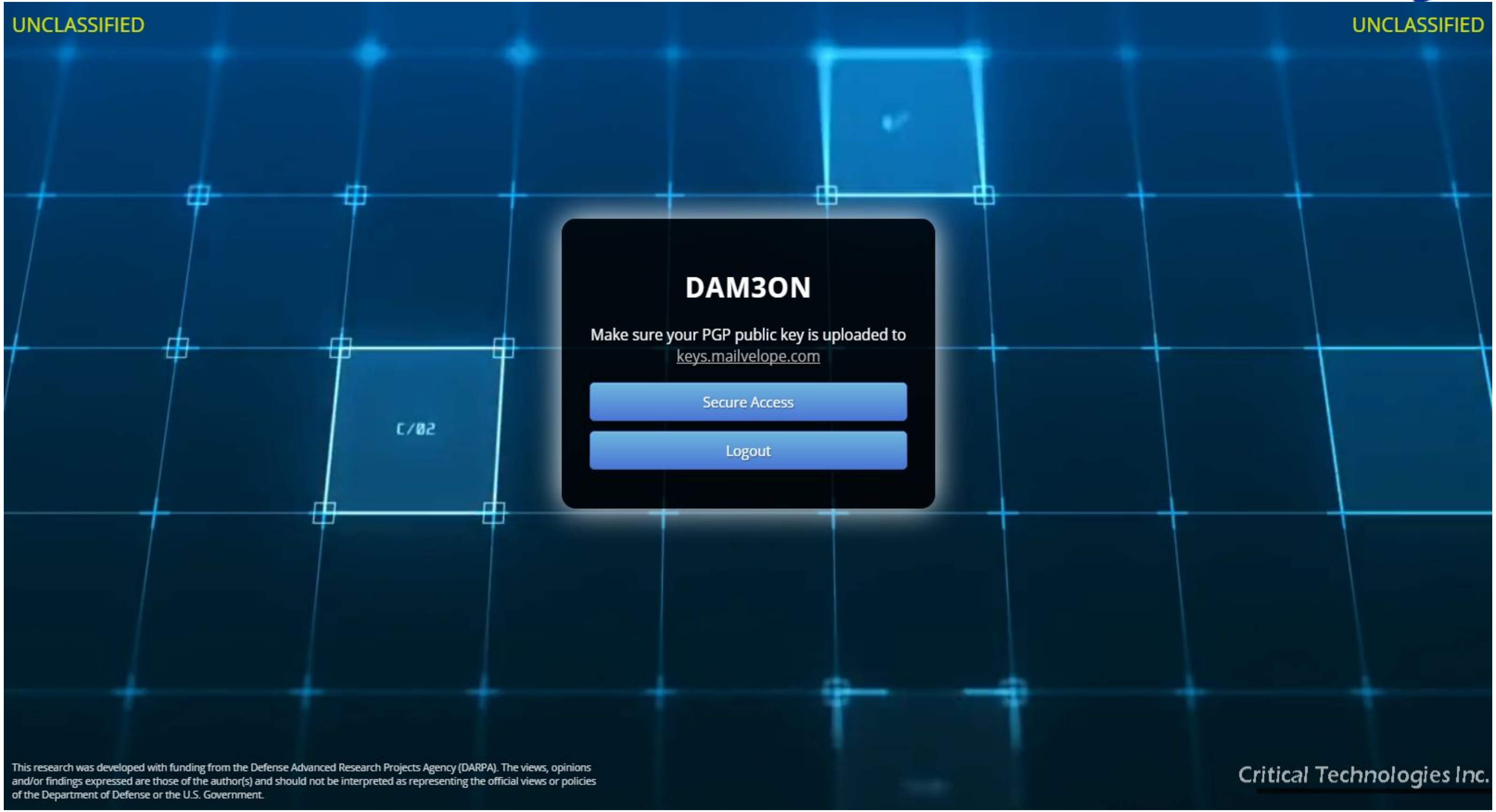
Secure Software Components Leveraging the seL4 Microkernel



UNCLASSIFIED UNCLASSIFIED

DAM3ON

Make sure your PGP public key is uploaded to keys.mailvelope.com

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

Critical Technologies Inc.

Our Goal

- Implement a way of *proving* you or your platform are trustworthy
- Create remote attestation scheme
- A new dynamic access control implementation
  - XACML implementation that includes making decisions based on remote attestation
    - Uses known security properties, or SAML (Security Assertion Markup Language) certificates that assert properties
    - Includes more traditional restrictions as well (geo-location, roles, ip white/blacklist, etc.)
    - Allows authentication without providing a password to the system (PGP)
  - Allow access to a resource based on properties of a system, rather than a specific configuration

Why it’s important

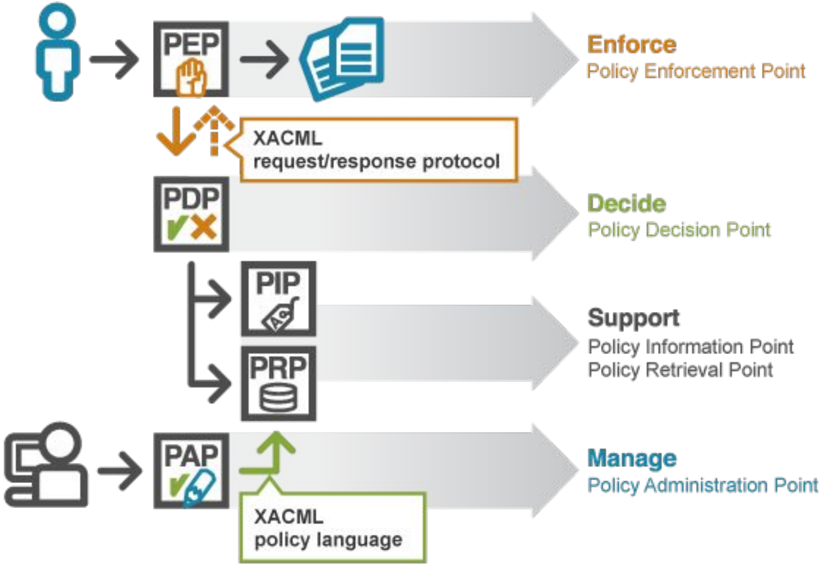
- Gives users the freedom to use software of their choice as long as they can attest to security properties required by policy
- No longer tied to using proprietary, closed-source software to access a resource
- User can choose their own software that may have all the properties required by several different resources, rather than using different software for each
- Allows the user to have a choice
- Gives companies much greater flexibility in writing access control policies
- Gives users much greater flexibility in satisfying those policies
- Allows a user to be anonymous
- Allows a user to not be anonymous
    - Don’t have to provide a password to a company
    - Company doesn’t have to store any information on a user (i.e. liability)

Secure Login

- DAM3ON allows users to securely access an arbitrary resource by:
    - Anonymous attestation
    - Authentication
    - Authentication + Attestation
    - None of the above

Access Control

Our modified XACML instance provides the decision engine, policy language, and enforcement points - among other things.



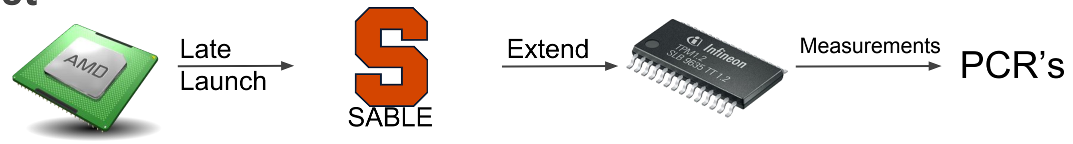
Access Control

XACML allows for very powerful policies. For example a policy can state things like:

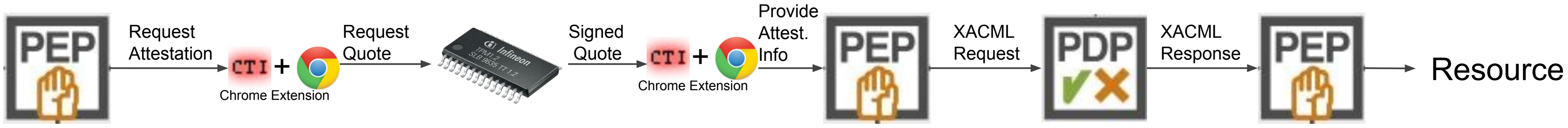
- A user can only access a resource if:
  - Their IP is in a whitelisted range
  - And they are located within a geo-fence
  - And they provide a valid PGP signature
  - And they attest that they are running SABLE v.05
  - And they attest they have some security property “P”
- Or
  - A user is New York OR Pennsylvania
  - User has a role of Admin
  - User is not on a blacklisted IP range
  - Etc.
- Or
  - Any number of restrictions (role, PGP signature, IP, geo location, attestation, etc.) that common logic statements can be applied to (AND, OR, set theory, custom functions, etc.)

Security & Trust Starting at Layer 0

On Boot



On Attestation



Authentication

- Uses PGP/GPG or Smart Cards
  - Tested & Tried & true
- User logins by signing a Subject, Action, Resource, and a nonce with their PGP key
  - Subject: who is trying to perform this action on this resource (bob@domain.com, anonymous, etc.)
  - Action: what the user wants to do with that resource (download, upload, access, view, etc.)
  - Resource: the thing a user wants to access (website, file, etc.)
  - Nonce: to prevent replay attacks
- If PGP we then retrieve asserted user's signature from an open-source, secure key-server (Mailvelope’s). If we can verify their signature with the key from the key-server, then we know that this person is who they claim to be
  - Mailvelope’s key-server will only add people who go through their process of: uploading a public key, then Mailvelope encrypts a message to that key, and sends it to the email address specified in the key. If the user can decrypt that message, and click on a verification URL, the key is added to their key-server

Authentication - Mailvelope

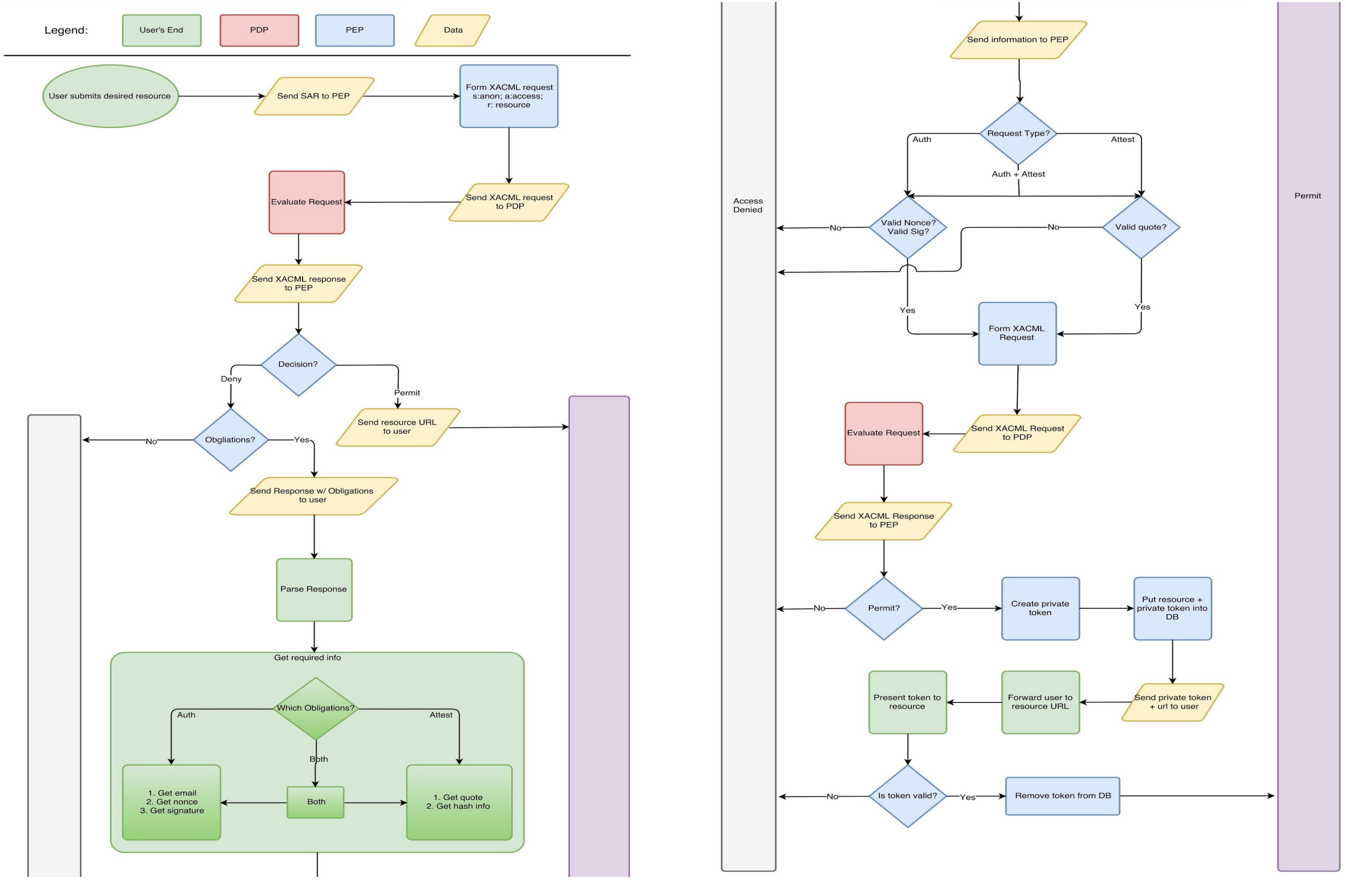
- While not developed by CTI, Mailvelope is a natural fit
- Mailvelope is an open-source browser plugin project based out of Germany
- Provides very friendly usage & access to using PGP
- Works as browser addon for major browsers. Will prompt to encrypt text input areas on specified websites (i.e. gmail.com, outlook.com)
- Mailvelope provides their own open source key-server as well
- Ultimately can allow a user to sign an arbitrary piece of information with their PGP key with only a couple clicks + password, which vastly reduces the barrier to entry with using PGP. No need for the command line, or difficult to use tools.

Attestation

- Done using a “TPM quote” from a TPM chip
- A TPM quote is a cryptographic signature on an arbitrary set of PCR values as they currently exist on a machine (PCR values being registers for measurements done by the TPM). A quote also includes a nonce, and is signed by an AIK (Attestation Identity Key)
- When attesting:
  - Quote is validated & verified by server (PEP)
  - After validation & verification, list of binaries comprising the quote are known, which are then used later on to look up known properties about them. If no properties are known, user is prompted to provide a URL to a certificate that asserts properties.
  - PEP then forms a request as usual, but now including a list of binary hashes, and sends to PDP
  - PDP, by means of a PIP (Policy Information Point) looks up which security properties are associated with each binary, which in turn is then used to see if a user satisfies that portion of policy

Login System

- Dynamically supports all login types from the same portal (attestation, authorization, both, neither)
  - Dynamically determines what forms to present to the user & what to do with them
- Attention to detail to make this accessible as possible
  - If it’s not easy to use, nobody will use it
  - Everything is automated except for the very few pieces that are absolutely required by a human



Legend: User's End PDP PEP Data

User submits desired resource

Send SAR to PEP

Form XACML request s:anon; a:access; r: resource

Send XACML request to PDP

Evaluate Request

Send XACML response to PEP

Decision?

Deny

Permit

Send resource URL to user

Obgliations?

No

Yes

Send Response w/ Obligations to user

Parse Response

Get required info

Which Obligations?

Auth

Attest

Both

1. Get email
2. Get nonce
3. Get signature

Both

1. Get quote
2. Get hash info

Send information to PEP

Request Type?

Auth

Attest

Auth + Attest

Access Denied

Permit

Valid Nonce? Valid Sig?

No

Valid quote?

No

Yes

Yes

Form XACML Request

Send XACML Request to PDP

Evaluate Request

Send XACML Response to PEP

Permit?

No

Yes

Create private token

Put resource + private token into DB

Send private token + url to user

Forward user to resource URL

Present token to resource

Is token valid?

No

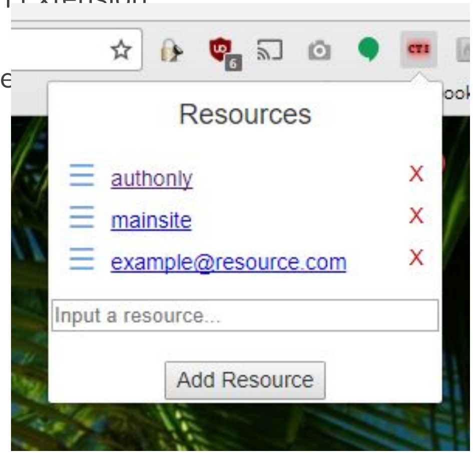
Yes

Remove token from DB

CTI Extension + Native Host

- CTI Extension
  - Responsible for communicating between a website and a native host
    - i.e. can communicate with cooperating native programs
  - Also provides small UI that provides a list of commonly used resources for a user
- Native Host
  - Native program responsible for communicating between TPM and the CTI Extension
  - Required for attestation
  - Prompts user for certificate input on binaries the server hasn't seen before

Website ←→ CTI Extension ←→ Native Host ←→ TPM

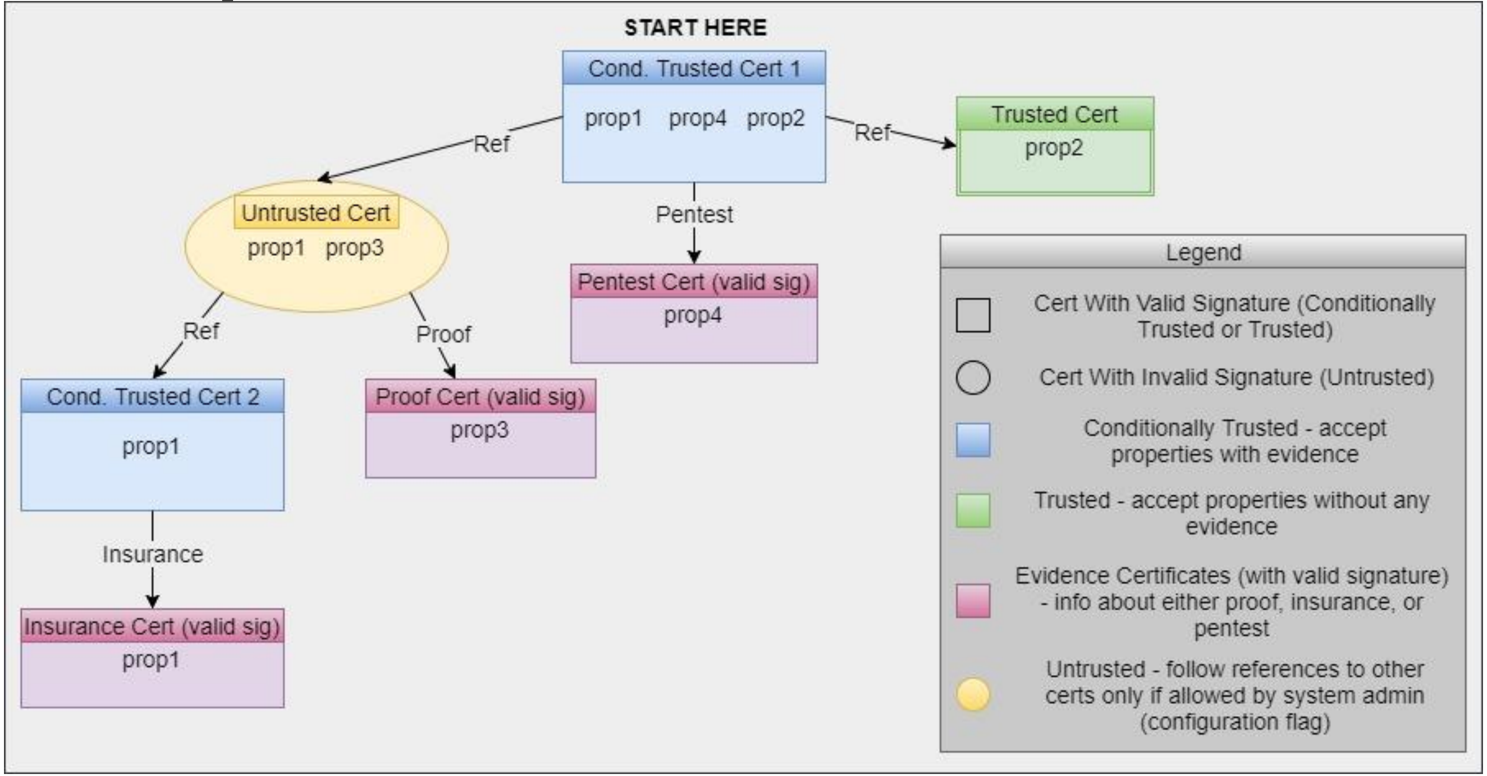

Resources

- authonly X
- mainsite X
- example@resource.com X

Security Property Certificates

- Custom XML schema based on Security Assertion Markup Language (SAML)
  - Still conforms to SAML 2.0 specification
  - More restrictive: requires all information that we need
- 4 Types of Certificates:
  - Normal Certificate - asserts a list a properties about a subject; each property has either a reference to another normal cert or a link to one of the other 3 types
  - Proof Certificate - list of properties to which this proof is about and information about how to validate the proof (ie. proof checker version, arguments, etc.)
  - Insurance Certificate - list of properties to be insured and information regarding the insurance policy (ie. insurance company, liability amount, etc.)
  - Pentest Certificate - list of properties for which a pentest was performed and information about the pentest (ie. pentesting organization, contact info, etc.)
- XML + XSL styling so certificates can be both machine and human readable

Example Certificate Chain



START HERE

Cond. Trusted Cert 1
prop1 prop4 prop2

Ref

Trusted Cert
prop2

Ref

Untrusted Cert
prop1 prop3

Pentest

Pentest Cert (valid sig)
prop4

Ref

Proof

Cond. Trusted Cert 2
prop1

Proof Cert (valid sig)
prop3

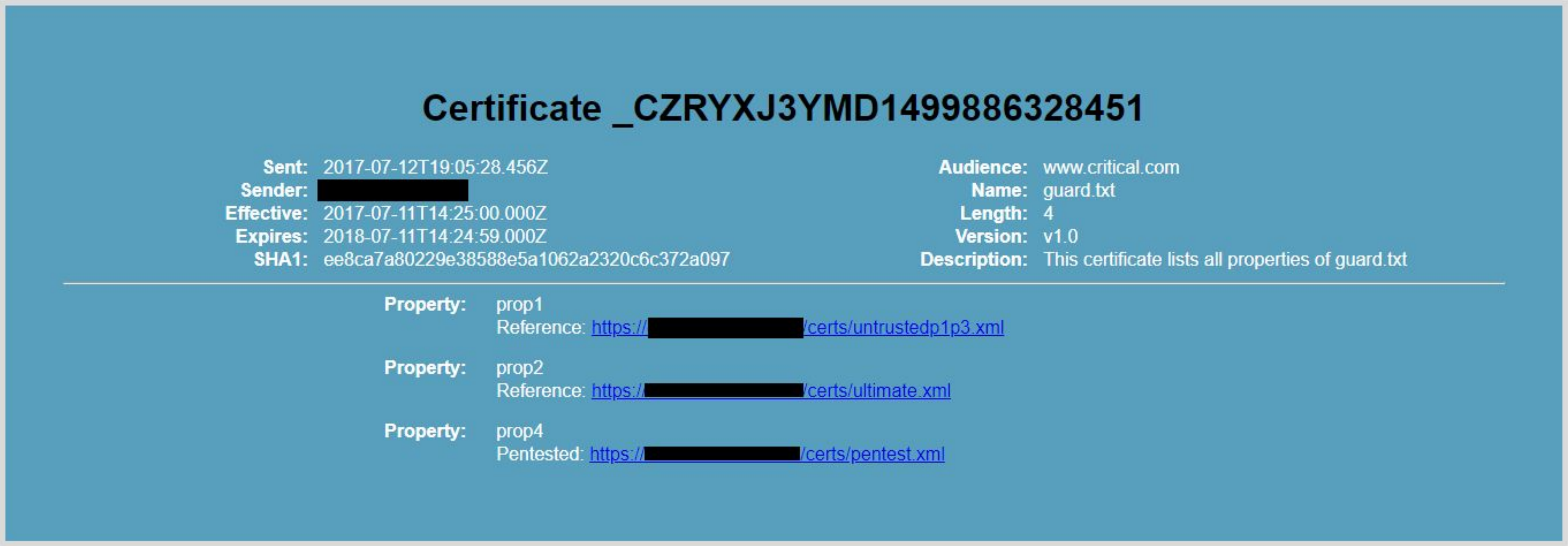
Insurance

Insurance Cert (valid sig)
prop1

Legend

□	Cert With Valid Signature (Conditionally Trusted or Trusted)
○	Cert With Invalid Signature (Untrusted)
🟦	Conditionally Trusted - accept properties with evidence
🟩	Trusted - accept properties without any evidence
🟪	Evidence Certificates (with valid signature) - info about either proof, insurance, or pentest
🟡	Untrusted - follow references to other certs only if allowed by system admin (configuration flag)

Example Certificate (human readable)



Certificate _CZRYXJ3YMD1499886328451

Sent: 2017-07-12T19:05:28.456Z
Sender: [redacted]
Effective: 2017-07-11T14:25:00.000Z
Expires: 2018-07-11T14:24:59.000Z
SHA1: ee8ca7a80229e38588e5a1062a2320c6c372a097

Audience: www.critical.com
Name: guard.txt
Length: 4
Version: v1.0
Description: This certificate lists all properties of guard.txt

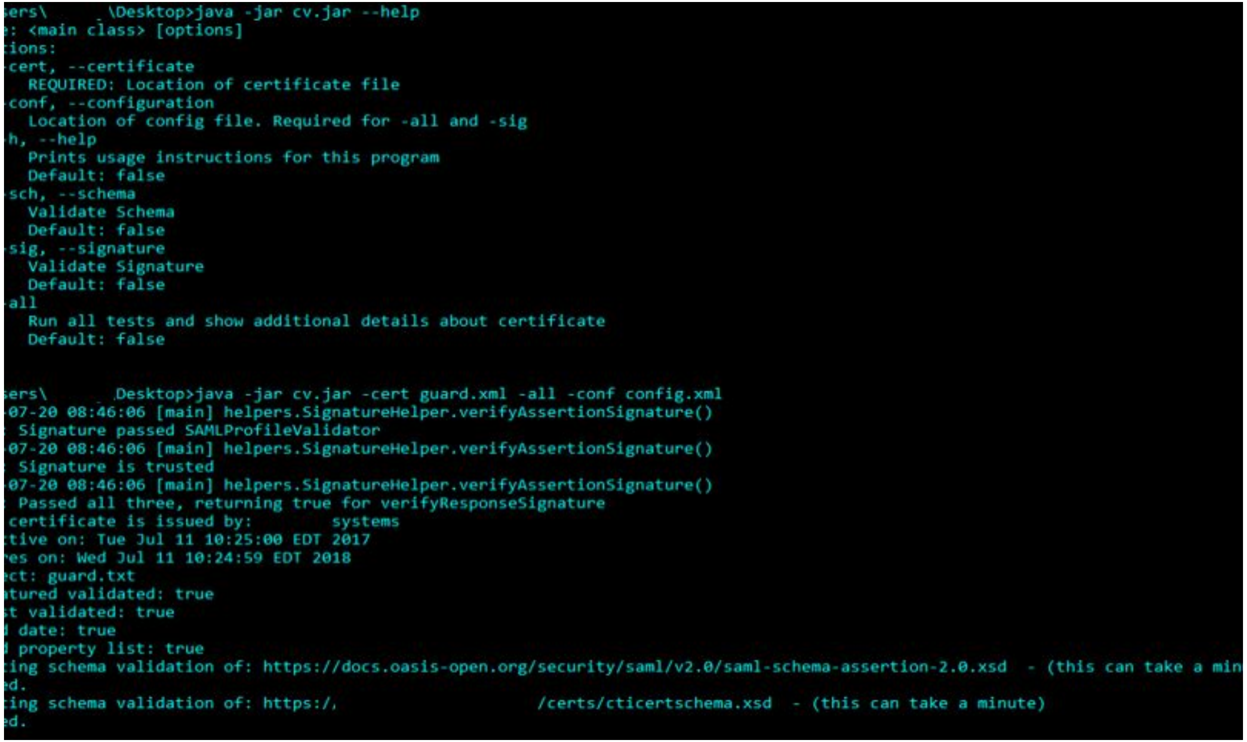
Property: prop1
Reference: https://[redacted]/certs/untrustedp1p3.xml

Property: prop2
Reference: https://[redacted]/certs/ultimate.xml

Property: prop4
Pentested: https://[redacted]/certs/pentest.xml

Certificate Validator

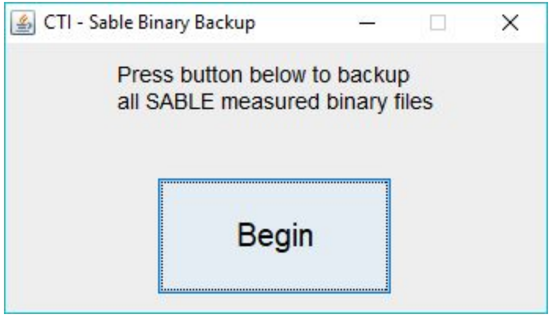
Small systems administrator tool for viewing, validating, and verifying certificates



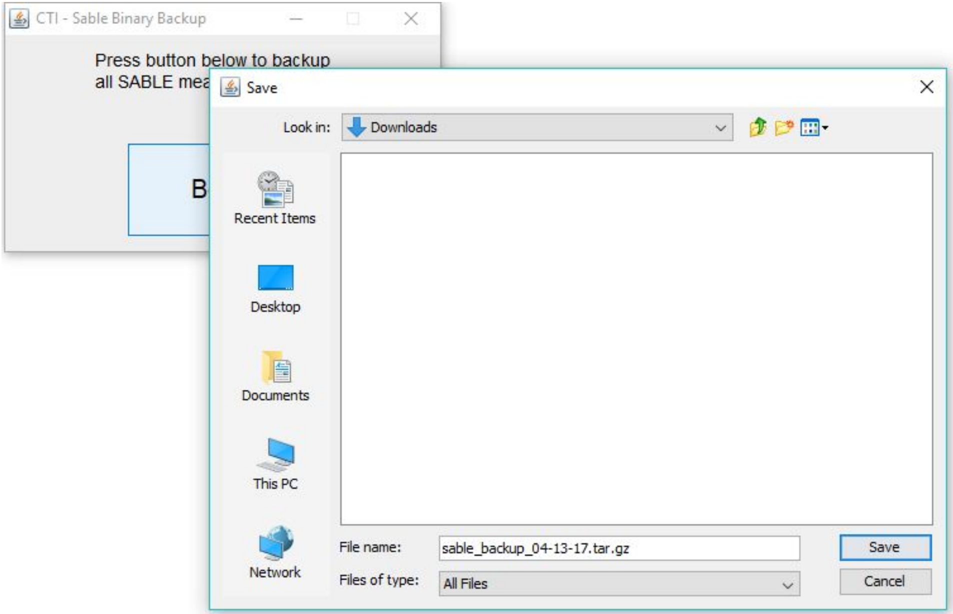
```
ers\      \Desktop>java -jar cv.jar --help
: <main class> [options]
ions:
cert, --certificate
  REQUIRED: Location of certificate file
conf, --configuration
  Location of config file. Required for -all and -sig
h, --help
  Prints usage instructions for this program
  Default: false
sch, --schema
  Validate Schema
  Default: false
sig, --signature
  Validate Signature
  Default: false
all
  Run all tests and show additional details about certificate
  Default: false


ers\       Desktop>java -jar cv.jar -cert guard.xml -all -conf config.xml
07-20 08:46:06 [main] helpers.SignatureHelper.verifyAssertionSignature()
 Signature passed SAMLProfileValidator
07-20 08:46:06 [main] helpers.SignatureHelper.verifyAssertionSignature()
 Signature is trusted
07-20 08:46:06 [main] helpers.SignatureHelper.verifyAssertionSignature()
 Passed all three, returning true for verifyResponseSignature
certificate is issued by:        systems
tive on: Tue Jul 11 10:25:00 EDT 2017
es on: Wed Jul 11 10:24:59 EDT 2018
ct: guard.txt
tured validated: true
t validated: true
 date: true
 property list: true
ing schema validation of: https://docs.oasis-open.org/security/saml/v2.0/saml-schema-assertion-2.0.xsd  - (this can take a min
d.
ing schema validation of: https:/,                    /certs/cticertschema.xsd  - (this can take a minute)
d.
```

Binary Backup Program


CTI - Sable Binary Backup

Press button below to backup
all SABLE measured binary files


CTI - Sable Binary Backup

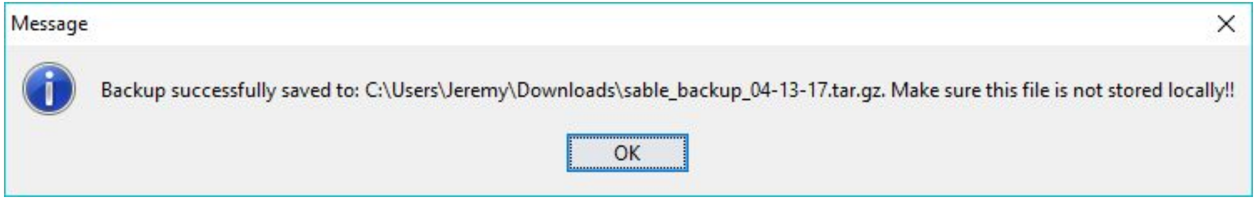
Press button below to backup
all SABLE mea

Save

Look in: Downloads

File name: sable_backup_04-13-17.tar.gz

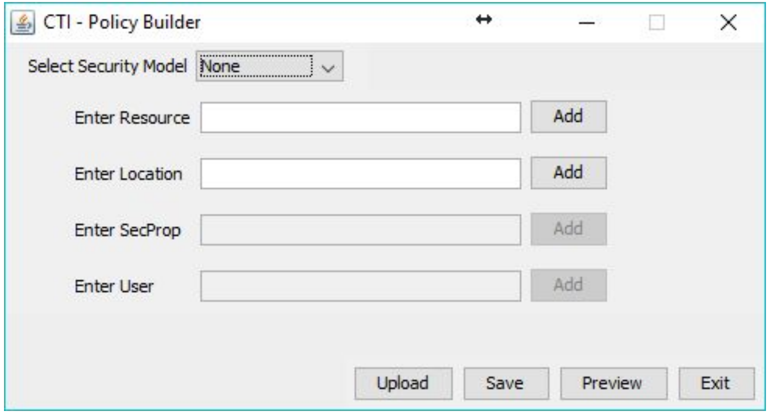
Files of type: All Files


Message

Backup successfully saved to: C:\Users\Jeremy\Downloads\sable_backup_04-13-17.tar.gz. Make sure this file is not stored locally!!

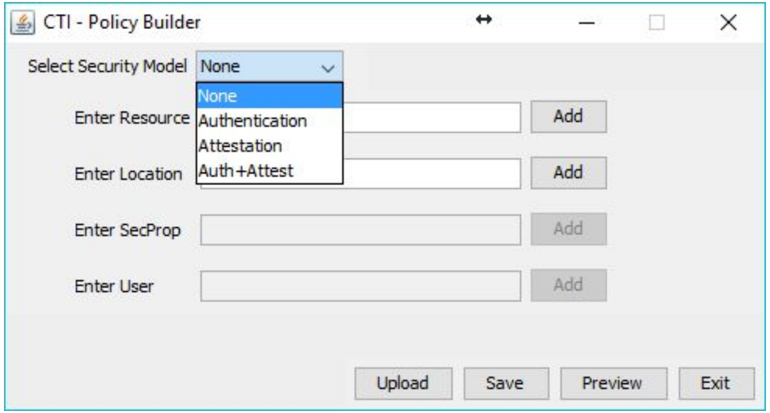
Policy Creator (deprecated)

- Simple GUI to create basic XACML 2.0 policies
- Select security model and add multiple resources, locations, security properties, or specific users



CTI - Policy Builder

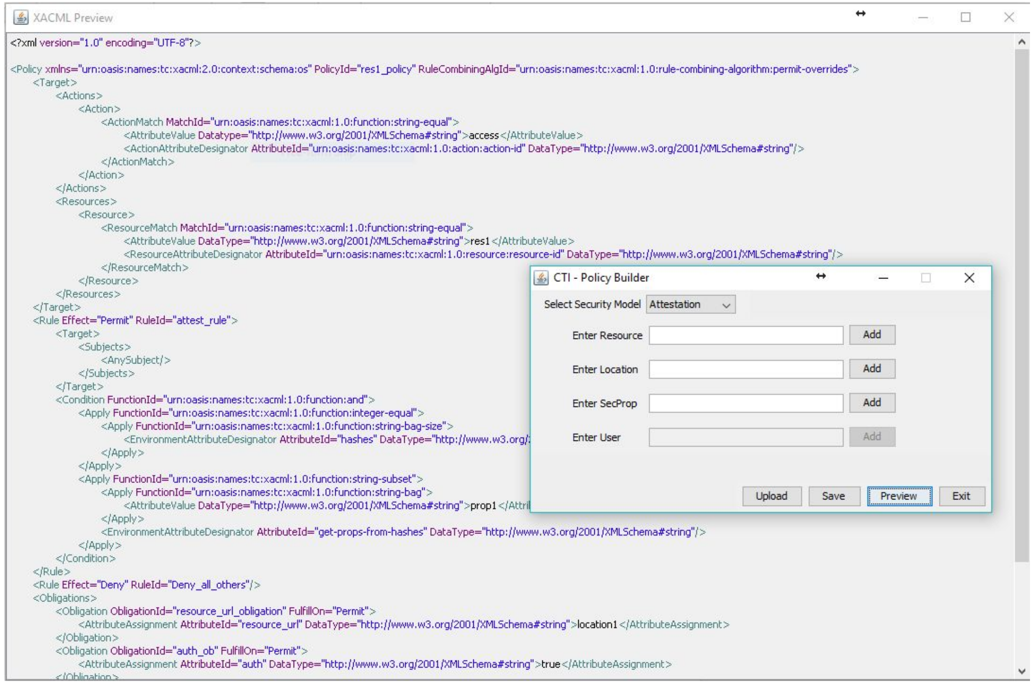
Select Security Model: None



CTI - Policy Builder

Select Security Model: None

Policy Creator (deprecated)



```
<?xml version="1.0" encoding="UTF-8"?>

<Policy xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os" PolicyId="res1_policy" RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-overrides">
    <Target>
        <Actions>
            <Action>
                <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue Datatype="http://www.w3.org/2001/XMLSchema#string">access</AttributeValue>
                    <ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id" DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
        </Actions>
        <Resources>
            <Resource>
                <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">res1</AttributeValue>
                    <ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id" DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ResourceMatch>
            </Resource>
        </Resources>
    </Target>
    <Rule Effect="Permit" RuleId="attest_rule">
        <Target>
            <Subjects>
                <AnySubject/>
            </Subjects>
        </Target>
        <Condition FunctionId="urn:oasis:names:tc:xacml:1.0:function:and">
            <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-equal">
                <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-bag-size">
                    <EnvironmentAttributeDesignator AttributeId="hashes" DataType="http://www.w3.org/
                </Apply>
            </Apply>
            <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-subset">
                <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-bag">
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">prop1</Attri
                </Apply>
                <EnvironmentAttributeDesignator AttributeId="get-props-from-hashes" DataType="http://www.w3.org/2001/XMLSchema#string"/>
            </Apply>
        </Condition>
    </Rule>
    <Rule Effect="Deny" RuleId="Deny_all_others"/>
    <Obligations>
        <Obligation ObligationId="resource_url_obligation" FulfillOn="Permit">
            <AttributeAssignment AttributeId="resource_url" DataType="http://www.w3.org/2001/XMLSchema#string">location1</AttributeAssignment>
        </Obligation>
        <Obligation ObligationId="auth_ob" FulfillOn="Permit">
            <AttributeAssignment AttributeId="auth" DataType="http://www.w3.org/2001/XMLSchema#string">true</AttributeAssignment>
        </Obligation>
```

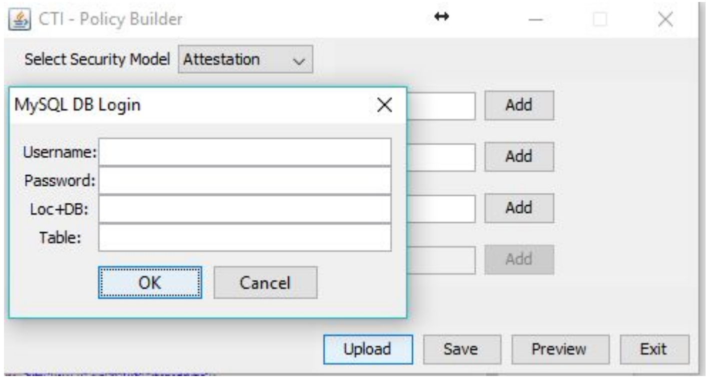
Select Security Model: Attestation

Enter Resource:

Enter Location:

Enter SecProp:

Enter User:



Select Security Model: Attestation

Username:

Password:

Loc+DB:

Table:

File-by-file Access

- Also added the ability to restrict files in a git-annex repository on a file-by-file or directory basis.
    - allows a user to enter a URI to a file, directly into the resource login page. For example, a user could enter “file://Example_repo/subdir/file.txt”.
    - just as any other resource, the user may be asked to authenticate or attest based on what the matching policy requires. Policies can be written around files, sub directories, the top level directory, or any combination of those locations. This is currently a work in progress, but is almost finished.
- As part of the request process, the encrypted capability for the requested file is sent along in the XACML request to access the resource. If permitted, the PDP will decrypt the capability, and return it to the requesting PEP.
- The user is then sent to a page where that decrypted capability is “cashed in”, and used to retrieve the actual file, and ultimately serve it to the requesting user. Immediately after the file is retrieved, the capability is re-encrypted and the file is dropped from the PEP machine.

Future

- Bring prototype to a market ready product
  - Polish existing administrator tools
  - Upgrade installer to include dependencies and default configuration
  - Create new administrator tools to create more dynamic policies easily, without having to learn XACML
  - Other Administration tools (System monitor, configuration editor, etc.)
  - In general, polishing this product such that it can be deployed to a mass market in hopes of changing how the world views security & trust. This project must be as polished as possible for it to have a chance of being adopted
- Upgrade DAM3ON to TPM2.0
  - A more secure, modern version of TPM
- Evaluate feasibility of porting TPM related features to Windows 10
- Further testing, evaluation, and documentation