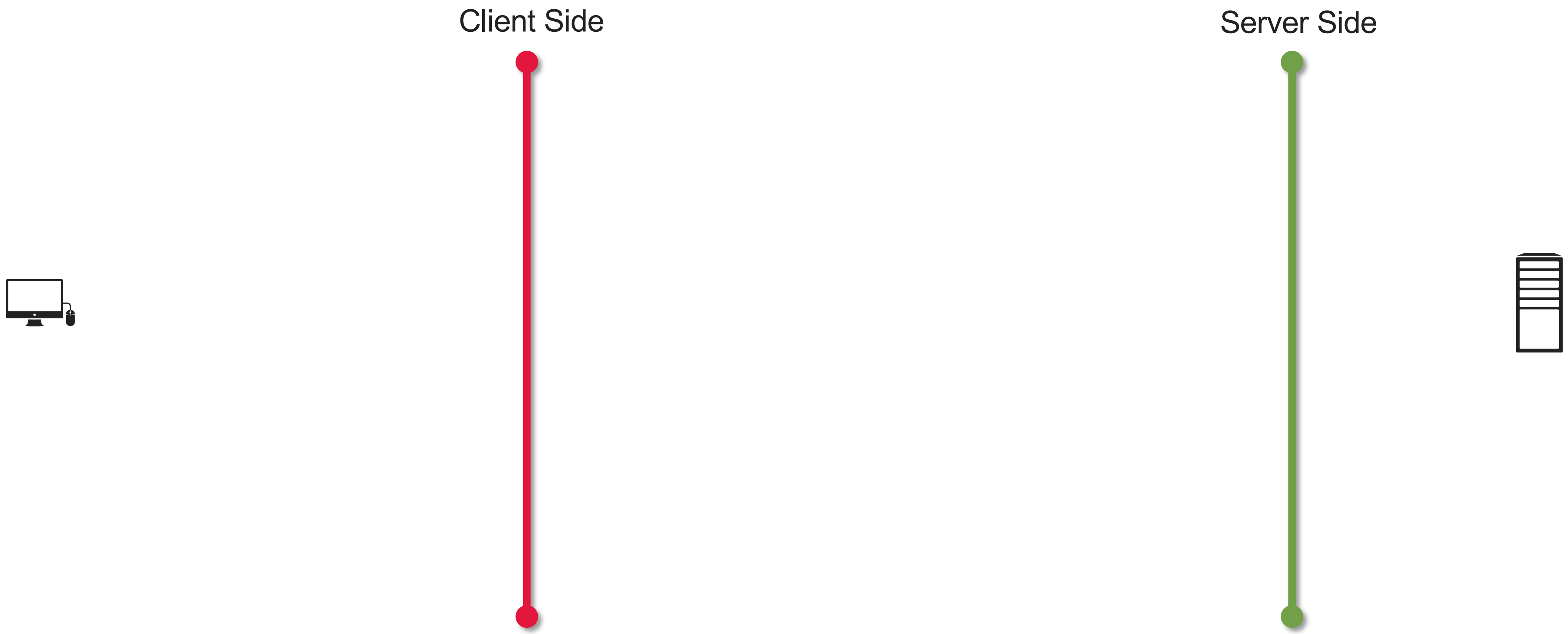
○	Client for MS Exchange	Check for client for MS Exchange Server, such as MS Outlook, etc. This action requires an Exchange profile
○	Client OS	Create branch rules for different operating systems
○	Client Type	Determine whether the user is connecting via a full or mobile browser, F5 MAM Client, Edge Client, Edge Portal, Citrix Receiver or VMware View client
○	Client-Side Capability	Determine if the client is capable of running ActiveX controls or other plug-ins
○	Date Time	Create branch rules based on day or time
○	IP Geolocation Match	Determine user's geographic location
○	IP Reputation	Check Client's IP Reputation
○	IP Subnet Match	Create policy branch rules based on user's subnet
○	Jailbroken or Rooted Device Detection	Detect jailbroken or rooted mobile devices
○	Landing URI	Create branch rules based on URI entered in the browser when connecting
○	License	Create branch rules based on concurrent user license usage

Access Policy Design

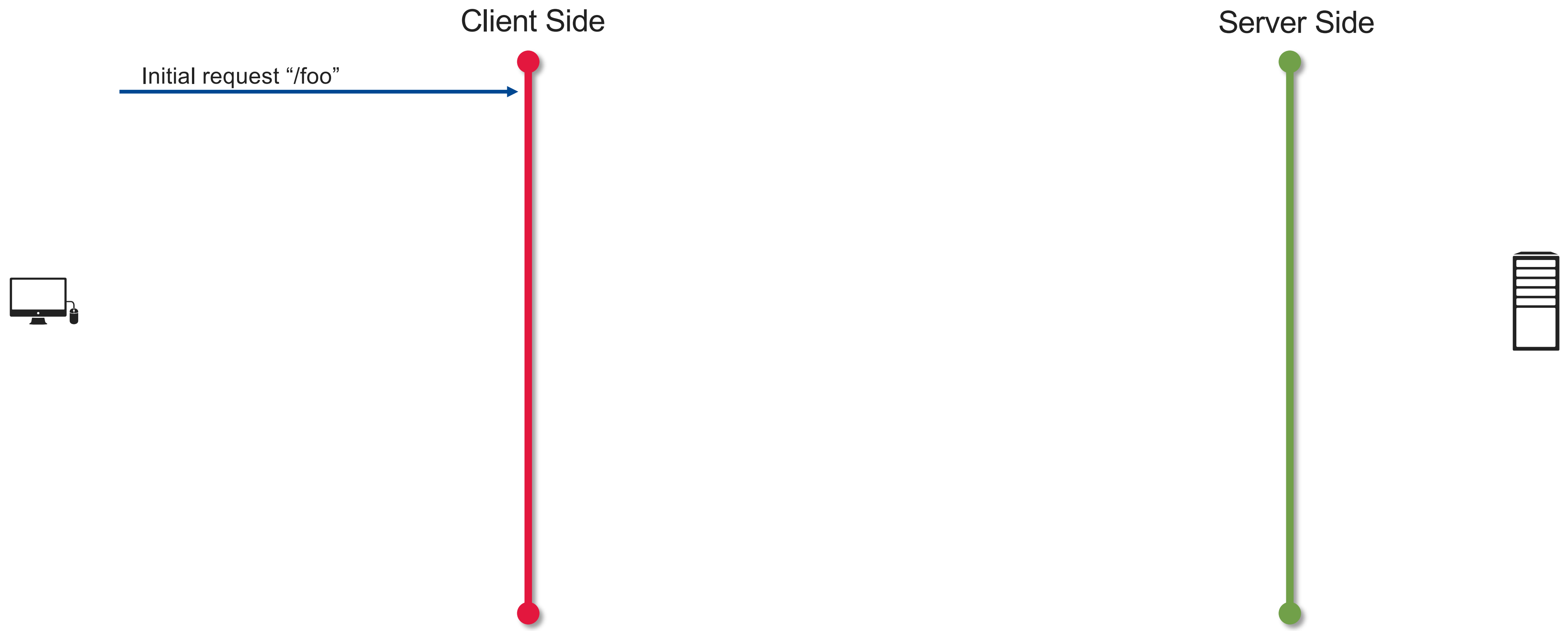

Apply Acc
Access Policy: /Co
Start
fallback
Logon
nins
Allow
Deny
Add New Macro
An access policy consists o
You can get started with D
Please see the Online Help
e an action or ending, click on
ards, then start an APM Conf
Begin typing to search
Logon | Authentication | Assignment | Endpoint Security (Server-Side) | Endpoint Security (Client-Side) | General Purpose

Anti-Spyware	Anti-spyware Software Check for Windows and Mac
Antivirus	Antivirus Software Check for Windows, Mac and Linux
Firewall	Firewall Software Check for Windows, Mac and Linux
Hard Disk Encryption	Hard Disk Encryption Software Check for Windows and Mac
Linux File	Determine if particular Linux file exists
Linux Process	Determine if particular Linux process exists
Mac File	Determine if particular Macintosh file exists
Mac Process	Determine if particular Macintosh process exists
Machine Cert Auth	Determine if a machine certificate is installed and is valid
Machine Info	Collects machine information from the client system, such as CPU, BIOS, network adapter, and hard disk details
Patch Management	Patch Management Software Check for Windows, Mac and Linux
Peer-to-peer	Peer-to-peer Software Check for Windows, Mac and Linux
Windows Cache and Session Control	Enable Windows browser cache and session controls
Windows File	Determine if particular Windows file exists
Windows Group Policy	Enable built-in Windows Group Policy for the user's session
Windows Health Agent	Windows Health Agent Software Check
Windows Info	Determine details of Windows OS version and service packs installed
Windows Process	Determine if particular Windows process exists
Windows Protected Workspace	Enable Windows secure virtual workspace and controls for the user's session
Windows Registry	Determine if a particular Windows Registry value exists

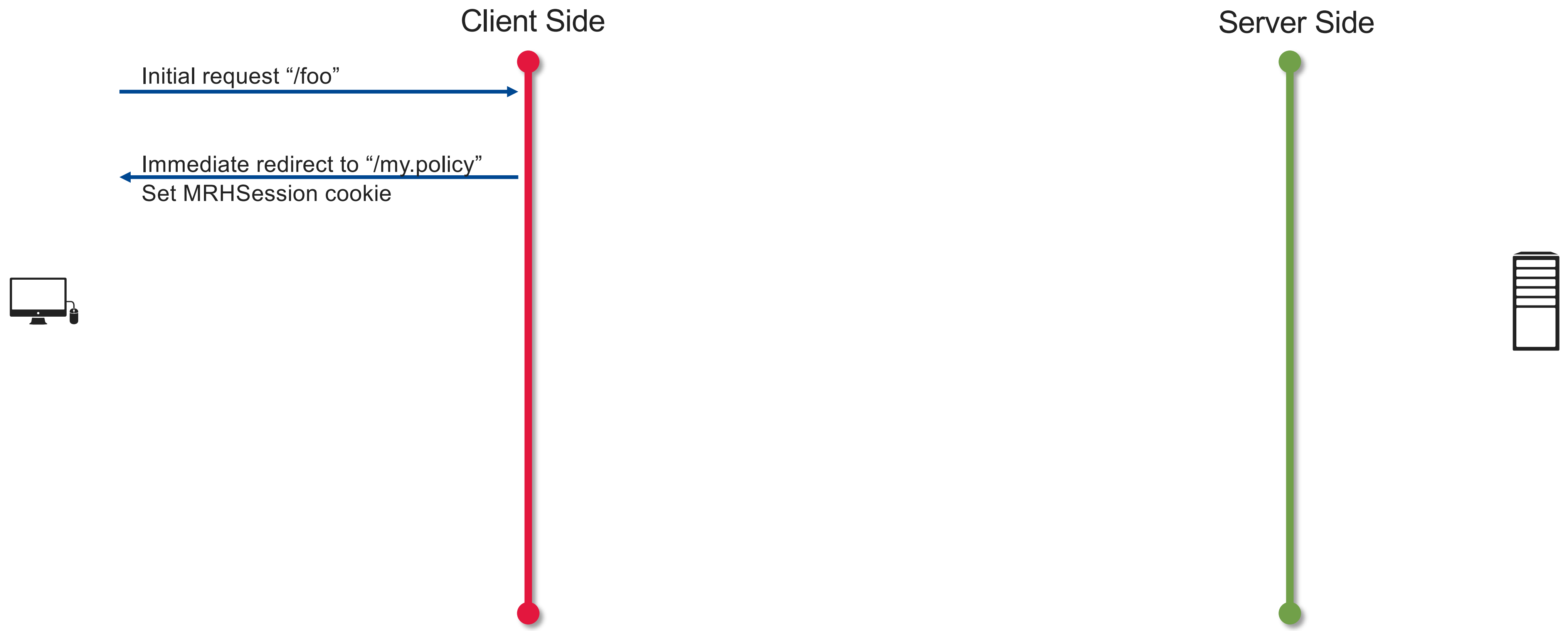
General protocol flow



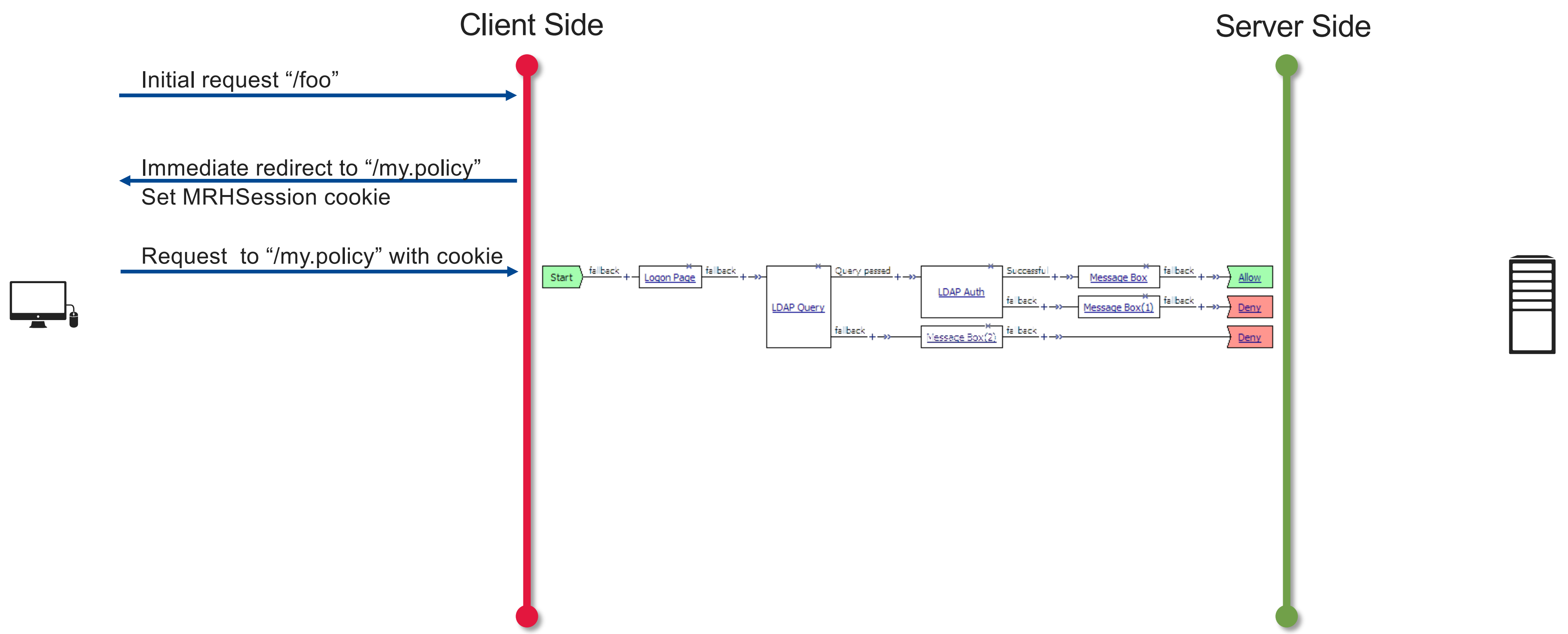
General protocol flow



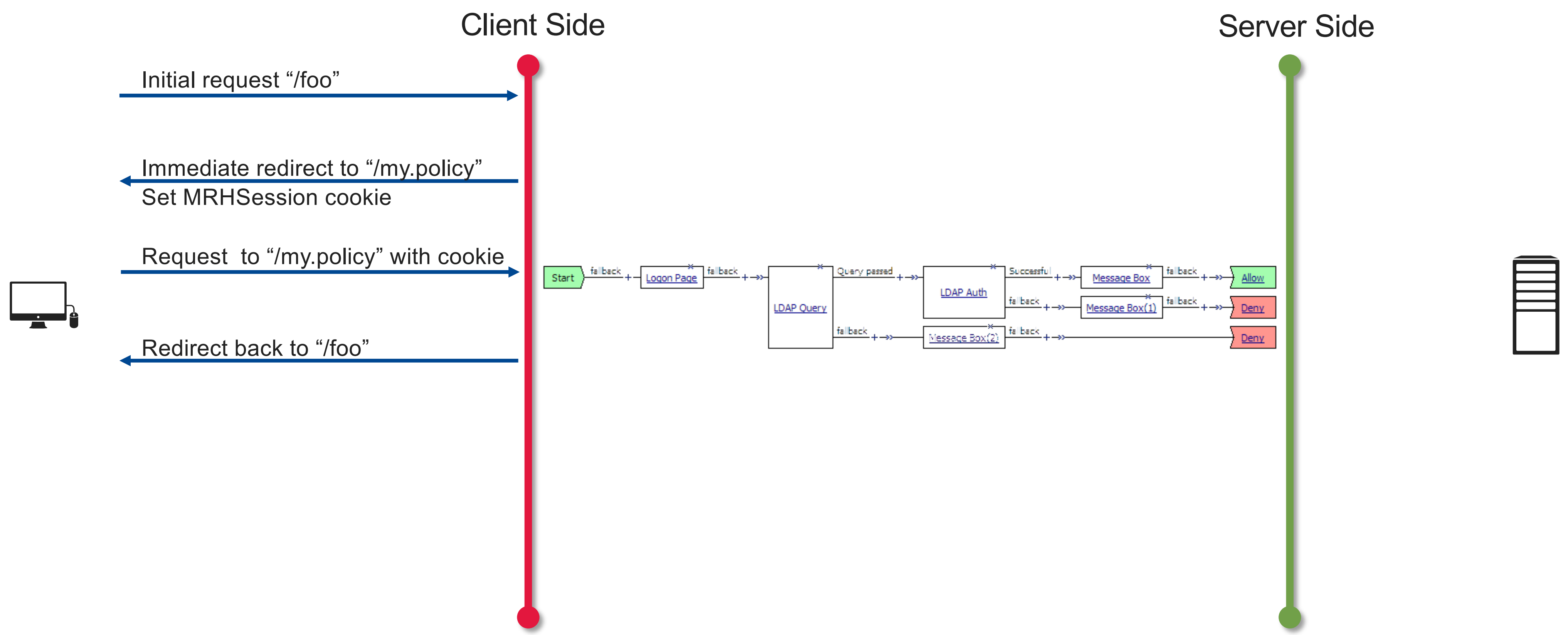
General protocol flow



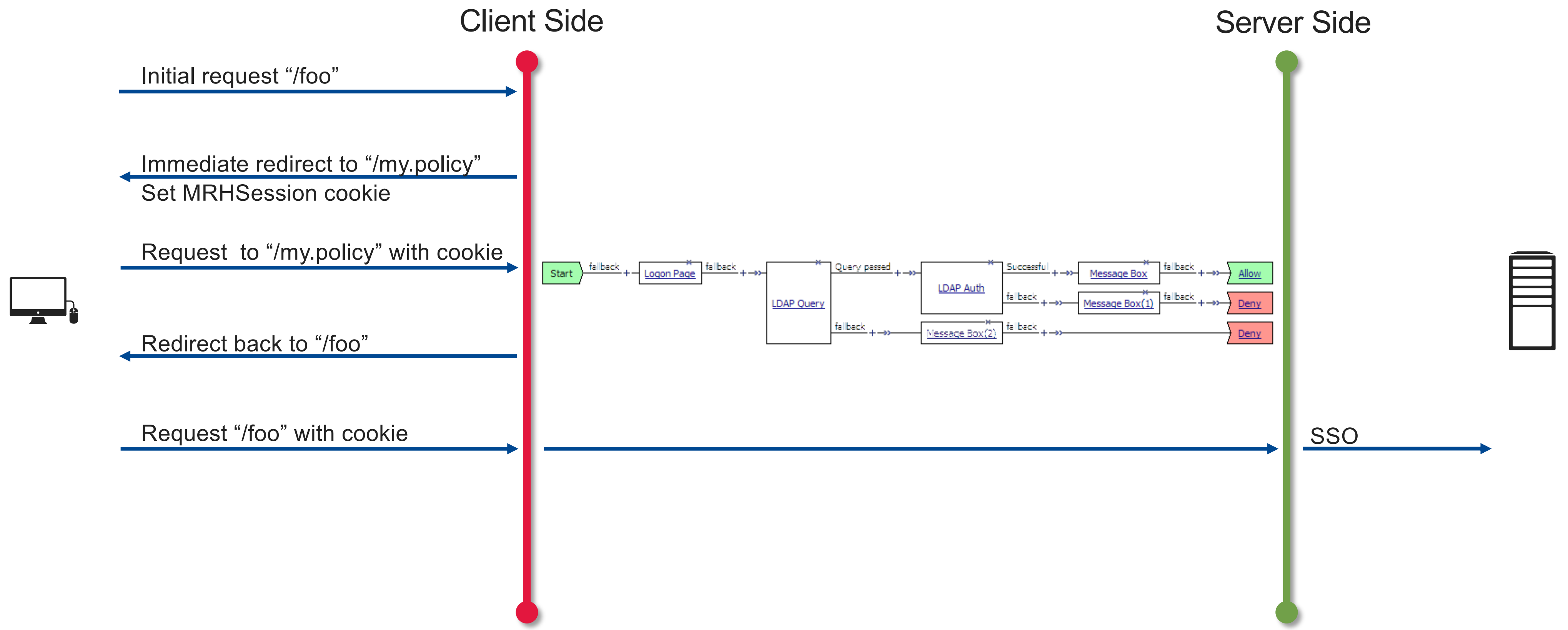
General protocol flow



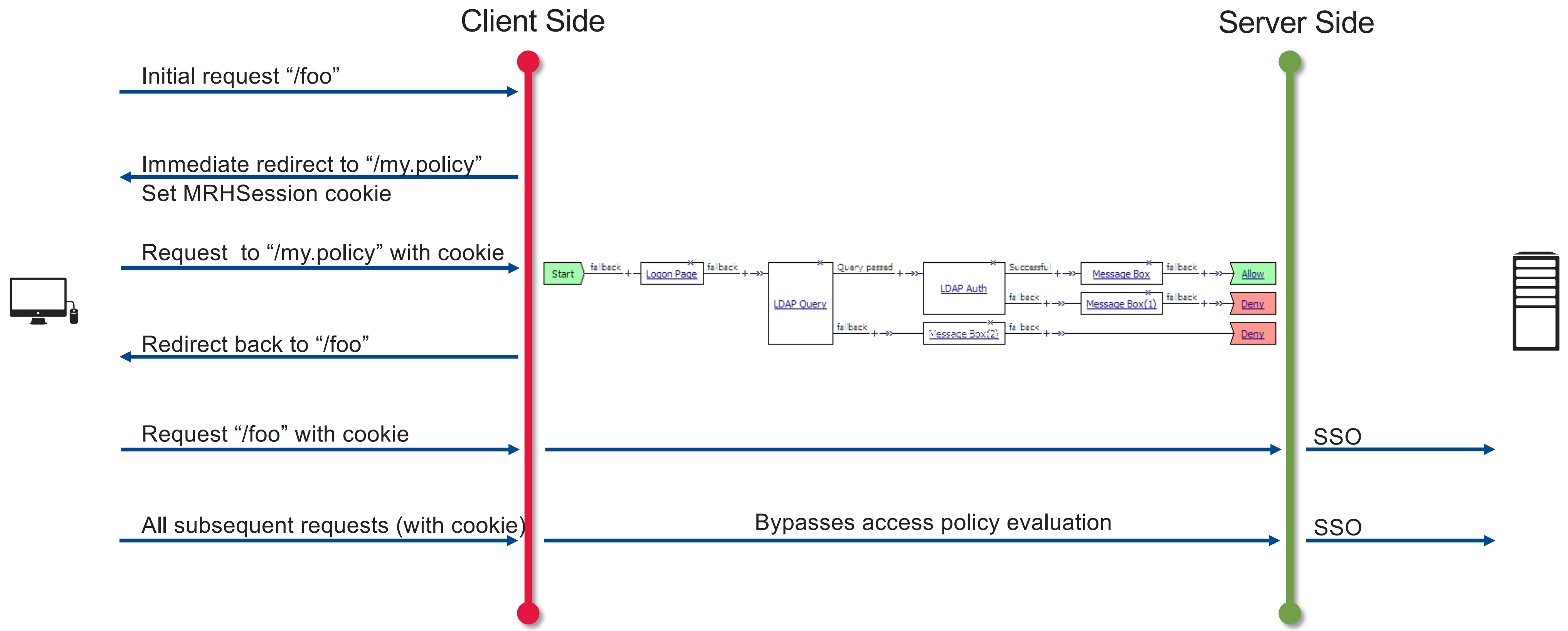
General protocol flow



General protocol flow



General protocol flow



Smart Cards and APM

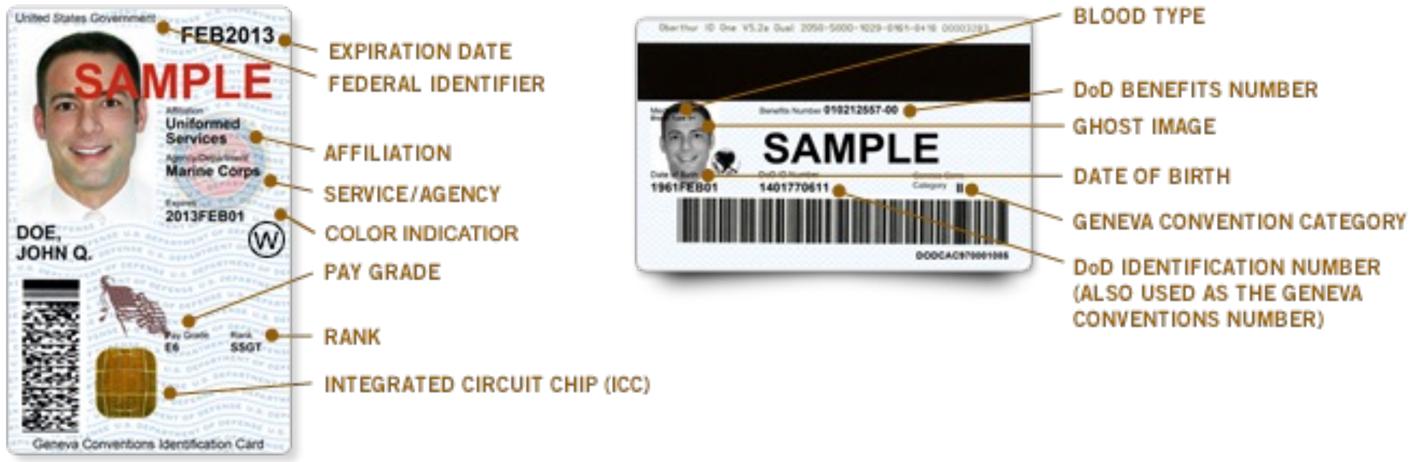
What is a Common Access Card (CAC)/Personal Identity Verification (PIV)?

- A CAC or PIV is a collection of public and private keys stored on a Smart Card issued by a Public Key Infrastructure (PKI).
- The certificates can be used to establish a mutual trust between the user and the server.
- A client, such as a browser, can be used to provide this credential to a server or website.
- When a smart card is inserted into a machine the public certs are copied to the system’s certificate store.

What is a Common Access Card (CAC)/Personal Identity Verification (PIV)?

- A CAC or PIV is a collection of public and private keys stored on a sm
- Th ween the use
- A c dential to a ser
- Wh rts are co

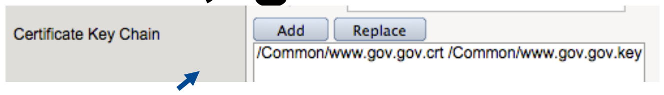
CAC EXAMPLE



Configuration Requirements

- Certificate bundle to authenticate the certificate from the CAC/PIV.
- Online Certificate Status Protocol (OCSP) server, Certificate Revocation List (CRL), or CRL Distribution Point (CRLDP) for revocation checking of the certificate.
- Active Directory, LDAP, or another directory service to query the identity of the authenticated user.

SSL protocol flow with APM ODCA, ignore in clientssl profile



Certificate Key Chain	Add Replace /Common/www.gov.gov.crt /Common/www.gov.gov.key

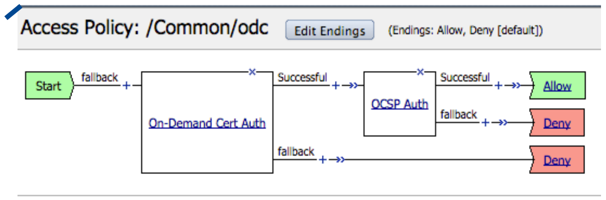
Client Side

Server Side

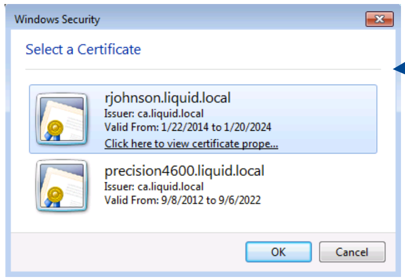
1. Client initiates SSL session

2. LTM presents server cert from clientssl profile

3. SSL session is established, client begins to be evaluated by APM VPE



Access Policy: /Common/odc Edit Endings (Endings: Allow, Deny [default])



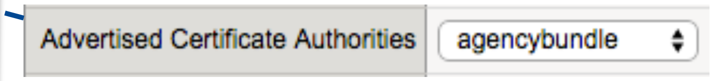
Windows Security

Select a Certificate

rjohnson.liquid.local
Issuer: ca.liquid.local
Valid From: 1/22/2014 to 1/20/2024
Click here to view certificate prope...

precision4600.liquid.local
Issuer: ca.liquid.local
Valid From: 9/8/2012 to 9/6/2022

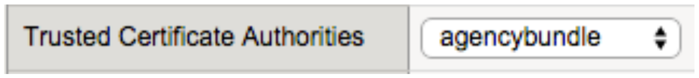
4. Client browser shows pop-up with available certs from user cert store, because ODCA causes an SSL renegotiation which requests a client cert. Available certs are controlled by the Advertised Certificate Authorities in the clientssl profile



Advertised Certificate Authorities	agencybundle

5. Client chooses their cert, the software that talks to the smart card will prompt for PIN to decrypt the user's private key on the card

6. The F5 will verify the client cert against the certificate bundle in the Trusted Certificate Authorities



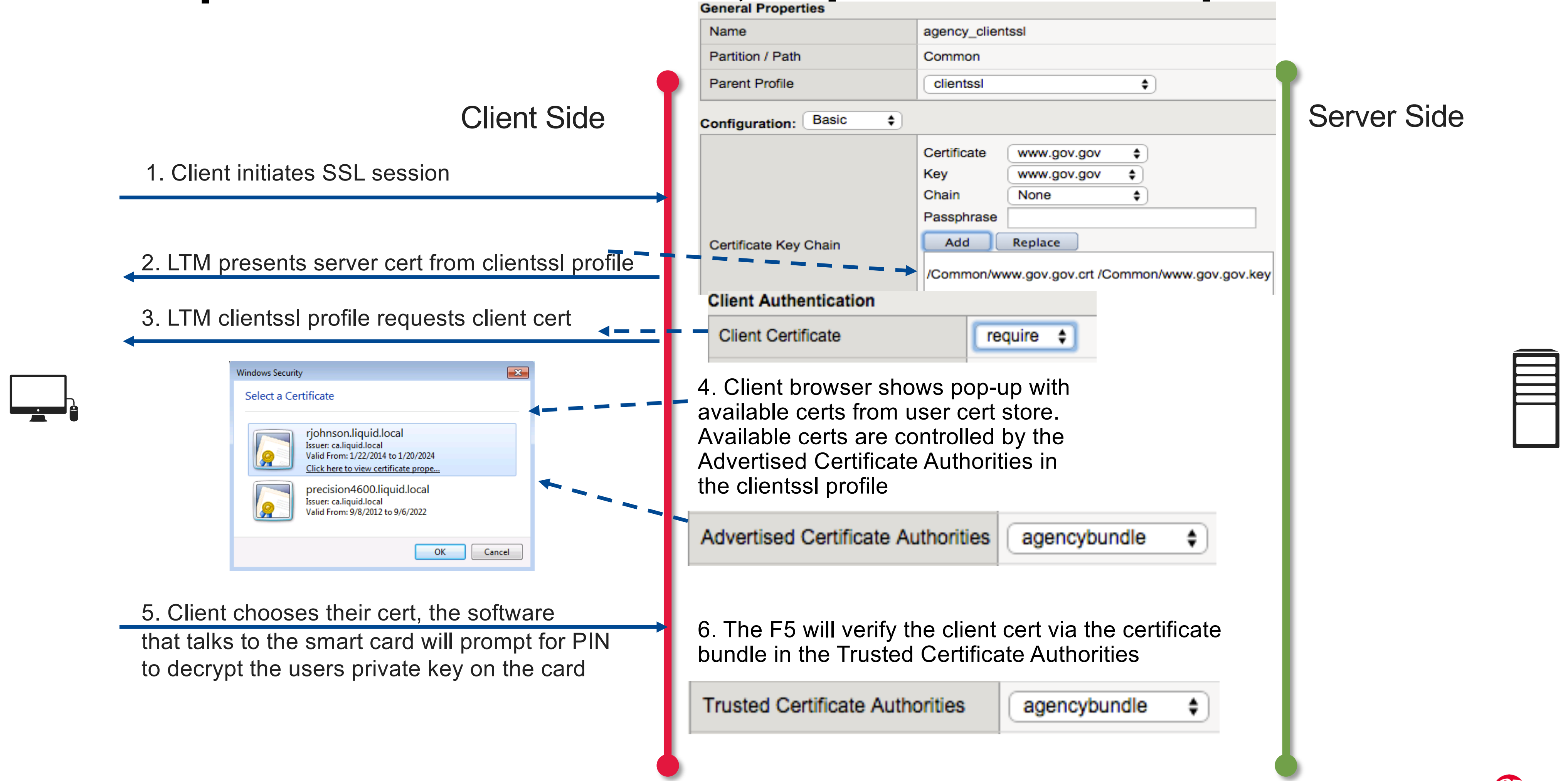
Trusted Certificate Authorities	agencybundle

7. Client is allowed access to the app

BIG-IP APM On Demand Certificate Authentication (ODCA)

- ODCA has been the preferred method of requesting the client certificate from a CAC/PIV for many years. It’s flexible.
- Users can browse the website until they try to access restricted resources and perform “step up” authentication which would request or require the user certificate.
- ODCA allows fallback authentication options like username/password.
- Vulnerable to session hijacking if a Man in the Browser (MITB) is present.
  - MITB steals the cookie and sends it to the attacker
  - Stolen cookies could be used to resume sessions

SSL protocol flow with APM, require in clientssl profile



1. Client initiates SSL session

2. LTM presents server cert from clientssl profile

3. LTM clientssl profile requests client cert

4. Client browser shows pop-up with available certs from user cert store. Available certs are controlled by the Advertised Certificate Authorities in the clientssl profile

5. Client chooses their cert, the software that talks to the smart card will prompt for PIN to decrypt the users private key on the card

6. The F5 will verify the client cert via the certificate bundle in the Trusted Certificate Authorities

General Properties

Name	agency_clientssl
Partition / Path	Common
Parent Profile	clientssl

Configuration: Basic

Certificate Key Chain	Certificate: www.gov.gov; Key: www.gov.gov; Chain: None; Passphrase:
	/Common/www.gov.gov.crt /Common/www.gov.gov.key

Client Authentication

Client Certificate	require
Advertised Certificate Authorities	agencybundle
Trusted Certificate Authorities	agencybundle

Windows Security

Select a Certificate

rjohnson.liquid.local
Issuer: ca.liquid.local
Valid From: 1/22/2014 to 1/20/2024
Click here to view certificate prope...

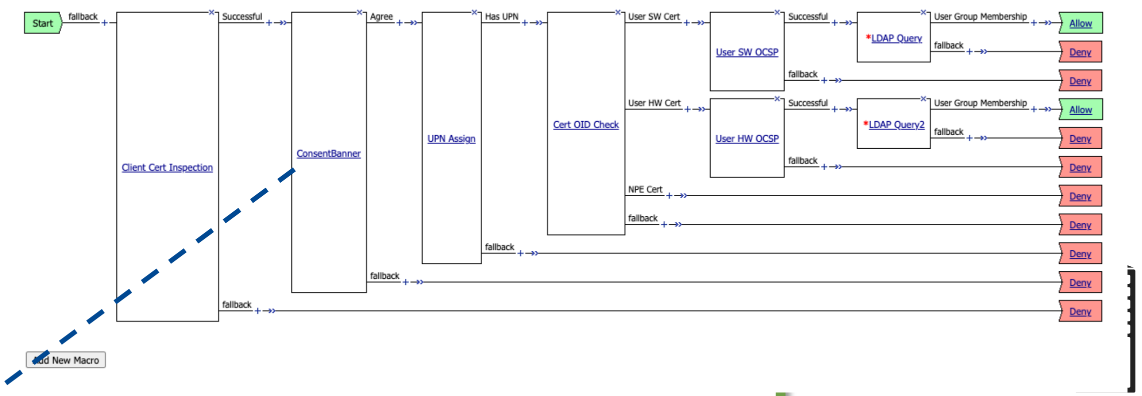
precision4600.liquid.local
Issuer: ca.liquid.local
Valid From: 9/8/2012 to 9/6/2022

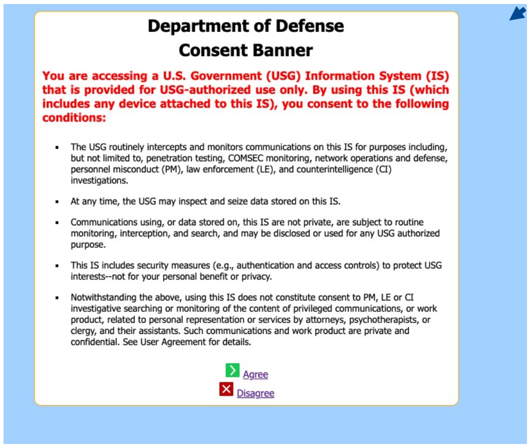
Continued…

Client Side

7. Client begins to be evaluated by APM VPE
8. APM prompts user with a consent banner
9. UPN/EDIPI is verified from certificate.
10. Cert OID is verified from certificate.
11. OCSP request to check certificate revocation status.
12. LDAP/AD Query to verify user exists and account is enabled.

Server Side




Department of Defense
Consent Banner

You are accessing a U.S. Government (USG) Information System (IS) that is provided for USG-authorized use only. By using this IS (which includes any device attached to this IS), you consent to the following conditions:

- The USG routinely intercepts and monitors communications on this IS for purposes including, but not limited to, penetration testing, COMSEC monitoring, network operations and defense, personnel misconduct (PM), law enforcement (LE), and counterintelligence (CI) investigations.
- At any time, the USG may inspect and seize data stored on this IS.
- Communications using, or data stored on, this IS are not private, are subject to routine monitoring, interception, and search, and may be disclosed or used for any USG authorized purpose.
- This IS includes security measures (e.g., authentication and access controls) to protect USG interests--not for your personal benefit or privacy.
- Notwithstanding the above, using this IS does not constitute consent to PM, LE or CI investigative searching or monitoring of the content of privileged communications, or work product, related to personal representation or services by attorneys, psychotherapists, or clergy, and their assistants. Such communications and work product are private and confidential. See User Agreement for details.

13. Client is allowed access to the app

BIG-IP APM with ClientSSL set to required

- This is the most secure option. It’s rigid.

- This method will also work with clients/agents that are unable to handle redirects from APM. This is known as clientless mode.

- If the user does not present a certificate the browser will fail to an SSL error page – no comfort pages.

Configuration Walkthrough and Recommended Practices

Disclaimer

- The settings and recommended practices discussed have not been tested with every possible configuration and may negatively impact your environments.
- Please test before making these changes in production.
- Understand the settings and how they may affect your environment.

K000138221: Mitigate potential attacks using features included with BIG-IP APM

- Maximum Session Timeout
- Max Sessions Per User
- Max In Progress Sessions Per Client IP
- Restrict to Single Client IP
- HTTP Only Cookies
- ”Persistent” Cookies
- Samesite Cookies
- Revocation Checks
- …and more

K000138221: Mitigate potential attacks using features included with BIG-IP APM

- The **Maximum Session Timeout** setting is an important attribute for a BIG-IP APM access profile because it defines the maximum length of time a session can be active before it is automatically terminated. By limiting the duration of each session, you can mitigate the risk of session hijacking where an attacker could steal or use the session cookie to gain unauthorized access to confidential resources.
- The **Max Sessions Per User** setting can be used to limit the number of times an *individual user* can create sessions into your application. It may not be unusual for a user to create multiple sessions into your application but this can be limited to reduce the possibility of session hijacking.

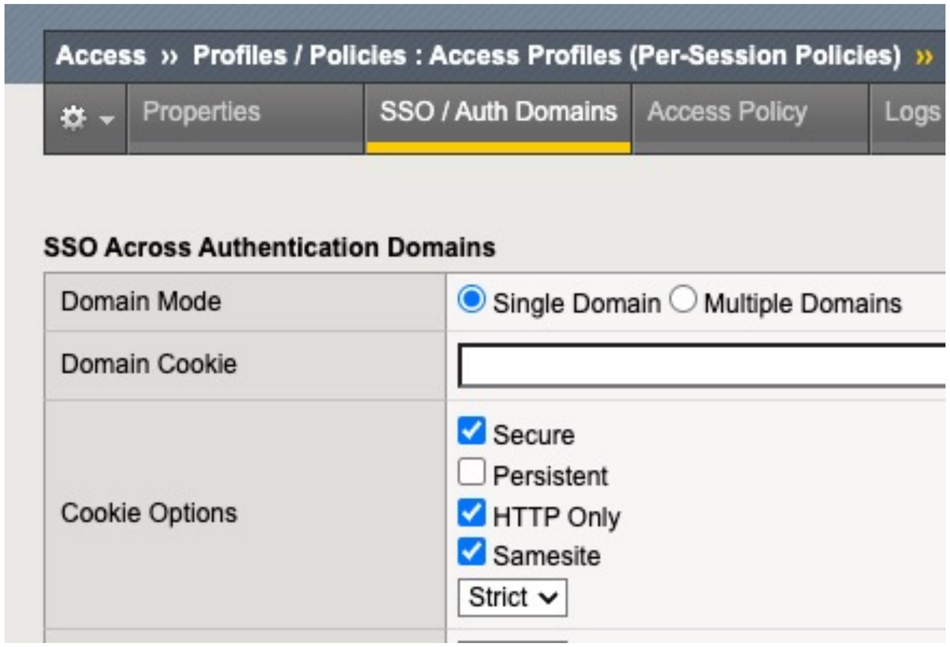
K000138221: Mitigate potential attacks using features included with BIG-IP APM

- The **Max In Progress Sessions Per Client IP** setting in a BIG-IP APM access profile is a security configuration that limits the number of simultaneous sessions that a client can initiate from *a single IP address*. This setting can be helpful to prevent either accidental or intentional session flooding on the BIG-IP, however if your clients are behind a proxy this setting may cause issues.
- The **Restrict to Single Client IP** setting is an essential security measure within a BIG-IP APM access profile. When enabled, this setting ensures that a session can be accessed only from the *same IP address* from which it was initially created. This is a potent safeguard against attacks such as session hijacking or cookie theft, as even if an attacker manages to steal a session cookie, they cannot use it from a different IP address. This setting effectively ties the user session to a specific IP address, further enhancing the security of the BIG-IP APM access profiles.

K000138221: Mitigate potential attacks using features included with BIG-IP APM – Cookie options

- F5 recommends enabling the **HTTP Only** option. This measure is designed to mitigate the risk of client-side scripts gaining access to the BIG-IP APM session cookies, thus enhancing the security of your sessions.
- The **Persistent** cookie option in APM can present a security risk and is disabled by default. This option is primarily used when the session needs to be resumed by another application, such as Office Suite for Sharepoint. The cookies are set to expire after 60 seconds. Persistent cookies can be accessed by other processes.
- **Samesite** cookie protection was added as an option beginning in BIG-IP APM 16.0. You can enable this setting to add the samesite attribute to the APM session cookie. This attribute enforces same-site usage and prevents the cookie from being included with cross-site requests.

K000138221: Mitigate potential attacks using features included with BIG-IP APM


Access » Profiles / Policies : Access Profiles (Per-Session Policies) »
Properties
SSO / Auth Domains
Access Policy
Logs

SSO Across Authentication Domains

Domain Mode	[✓ Single Domain] Multiple Domains
Domain Cookie	
Cookie Options	✓ Secure ☐ Persistent ✓ HTTP Only ✓ Samesite Strict

Strict: Only include the cookie with requests originating from the same site as the cookie.
Lax: Include the cookie with same-site requests and with top-level cross-site navigations that use a safe HTTP method.

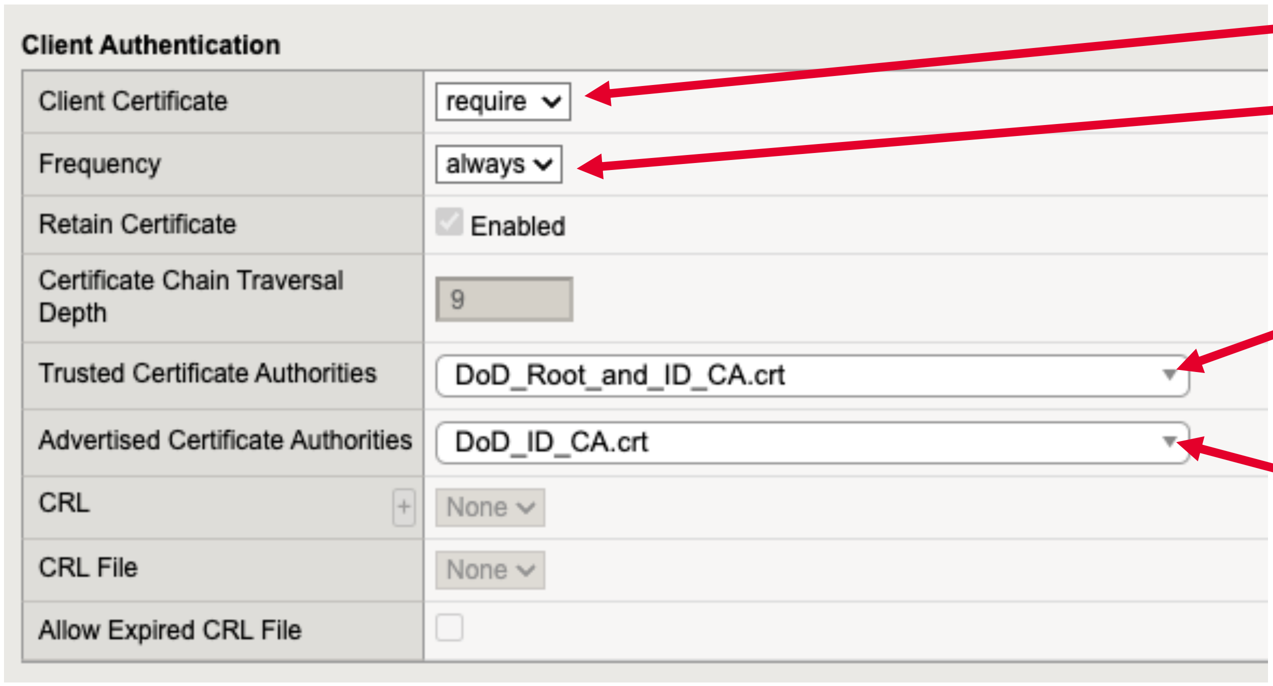
K000138221: Mitigate potential attacks using features included with BIG-IP APM

- Revocation checks
  - The best revocation check option is **Online Certificate Status Protocol (OCSP)**. OCSP can be configured within the BIG-IP APM to either use a configured responder or reference the responders within the Authority Information Access (AIA) extension of the certificate.
  - **Certificate Revocation Lists (CRL)** can be manually or automatically updated on the BIG-IP to verify revocation status of client certificates. The max file size for a CRL is now 192MB (15.x+).
  - **Certificate Revocation List Distribution Points (CRLDP)** is the final option for revocation status checking. This is the least desirable option due to the time it takes to pull a large CRL file. The CRLDP, like OCSP, can be statically defined or pulled from the AIA extension.

…and more

- **UPN Checks** – verify the cert contains a properly formatted User Principal Name (UPN) in the cert extensions
- **Cert OID Checks** – verify appropriate user OIDs are present in the cert extensions **-** deny Non-Person Entity (NPE) certs
- **ClientSSL profile frequency always** – enforce mTLS continuously
- **Limit the scope of advertised CAs and trusted CAs** – bundle manager
- **Serial Number Check** – verify serial number of cert matches initial APM session variables

ClientSSL and Bundles



Client Authentication

Client Certificate	require
Frequency	always
Retain Certificate	✓ Enabled
Certificate Chain Traversal Depth	9
Trusted Certificate Authorities	DoD_Root_and_ID_CA.crt
Advertised Certificate Authorities	DoD_ID_CA.crt
CRL	None
CRL File	None
Allow Expired CRL File	☐

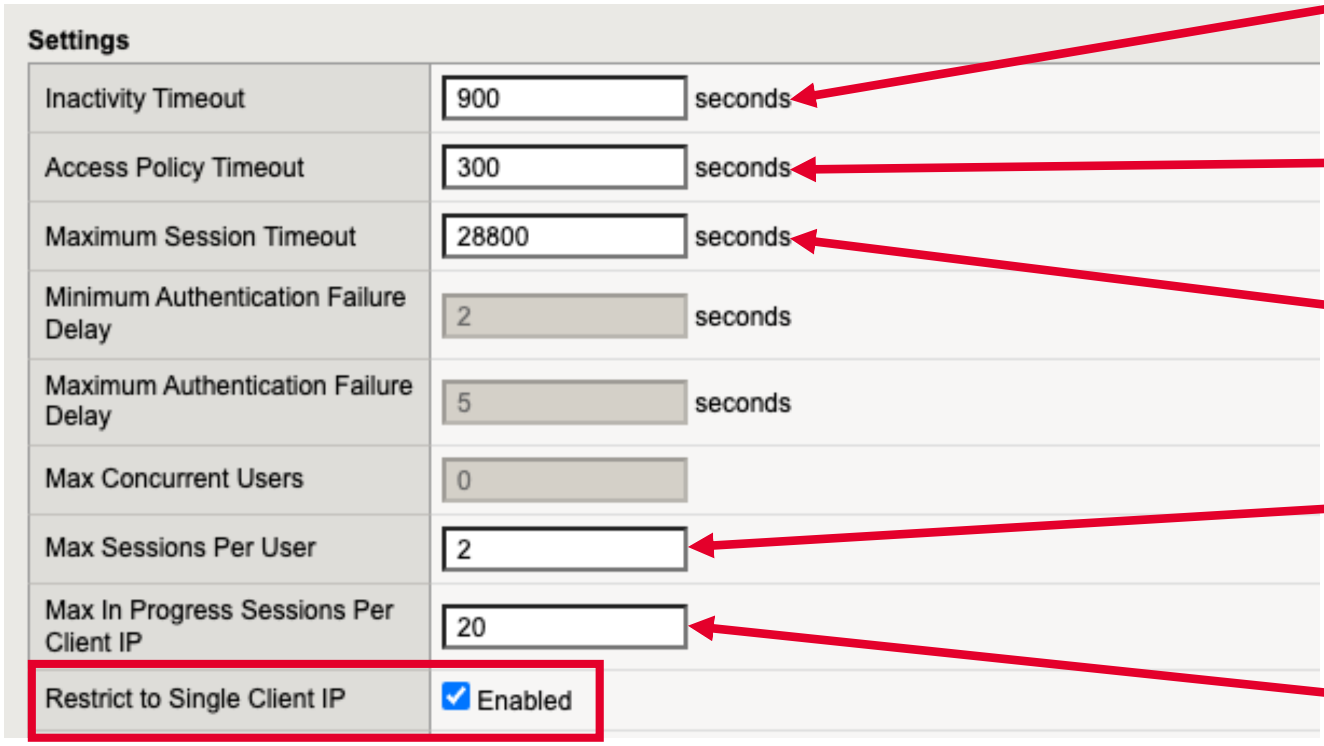
Client Certificate is required and will fail negotiation if no certificate is present.

Always will force the browser to continuously send the certificate on session resumption.

This bundle contains the full trust chain to validate the client certificate.

This bundle contains only the certificate authorities that sign the client certificate. This limits the advertised scope in the SSL negotiation within the browser.

Policy Properties



Settings

Inactivity Timeout	900 seconds
Access Policy Timeout	300 seconds
Maximum Session Timeout	28800 seconds
Minimum Authentication Failure Delay	2 seconds
Maximum Authentication Failure Delay	5 seconds
Max Concurrent Users	0
Max Sessions Per User	2
Max In Progress Sessions Per Client IP	20
Restrict to Single Client IP	✓ Enabled

Default 15 minutes

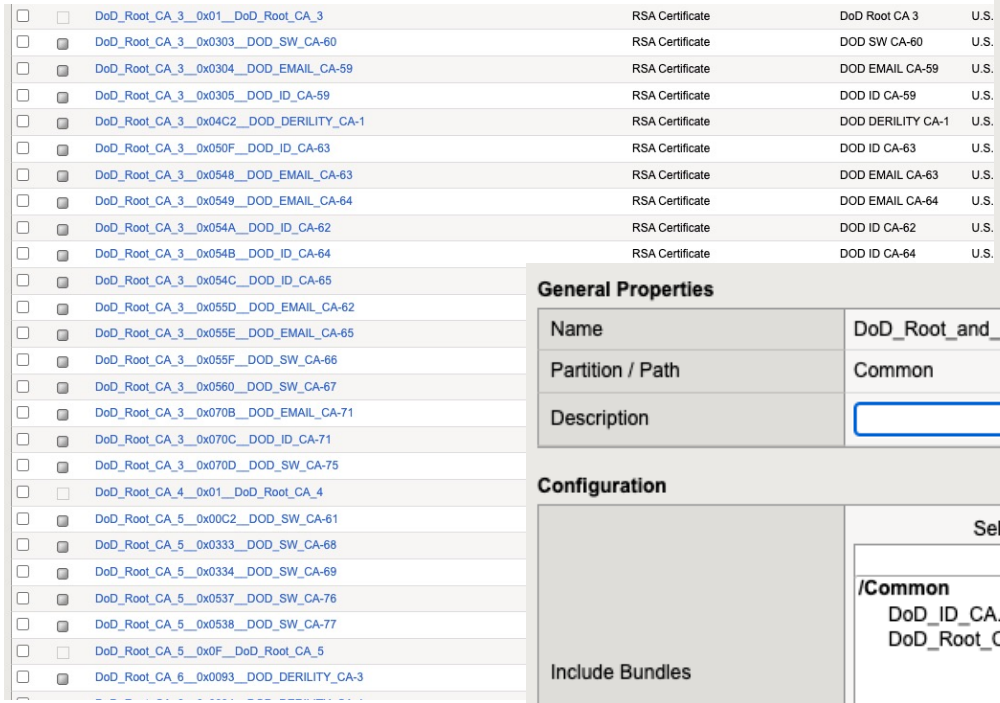
Default 5 minutes

Default 7 days!

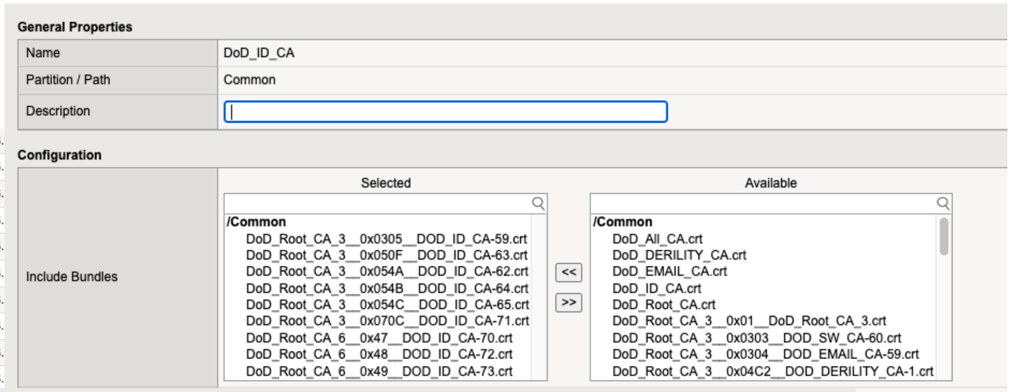
Default unlimited!

Default 128!

Certificate Bundles



DoD_Root_CA_3__0x01__DoD_Root_CA_3	RSA Certificate	DoD Root CA 3	U.S.
DoD_Root_CA_3__0x0303__DOD_SW_CA-60	RSA Certificate	DOD SW CA-60	U.S.
DoD_Root_CA_3__0x0304__DOD_EMAIL_CA-59	RSA Certificate	DOD EMAIL CA-59	U.S.
DoD_Root_CA_3__0x0305__DOD_ID_CA-59	RSA Certificate	DOD ID CA-59	U.S.
DoD_Root_CA_3__0x04C2__DOD_DERILITY_CA-1	RSA Certificate	DOD DERILITY CA-1	U.S.
DoD_Root_CA_3__0x050F__DOD_ID_CA-63	RSA Certificate	DOD ID CA-63	U.S.
DoD_Root_CA_3__0x0548__DOD_EMAIL_CA-63	RSA Certificate	DOD EMAIL CA-63	U.S.
DoD_Root_CA_3__0x0549__DOD_EMAIL_CA-64	RSA Certificate	DOD EMAIL CA-64	U.S.
DoD_Root_CA_3__0x054A__DOD_ID_CA-62	RSA Certificate	DOD ID CA-62	U.S.
DoD_Root_CA_3__0x054B__DOD_ID_CA-64	RSA Certificate	DOD ID CA-64	U.S.
DoD_Root_CA_3__0x054C__DOD_ID_CA-65			
DoD_Root_CA_3__0x055D__DOD_EMAIL_CA-62			
DoD_Root_CA_3__0x055E__DOD_EMAIL_CA-65			
DoD_Root_CA_3__0x055F__DOD_SW_CA-66			
DoD_Root_CA_3__0x0560__DOD_SW_CA-67			
DoD_Root_CA_3__0x070B__DOD_EMAIL_CA-71			
DoD_Root_CA_3__0x070C__DOD_ID_CA-71			
DoD_Root_CA_3__0x070D__DOD_SW_CA-75			
DoD_Root_CA_4__0x01__DoD_Root_CA_4			
DoD_Root_CA_5__0x00C2__DOD_SW_CA-61			
DoD_Root_CA_5__0x0333__DOD_SW_CA-68			
DoD_Root_CA_5__0x0334__DOD_SW_CA-69			
DoD_Root_CA_5__0x0537__DOD_SW_CA-76			
DoD_Root_CA_5__0x0538__DOD_SW_CA-77			
DoD_Root_CA_5__0x0F__DoD_Root_CA_5			
DoD_Root_CA_6__0x0093__DOD_DERILITY_CA-3			



General Properties

Name	DoD_ID_CA
Partition / Path	Common
Description	

Configuration

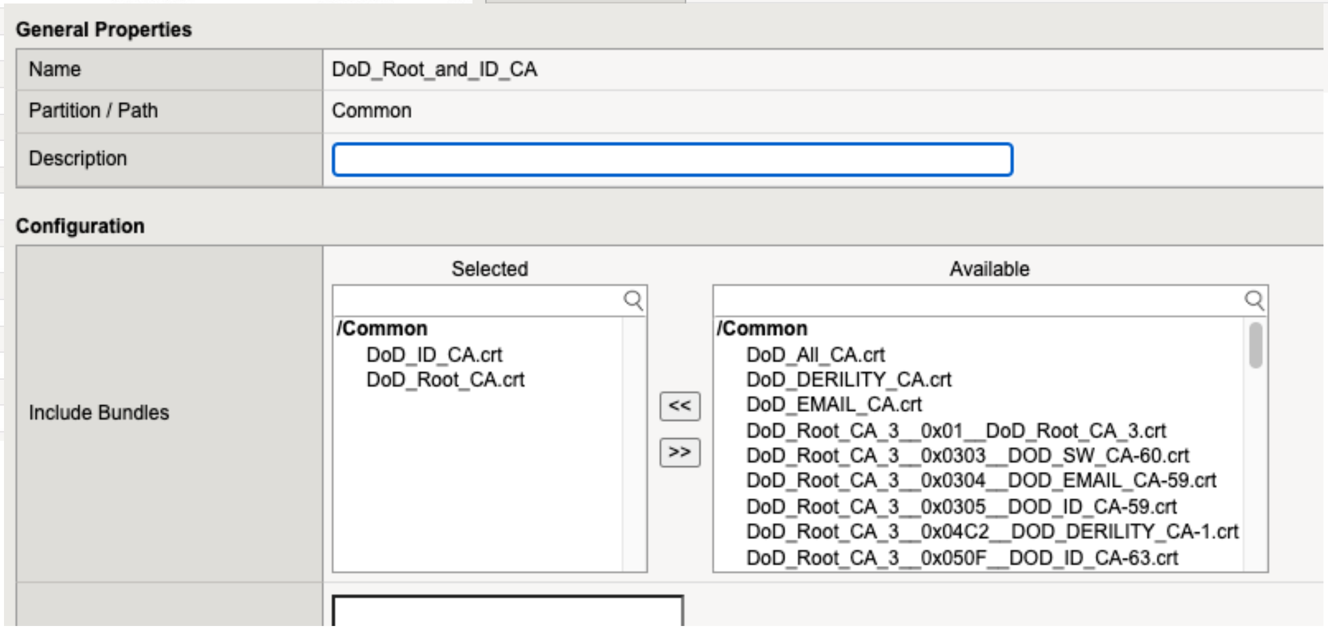
Include Bundles

Selected:

/Common
- DoD_Root_CA_3__0x0305__DOD_ID_CA-59.crt
- DoD_Root_CA_3__0x050F__DOD_ID_CA-63.crt
- DoD_Root_CA_3__0x054A__DOD_ID_CA-62.crt
- DoD_Root_CA_3__0x054B__DOD_ID_CA-64.crt
- DoD_Root_CA_3__0x054C__DOD_ID_CA-65.crt
- DoD_Root_CA_3__0x070C__DOD_ID_CA-71.crt
- DoD_Root_CA_6__0x47__DOD_ID_CA-70.crt
- DoD_Root_CA_6__0x48__DOD_ID_CA-72.crt
- DoD_Root_CA_6__0x49__DOD_ID_CA-73.crt

Available:

/Common
- DoD_All_CA.crt
- DoD_DERILITY_CA.crt
- DoD_EMAIL_CA.crt
- DoD_ID_CA.crt
- DoD_Root_CA.crt
- DoD_Root_CA_3__0x01__DoD_Root_CA_3.crt
- DoD_Root_CA_3__0x0303__DOD_SW_CA-60.crt
- DoD_Root_CA_3__0x0304__DOD_EMAIL_CA-59.crt
- DoD_Root_CA_3__0x04C2__DOD_DERILITY_CA-1.crt



General Properties

Name	DoD_Root_and_ID_CA
Partition / Path	Common
Description	

Configuration

Include Bundles

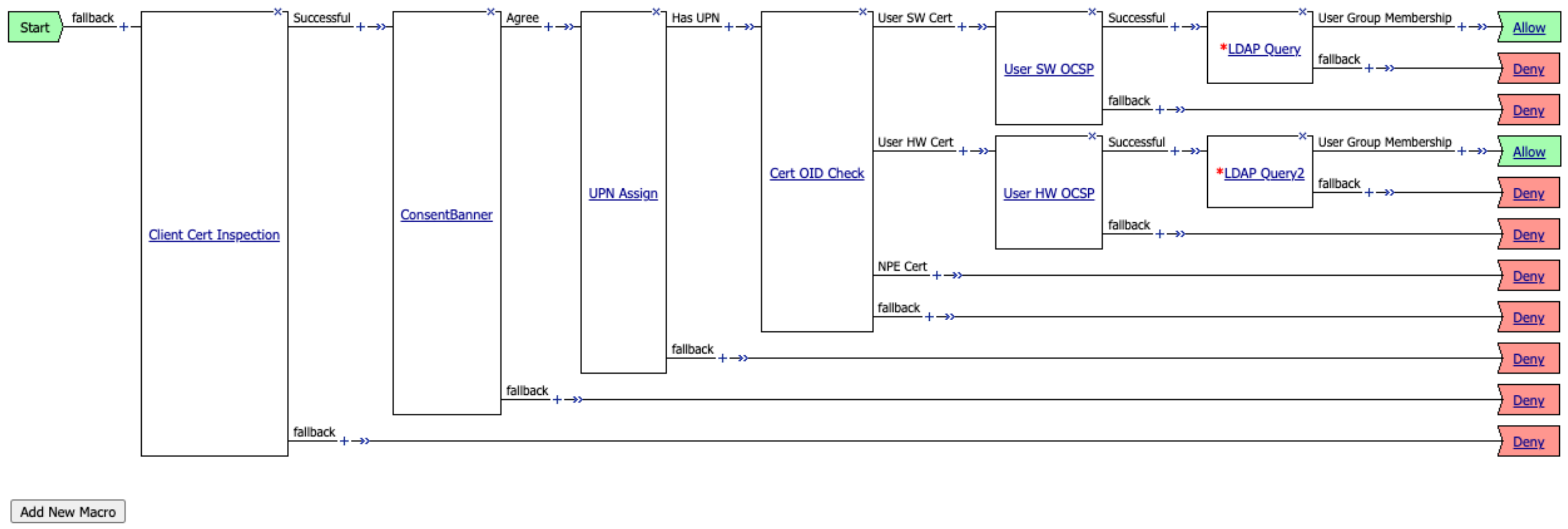
Selected:

/Common
- DoD_ID_CA.crt
- DoD_Root_CA.crt

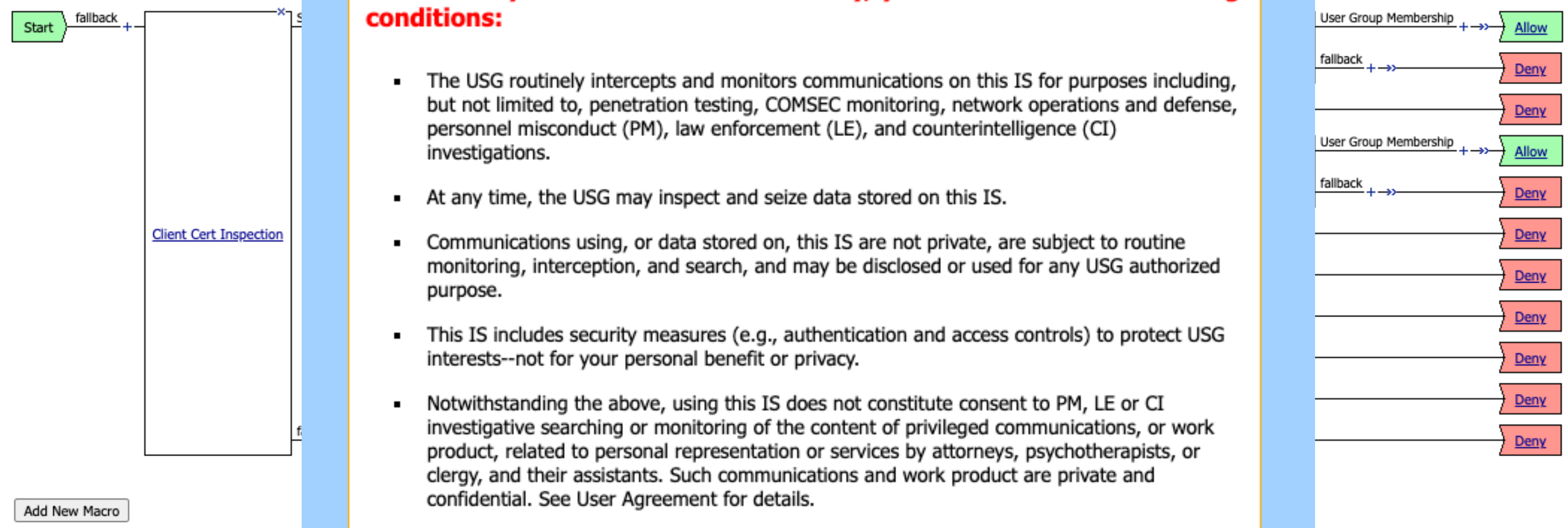
Available:

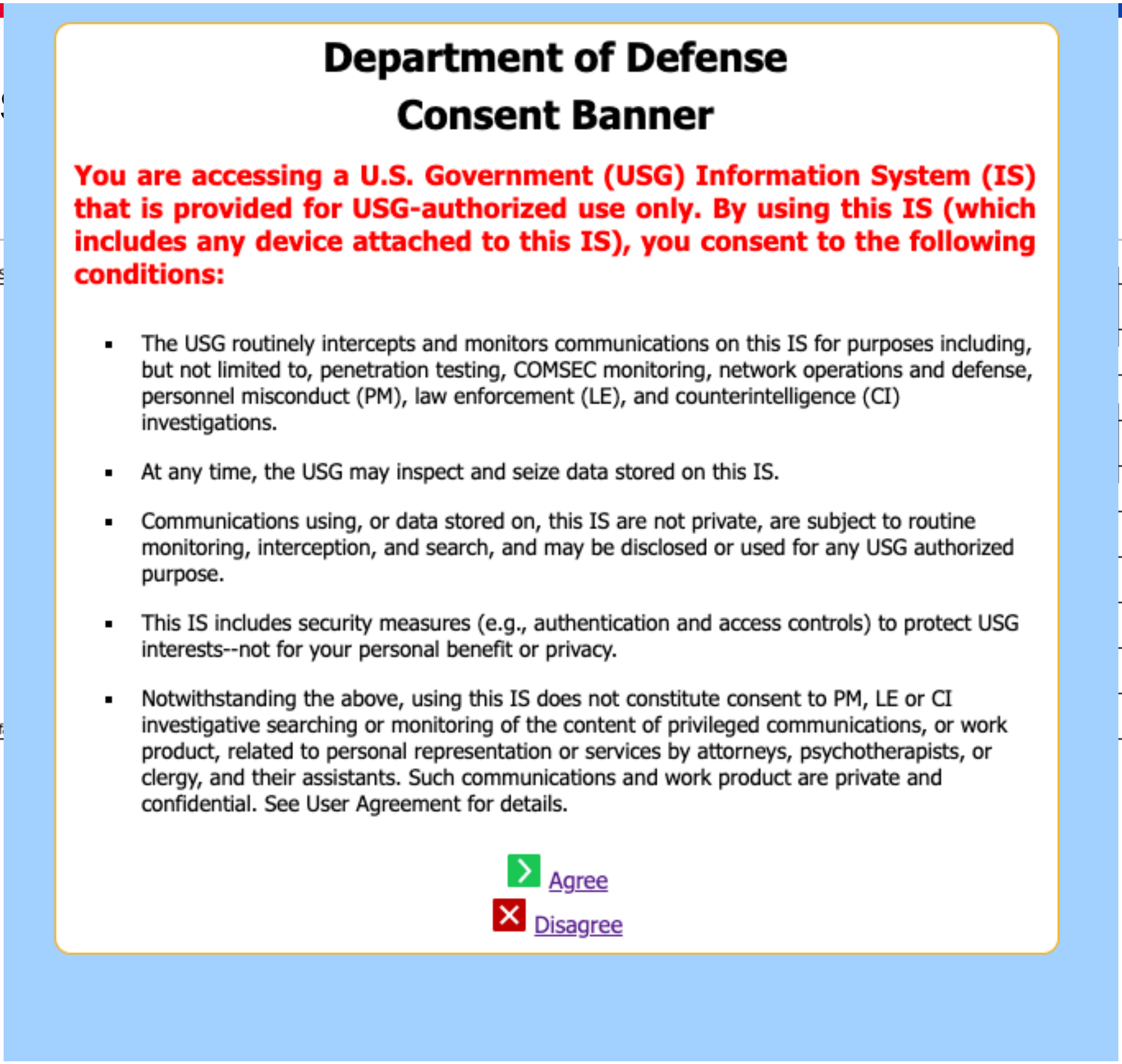
/Common
- DoD_All_CA.crt
- DoD_DERILITY_CA.crt
- DoD_EMAIL_CA.crt
- DoD_Root_CA_3__0x01__DoD_Root_CA_3.crt
- DoD_Root_CA_3__0x0303__DOD_SW_CA-60.crt
- DoD_Root_CA_3__0x0304__DOD_EMAIL_CA-59.crt
- DoD_Root_CA_3__0x0305__DOD_ID_CA-59.crt
- DoD_Root_CA_3__0x04C2__DOD_DERILITY_CA-1.crt
- DoD_Root_CA_3__0x050F__DOD_ID_CA-63.crt

Example Visual Policy Editor (VPE)



Example Vi





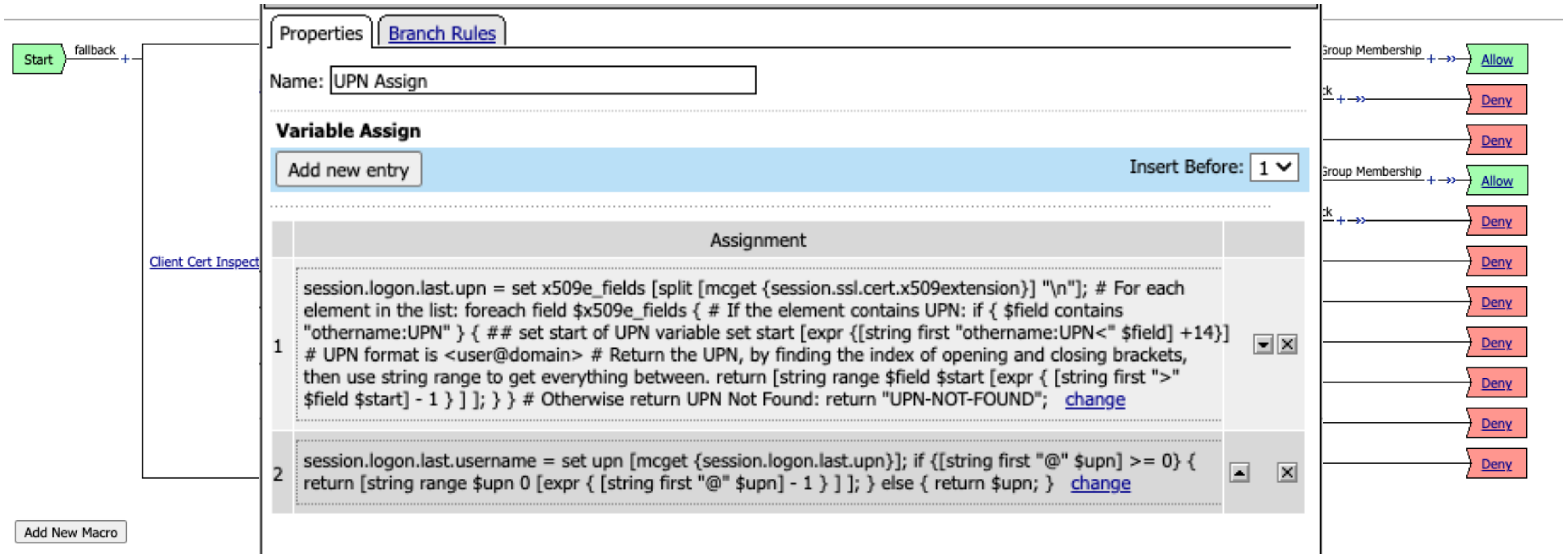
Department of Defense
Consent Banner

You are accessing a U.S. Government (USG) Information System (IS) that is provided for USG-authorized use only. By using this IS (which includes any device attached to this IS), you consent to the following conditions:

- The USG routinely intercepts and monitors communications on this IS for purposes including, but not limited to, penetration testing, COMSEC monitoring, network operations and defense, personnel misconduct (PM), law enforcement (LE), and counterintelligence (CI) investigations.
- At any time, the USG may inspect and seize data stored on this IS.
- Communications using, or data stored on, this IS are not private, are subject to routine monitoring, interception, and search, and may be disclosed or used for any USG authorized purpose.
- This IS includes security measures (e.g., authentication and access controls) to protect USG interests--not for your personal benefit or privacy.
- Notwithstanding the above, using this IS does not constitute consent to PM, LE or CI investigative searching or monitoring of the content of privileged communications, or work product, related to personal representation or services by attorneys, psychotherapists, or clergy, and their assistants. Such communications and work product are private and confidential. See User Agreement for details.

Agree
Disagree

Example Visual Policy Editor (VPE)

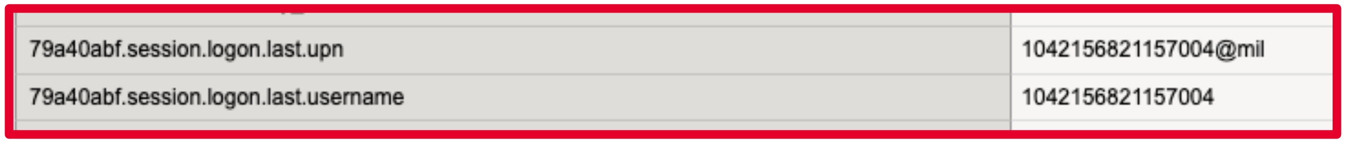


Name: UPN Assign

Variable Assign

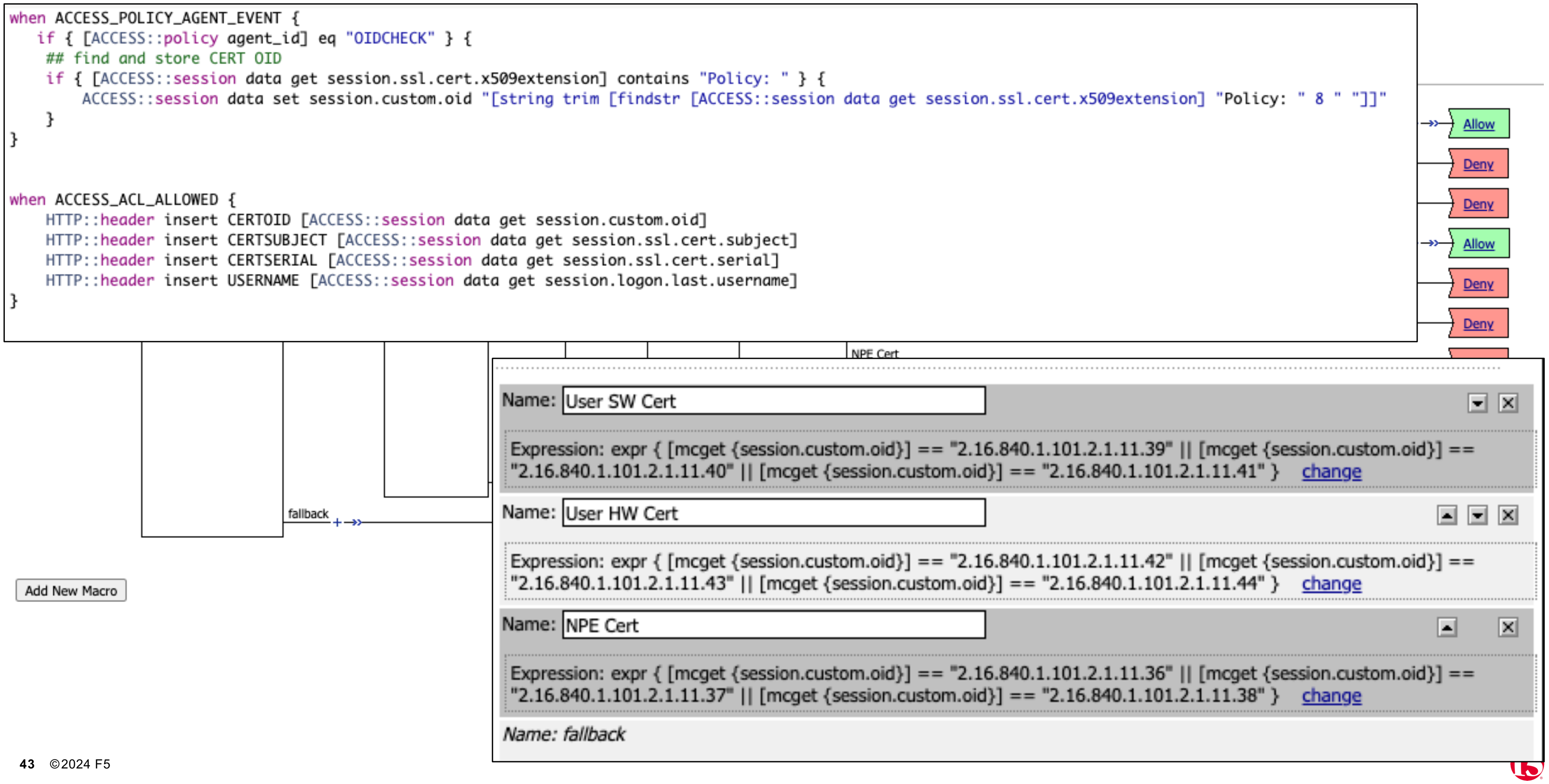
Add new entry   Insert Before: 1

	Assignment
1	session.logon.last.upn = set x509e_fields [split [mcget {session.ssl.cert.x509extension}] "\n"]; # For each element in the list: foreach field $x509e_fields { # If the element contains UPN: if { $field contains "othername:UPN" } { ## set start of UPN variable set start [expr {[string first "othername:UPN<" $field] +14}] # UPN format is <user@domain> # Return the UPN, by finding the index of opening and closing brackets, then use string range to get everything between. return [string range $field $start [expr { [string first ">" $field $start] - 1 }]]; } } # Otherwise return UPN Not Found: return "UPN-NOT-FOUND"; change
2	session.logon.last.username = set upn [mcget {session.logon.last.upn}]; if {[string first "@" $upn] >= 0} { return [string range $upn 0 [expr { [string first "@" $upn] - 1 }]]; } else { return $upn; } change



79a40abf.session.logon.last.upn	1042156821157004@mil
79a40abf.session.logon.last.username	1042156821157004

Example Visual Policy Editor (VPE)



```
when ACCESS_POLICY_AGENT_EVENT {
   if { [ACCESS::policy agent_id] eq "OIDCHECK" } {
    ## find and store CERT OID
    if { [ACCESS::session data get session.ssl.cert.x509extension] contains "Policy: " } {
        ACCESS::session data set session.custom.oid "[string trim [findstr [ACCESS::session data get session.ssl.cert.x509extension] "Policy: " 8 " "]]"
    }
}


when ACCESS_ACL_ALLOWED {
    HTTP::header insert CERTOID [ACCESS::session data get session.custom.oid]
    HTTP::header insert CERTSUBJECT [ACCESS::session data get session.ssl.cert.subject]
    HTTP::header insert CERTSERIAL [ACCESS::session data get session.ssl.cert.serial]
    HTTP::header insert USERNAME [ACCESS::session data get session.logon.last.username]
}
```

Name: User SW Cert

Expression: expr { [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.39" || [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.40" || [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.41" } change

Name: User HW Cert

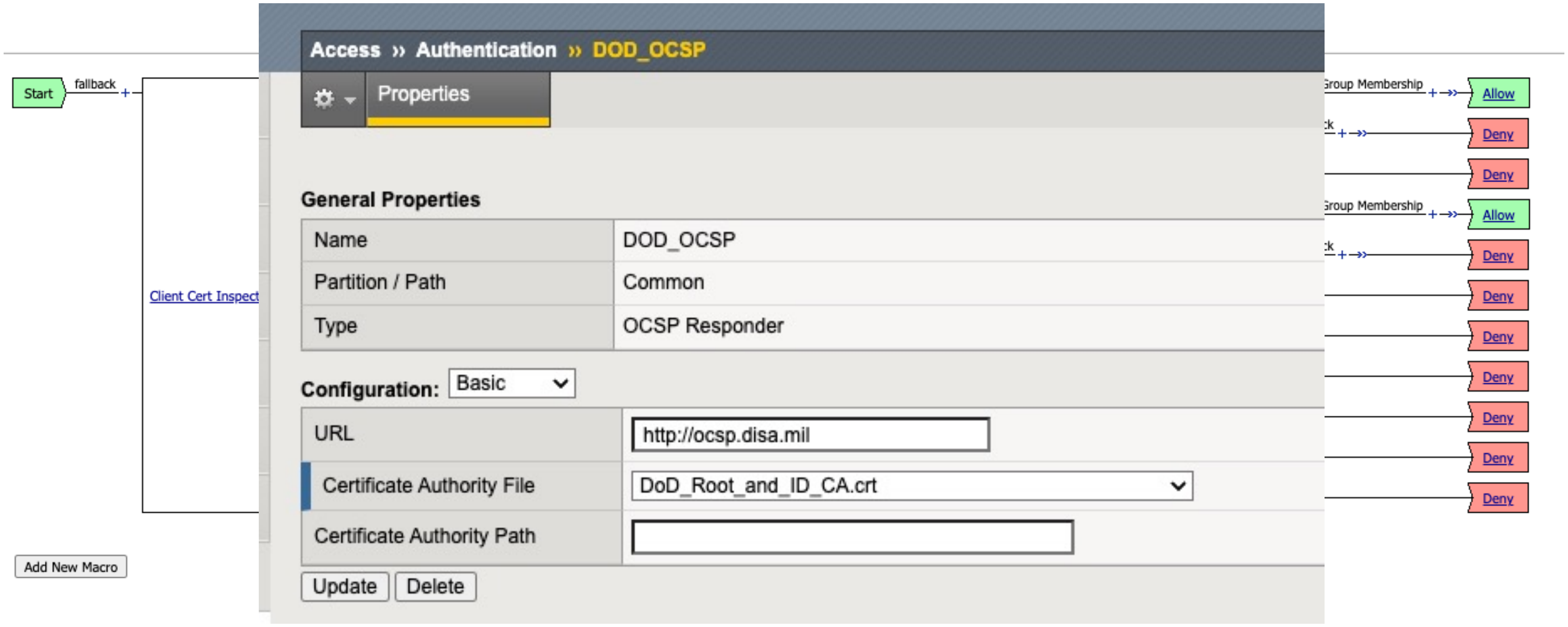
Expression: expr { [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.42" || [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.43" || [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.44" } change

Name: NPE Cert

Expression: expr { [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.36" || [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.37" || [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.38" } change

Name: fallback

Example Visual Policy Editor (VPE)



Access » Authentication » DOD_OCSP

Properties

General Properties

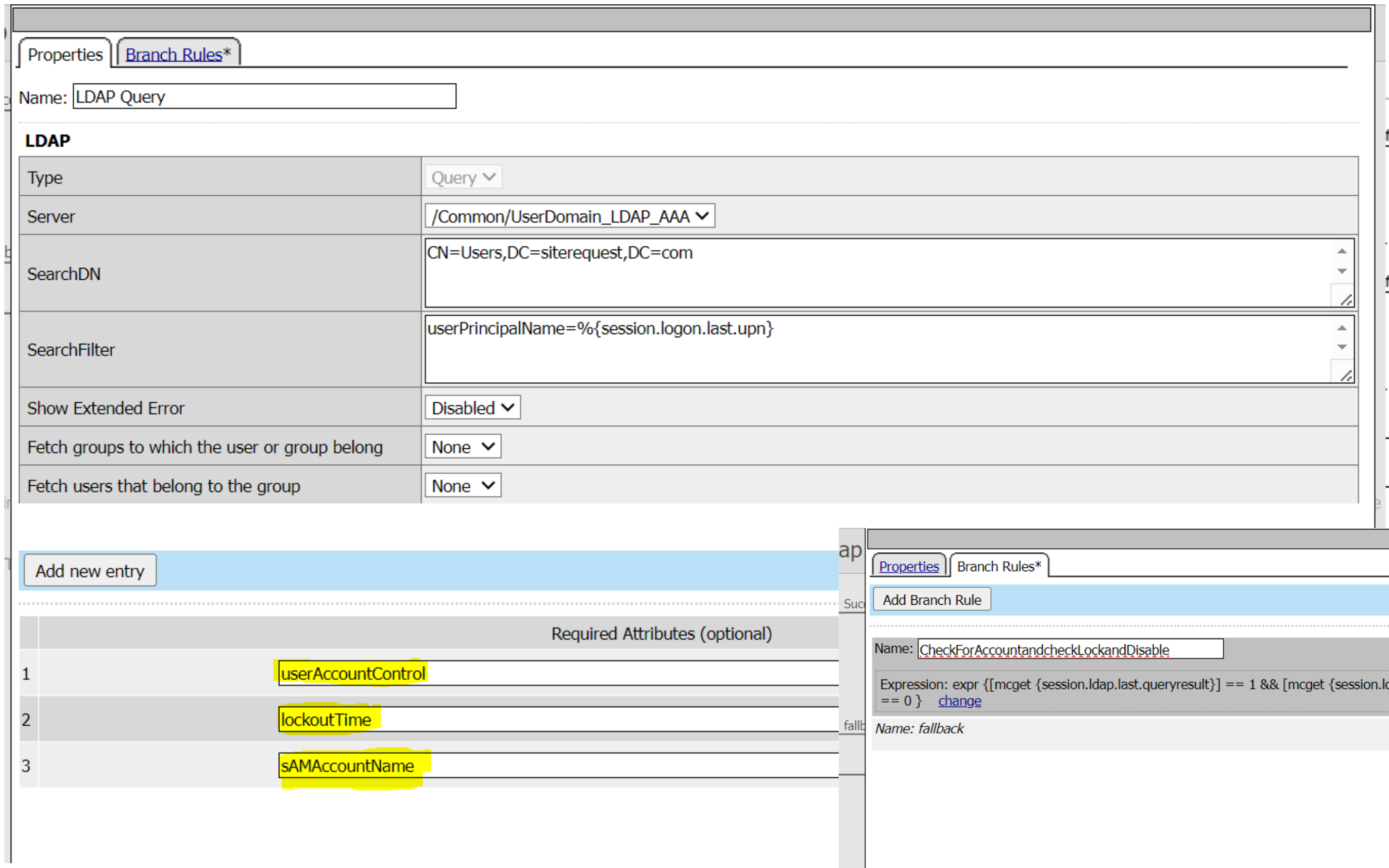
Name	DOD_OCSP
Partition / Path	Common
Type	OCSP Responder

Configuration: Basic

URL	http://ocsp.disa.mil
Certificate Authority File	DoD_Root_and_ID_CA.crt
Certificate Authority Path	

Update Delete

Example Visual Policy Editor (VPE)



Properties | Branch Rules*

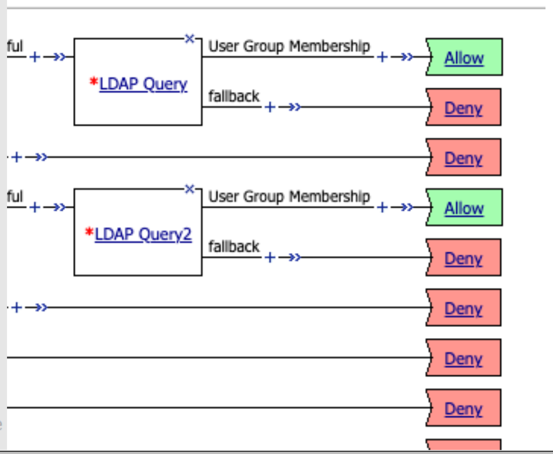
Name: LDAP Query

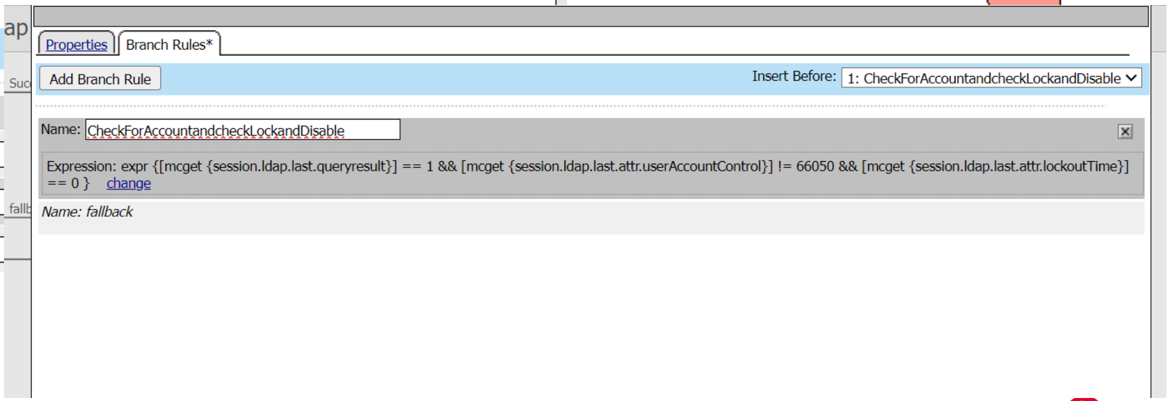
LDAP

Type	Query
Server	/Common/UserDomain_LDAP_AAA
SearchDN	CN=Users,DC=siterequest,DC=com
SearchFilter	userPrincipalName=%{session.logon.last.upn}
Show Extended Error	Disabled
Fetch groups to which the user or group belong	None
Fetch users that belong to the group	None

Add new entry

	Required Attributes (optional)
1	userAccountControl
2	lockoutTime
3	sAMAccountName





Properties | Branch Rules*

Add Branch Rule

Insert Before: 1: CheckForAccountandcheckLockandDisable

Name: CheckForAccountandcheckLockandDisable

```
Expression: expr {[mcget {session.ldap.last.queryresult}] == 1 && [mcget {session.ldap.last.attr.userAccountControl}] != 66050 && [mcget {session.ldap.last.attr.lockoutTime}] == 0 }
```
change

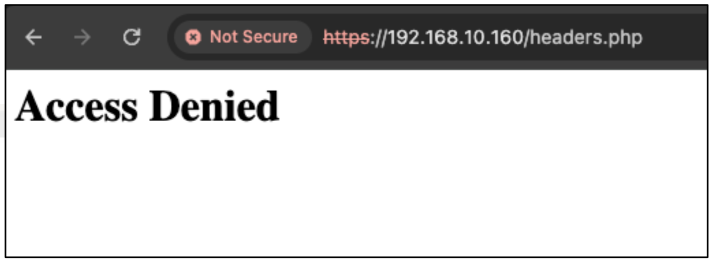
Name: fallback

More iRules!

```
# APM_X509_SN_Binding
# Copyright 2024 F5
# Binds an APM session to a X509 serial number from an mTLS connection
# invalidates SSL session and removes APM session on mismatch
# requires "client-certificate required" to be present in the clientssl profile
# compatible with TLS 1.2/1.3
when ACCESS_ACL_ALLOWED priority 100 {
    set tuple [IP::local_addr]:[TCP::local_port]->[IP::remote_addr]:[TCP::remote_port]
    # ensure client certificate is present
    if {[SSL::cert count] eq 0} {
        ACCESS::log accesscontrol.warn "APM_X509_SN_Binding - No Client Certificate present $tuple "
        ACCESS::respond 403 content {<html><h1>Access Denied</h1></html>}
        ACCESS::session remove
        SSL::session invalidate
        return
    }

    set sn [X509::serial_number [SSL::cert 0]]

    # does client-certificate serial match serial stored with APM session
    if {[ACCESS::session data get "session.ssl.cert.serial"] eq $sn} {
        ACCESS::log accesscontrol.info "APM_X509_SN_Binding - Client Certificate SN match: $sn"
        return
    }
    ACCESS::log accesscontrol.warn "APM_X509_SN_Binding - Attempted session hijack from $tuple with mismatched Client Certificate SN: $sn"
    ACCESS::respond 403 content {<html><h1>Access Denied</h1></html>}
    ACCESS::session remove
    SSL::session invalidate
}
```



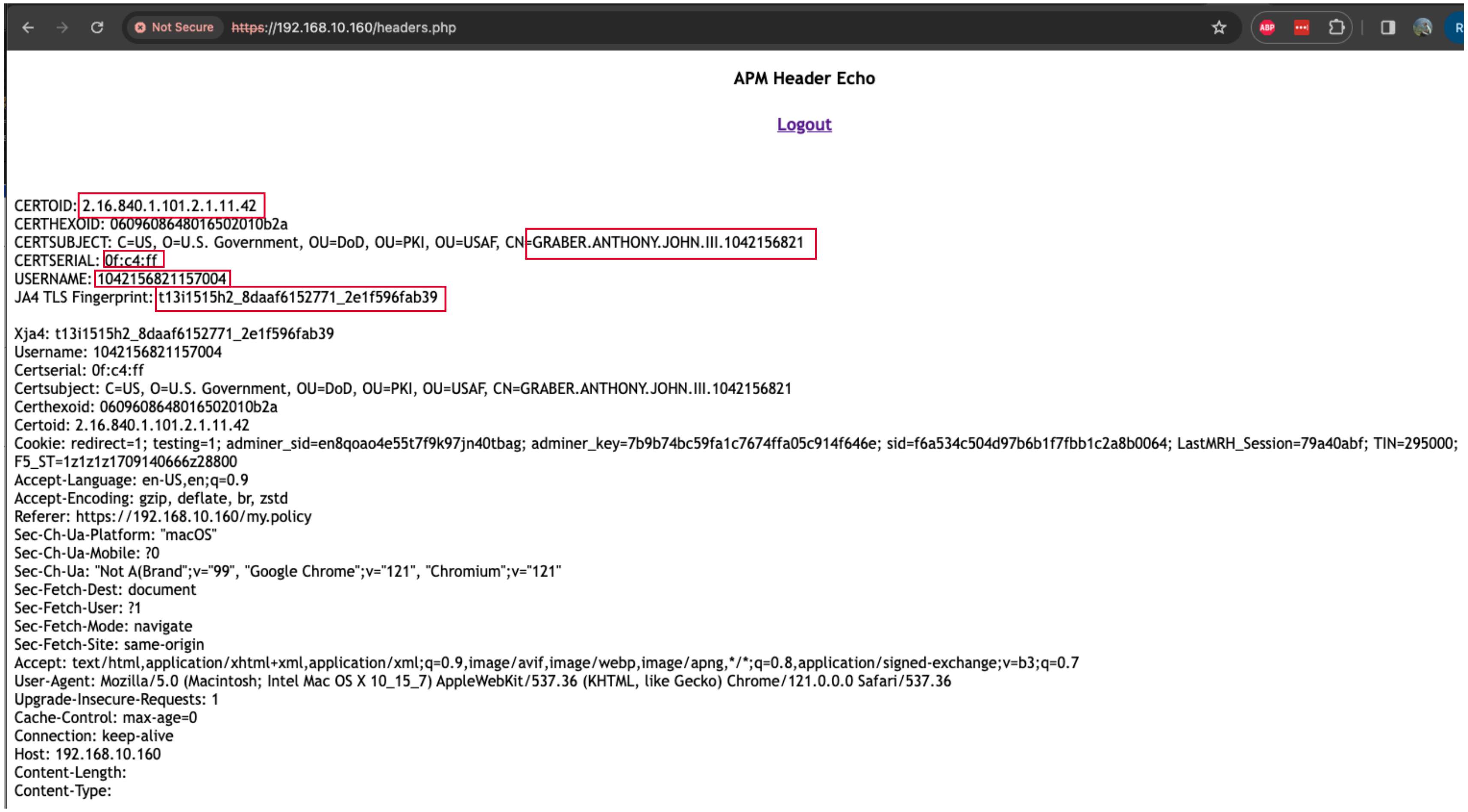
Access Denied

More iRules!

https://community.f5.com/kb/technicalarticles/fingerprinting-tls-clients-with-ja4-on-f5-big-ip/326298

```
proc getCipherList { payload rlen outer inner clientip serverip } {
    upvar cipher_cnt cipher_cnt
    ## Define GREASE values so these can be excluded from cipher list
    set greaseList "0a0a 1a1a 2a2a 3a3a 4a4a 5a5a 6a6a 7a7a 8a8a 9a9a aaaa baba caca dada eaea fafa"
    ## Skip over first 43 bytes (contains tls_type hello_len tls_ver, which we don't need)
    set field_offset 43
    ## Grab the session ID length value and increment field_offset.
    binary scan ${payload} @${field_offset}c sessID_len
    set field_offset [expr {${field_offset} + 1 + ${sessID_len}}]
    ## Grab ciphersuite list length (binary and hex values).
    binary scan ${payload} @${field_offset}S cipherList_len
    binary scan ${payload} @${field_offset}H4 cipherList_len_hex
    set cipherList_len_hex_text ${cipherList_len_hex}
    ## increment field_offset and get the ciphersuite list.
    set field_offset [expr {${field_offset} + 2}]
    set cipherList_len_hex [expr {${cipherList_len} * 2}]
    binary scan ${payload} @${field_offset}H${cipherList_len_hex} cipherlist
    ## Parse through cipherlist, add each non-GREASE cipher to cipherSuite list.
    set cipher_cnt 0
    set parsed_cl $cipherlist
    set cipherSuite {}
    set cl_offset 0
    while {[scan $parsed_cl %4s%n cipherhex length] == 2} {
        if { [lsearch -sorted -inline $greaseList $cipherhex] eq "" } {
            lappend cipherSuite $cipherhex
            incr cipher_cnt
```

This iRule is 214 lines of code to generate a fingerprint for the browser/client



APM Header Echo

Logout

```
CERTOID: 2.16.840.1.101.2.1.11.42
CERTHEXOID: 0609608648016502010b2a
CERTSUBJECT: C=US, O=U.S. Government, OU=DoD, OU=PKI, OU=USAF, CN=GRABER.ANTHONY.JOHN.III.1042156821
CERTSERIAL: 0f:c4:ff
USERNAME: 1042156821157004
JA4 TLS Fingerprint: t13i1515h2_8daaf6152771_2e1f596fab39

Xja4: t13i1515h2_8daaf6152771_2e1f596fab39
Username: 1042156821157004
Certserial: 0f:c4:ff
Certsubject: C=US, O=U.S. Government, OU=DoD, OU=PKI, OU=USAF, CN=GRABER.ANTHONY.JOHN.III.1042156821
Certhexoid: 0609608648016502010b2a
Certoid: 2.16.840.1.101.2.1.11.42
Cookie: redirect=1; testing=1; adminer_sid=en8qoao4e55t7f9k97jn40tbag; adminer_key=7b9b74bc59fa1c7674ffa05c914f646e; sid=f6a534c504d97b6b1f7fbb1c2a8b0064; LastMRH_Session=79a40abf; TIN=295000;
F5_ST=1z1z1z1709140666z28800
Accept-Language: en-US,en;q=0.9
Accept-Encoding: gzip, deflate, br, zstd
Referer: https://192.168.10.160/my.policy
Sec-Ch-Ua-Platform: "macOS"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua: "Not A(Brand";v="99", "Google Chrome";v="121", "Chromium";v="121"
Sec-Fetch-Dest: document
Sec-Fetch-User: ?1
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Connection: keep-alive
Host: 192.168.10.160
Content-Length:
Content-Type:
```

App
World
2024

Appendix

Decision pages – confirm_box.inc

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html><head><title>Department of Defense</title>

<link rel="stylesheet" type="text/css" HREF="/public/include/css/apm.css<? if ($GLOBALS["ap_version"]=="v2") { print("?v=v2"); } ?>">
<script language="JavaScript" src="/public/include/js/session_check.js?v=13"></script>
<script language="JavaScript" src="/public/include/js/agent_common.js"></script>
<script language="javascript"><!--//

if(self != top) { top.location = self.location; }
window.onerror=function(){ return function(){ return; } }

<? include_customized_page("logout", "session_expired.js"); ?>
function sessionTimedOut()
{
    window.sessionTimeout.showSplashLayer("MessageDIV", SessionExpired_CustomizedScreenGet());
}

function OnLoad()
{
    setFormAttributeByQueryParams("hidden_form", "action", "/confirm.php3");

    try{
        if ( "undefined" != typeof(window.external) && "unknown" != typeof(window.external)
            && "undefined" != typeof(window.external.WebLogonNotifyUser) && "unknown" != typeof(window.external.WebLogonNotifyUser) ){
            window.external.WebLogonNotifyUser();
        }
```

Decision pages – decision_box.inc

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html><head><title>Department of Defense</title>

<link rel="stylesheet" type="text/css" HREF="/public/include/css/apm.css<? if ($GLOBALS["ap_version"]=="v2") { print("?v=v2"); } ?>">
<script language="JavaScript" src="/public/include/js/session_check.js?v=13"></script>
<script language="JavaScript"  src="/public/include/js/web_host.js"></script>
<script language="javascript"><!--//

if(self != top) { top.location = self.location; }
window.onerror=function(){ return function(){ return; } }

<? include_customized_page("logout", "session_expired.js"); ?>
function sessionTimedOut()
{
   try{
      if ( externalWebHost.hasWebLogonClearSession() ){
         externalWebHost.webLogonClearSession();
      }
   }catch(e){};

   window.sessionTimeout.showSplashLayer("MessageDIV", SessionExpired_CustomizedScreenGet());
}

function OnLoad()
{
   try{
```

UPN and username variable assign

session.logon.last.upn

```
set x509e_fields [split [mcget {session.ssl.cert.x509extension}] "\n"];
# For each element in the list:
foreach field $x509e_fields {
# If the element contains UPN:
if { $field contains "othername:UPN" } {
## set start of UPN variable
set start [expr {[string first "othername:UPN<" $field] +14}]
# UPN format is <user@domain>
# Return the UPN, by finding the index of opening and closing brackets, then use string range to get everything between.
return [string range $field $start [expr { [string first ">" $field $start] - 1 } ] ];  } }
# Otherwise return UPN Not Found:
return "UPN-NOT-FOUND";
```

session.logon.last.username

```
set upn [mcget {session.logon.last.upn}]; if {[string first "@" $upn] >= 0} {
return [string range $upn 0 [expr { [string first "@" $upn] - 1 } ] ]; } else { return $upn; }
```

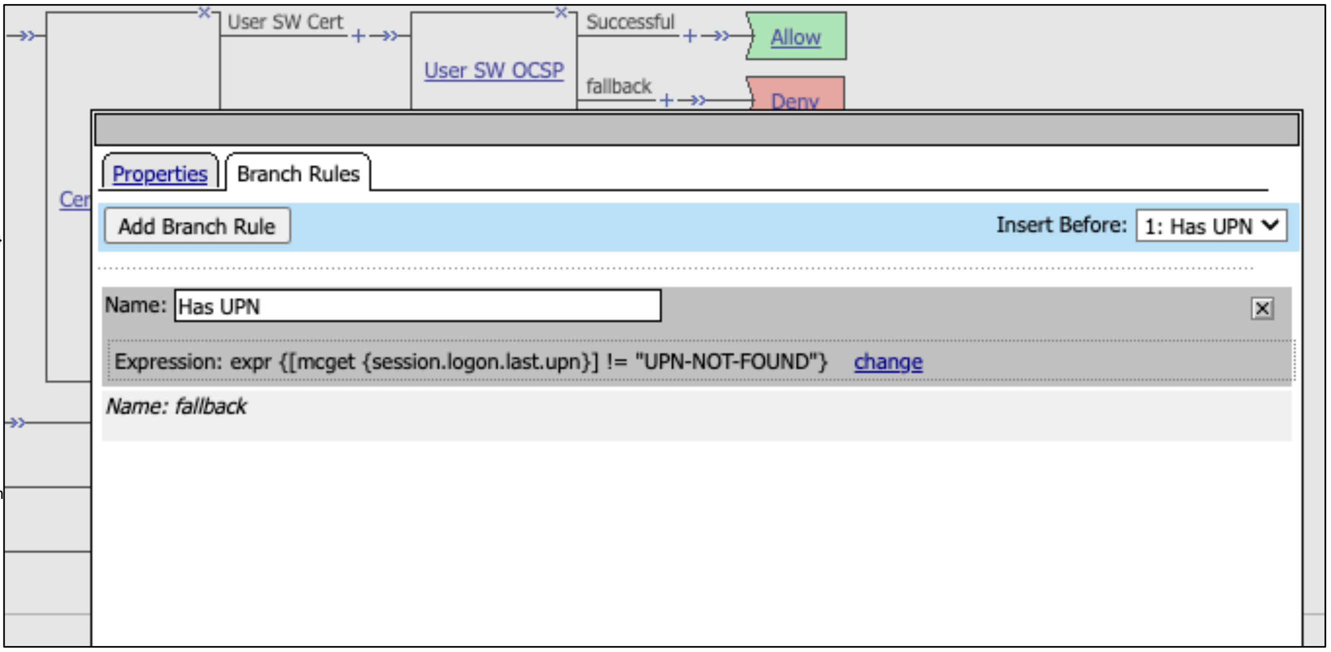
UPN and username variable assign

session.logon.last.upn

```
set x509e_fields [split [mcget {session.ssl.cert.x509extension}] "\n"];
# For each element in the list:
foreach field $x509e_fields {
# If the element contains UPN:
if { $field contains "othername:UPN" } {
## set start of UPN variable
set start [expr {[string first "othername:UPN<" $field] +14}]
# UPN format is <user@domain>
# Return the UPN, by finding the index of opening and closing brackets, then
return [string range $field $start [expr { [string first ">" $field $start] - 1 } ] ];  } }
# Otherwise return UPN Not Found:
return "UPN-NOT-FOUND";
```

session.logon.last.username

```
set upn [mcget {session.logon.last.upn}]; if {[string first "@" $upn] >= 0} {
return [string range $upn 0 [expr { [string first "@" $upn] - 1 } ] ]; } else { return
```



Insert Before: 1: Has UPN

Name: Has UPN

Expression: expr {[mcget {session.logon.last.upn}] != "UPN-NOT-FOUND"} change

Name: fallback

CERTOID Check

```
when ACCESS_POLICY_AGENT_EVENT {
    if { [ACCESS::policy agent_id] eq "OIDCHECK" } {
    ## find and store CERT OID
        if { [ACCESS::session data get session.ssl.cert.x509extension] contains "Policy: " } {
        ACCESS::session data set session.custom.oid "[string trim [findstr [ACCESS::session data get session.ssl.cert.x509extension] "Policy: " 8 " "]]"
        }
    }
}


when ACCESS_ACL_ALLOWED {
 #### OPTIONAL HEADER INSERTS
    HTTP::header insert CERTOID [ACCESS::session data get session.custom.oid]
    HTTP::header insert CERTSUBJECT [ACCESS::session data get session.ssl.cert.subject]
    HTTP::header insert CERTSERIAL [ACCESS::session data get session.ssl.cert.serial]
    HTTP::header insert USERNAME [ACCESS::session data get session.logon.last.username]
}
```

Cert OID Check Branch Rules

User SW Cert

```
expr { [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.39" || [mcget {session.custom.oid}]
== "2.16.840.1.101.2.1.11.40" || [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.41" }
```

User HW Cert

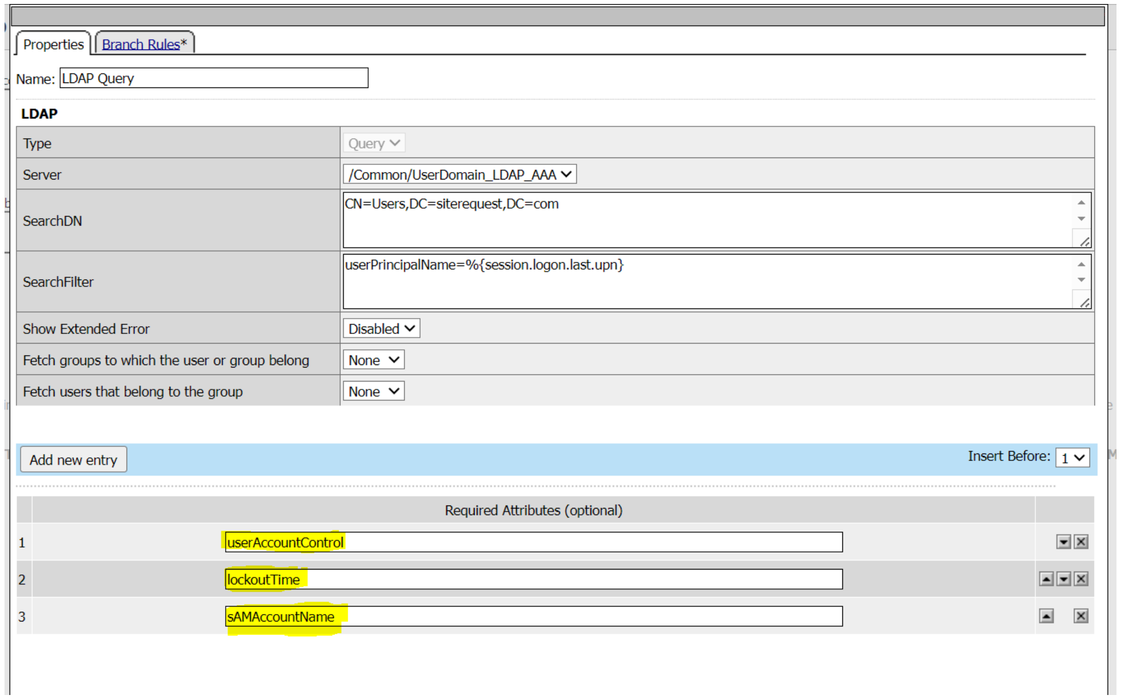
```
expr { [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.42" || [mcget {session.custom.oid}]
== "2.16.840.1.101.2.1.11.43" || [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.44" }
```

NPE Cert

```
expr { [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.36" || [mcget {session.custom.oid}]
== "2.16.840.1.101.2.1.11.37" || [mcget {session.custom.oid}] == "2.16.840.1.101.2.1.11.38" }
```

LDAP Query Branch Rule

```
expr {[mcget {session.ldap.last.queryresult}] == 1 && [mcget {session.ldap.last.attr.userAccountControl}] != 66050 &&  [mcget {session.ldap.last.attr.lockoutTime}] == 0 }
```



Name: LDAP Query

LDAP

Type	Query
Server	/Common/UserDomain_LDAP_AAA
SearchDN	CN=Users,DC=siterequest,DC=com
SearchFilter	userPrincipalName=%{session.logon.last.upn}
Show Extended Error	Disabled
Fetch groups to which the user or group belong	None
Fetch users that belong to the group	None

Insert Before: 1

	Required Attributes (optional)
1	userAccountControl
2	lockoutTime
3	sAMAccountName

Serial Number Binding

```
# APM_X509_SN_Binding
# Copyright 2024 F5
# Binds an APM session to a X509 serial number from an mTLS connection
# invalidates SSL session and removes APM session on mismatch
# requires "client-certificate required" to be present in the clientssl profile
# compatible with TLS 1.2/1.3
when ACCESS_ACL_ALLOWED priority 100 {
   set tuple [IP::local_addr]:[TCP::local_port]->[IP::remote_addr]:[TCP::remote_port]
   # ensure client certificate is present
   if {[SSL::cert count] eq 0} {
      ACCESS::log accesscontrol.warn "APM_X509_SN_Binding - No Client Certificate present $tuple "
      ACCESS::respond 403 content {<html><h1>Access Denied</h1></html>}
      ACCESS::session remove
      SSL::session invalidate
      return
   }

   set sn [X509::serial_number [SSL::cert 0]]

   # does client-certificate serial match serial stored with APM session
   if {[ACCESS::session data get "session.ssl.cert.serial"] eq $sn} {
      ACCESS::log accesscontrol.info "APM_X509_SN_Binding - Client Certificate SN match: $sn"
      return
   }
   ACCESS::log accesscontrol.warn "APM_X509_SN_Binding - Attempted session hijack from $tuple with mismatched Client Certificate SN: $sn"
   ACCESS::respond 403 content {<html><h1>Access Denied</h1></html>}
```