NASA Web Browser Cybersecurity Plan

# Written by Brian Russel Davis

## Version 1.0.0

## 2/17/2019

### 

### 

## Introduction

Safety, for individuals and for NASA as an organization, is a mantra. It is a mantra we practice when engineering rockets, when studying the bleeding edge science or when tackling our own Cybersecurity concerns. Since the Web Browser sits at the center of our agency's communication, it only makes sense that we enact a Web Browser Security Plan to ensure the safety of users and NASA as an organization. This plan will be enacted for all users in the Agency that use a web browser.

## Security Policy

This plan will institute mandatory and safe web browsing practices enforced with a custom distribution of the Chromium Browser. This new custom version of Google’s popular Chrome Browser will allow users to browse with security, safety, and confidence. This new browser, codenamed Jemison, will also enact a new form of access control, one based on discretionary access but bolstered by the power of Artificial Intelligence. Together these tools will help the agency ensure the confidentiality of our most sensitive scientific engineering and personal information, maintain the integrity of our systems and at the same time ensure access and availability to the users that rely on the information inside of these systems for their daily work.

### Security Principles

At NASA confidentiality is paramount. The information housed in the Agency’s walls is often considered the most sensitive and highly prized scientific data in the world. The scientific information used in space aviation, military weapon systems, and advanced research initiatives are among the most sensitive at NASA. Protecting this information and the systems that house it means implementing strict control on access, continuous scanning/monitoring for malicious threats and proactive intelligent threat hunting. When these tools are in the hands of trained staff, they are reinforced with the practice of good cyber hygiene and are referenced in engineering, human resource and communication policies, and practices, they form a web of integrated protection for the Agency and those that work in it. These principles can be distilled into the following three verbs:

**Look**. Use automation to actively look for threats where they most commonly appear. Threat hunting is usually something most organization consider aggressive. However, since safety is a mantra, aggressive threat hunting all the way down to actively looking at browser downloads is necessary. Creating automated flexible policies that can be enforced by the Jemison is an important part of this security plan.

**Listen**. Use automation and artificial intelligence to actively monitor changes in user access, the movement of data and activity created by applications. Create policies that can be automated and easily updated according to threat intelligence.

**Act**. Act responsibly and in a discretionary manner when giving access to users. Creating a system of access as a dynamic calculation of usage and trust is more secure that one based on static roles that may be compromised and misused.

## Physical, logical, and organizational security policies

### Physical Security

To ensure the physical security in relation to web browsing this security plan will implement two important policies that will impact the machines overall cyber safety.:

1. No CD or USB Access to machines. CD Drives and USB ports are an easy vector for attackers who like to then to introduce trojan or malware. Furthermore, because of the strength of the network, we don’t need to deliver information physically anymore. Researchers who need access to large datasets will access Virtual Machines with the ability to use large data sets in a manner that does not necessitate the movement of data.

2. Machines with sensitive data stay in-house. Machines with sensitive materials or elevated access will not be allowed outside of the NASA campus.

### Logical Security

1. Limited Discretionary Time Bound Access. Root access to machines for dev-ops engineering staff only, and only for limit periods of time.

2. Use of Virtual Machines for Access to Sensitive Data. Software Engineering staff will use virtual machines for development, not physical machines with root access. VM’s will use a cryptologic key authenticate the VM’s access rights.

### Organizational Security

Organizational security in the context of this plan means defining a plan for access to applications and files found in the private cloud network. Almost all of the files and applications are accessed from the Browser, so it makes sense that the Access Control System be integrated into Jemison’s Policy Agent. Jemison’s Dynamic Relational Trust-Based Access Control ( DRTBAC ) will be explained in the Technical section below.

## Security Measures

## Browser Security

**Browsers** are an application launchpad, an application container and the door to data storage. Desktops, laptops, phone, tablets and even some TVs have browsers. This security plan and the development efforts will include all devices that contain a Browser, or the capabilities to have a browser.

###### *General Audience*

**Users** operate browsers and their actions in the browser may cause the organization to be vulnerable to attacks. We need a way to **allow most actions**, but **limit the scope of these actions** so mistakes don’t lead breaches. Therefore our security plan will focus on creating smart dynamic policies around user behavior in the browser, policies that protect users and do not impede their work.

###### *Affected Systems*

**Browser Navigation Rules** or decisions about websites users can visit are usually handled by **endpoint protection**. “*The components involved in aligning the endpoint security management systems include a Virtual private network (VPN) client, an operating system and updated antivirus software.*”[[1]](#footnote-0) However, there are many instances when the user intentionally or unintentionally turns VPN and antivirus OFF, breaking most traditional endpoint detection. The solutions discussed here will seek to protect the user, even in these two layers of protection are gone.

###### *Development Stakeholders*

**Product Managers from Security Operations** will be responsible for creating the product requirements.

**System Engineering** will be responsible for the technical requirement and the developing solution.

**Quality Assurance** will be responsible for developing acceptance criteria and performing UAT.

##### Technical Description

Chromium development starts with understanding the Chromium code base. First of all, Chromium is not the Chrome Browser itself. The [Chromium Open Source Project](https://chromium.googlesource.com/chromium/src/), much like the Mozilla Project, is supported by the community. Also, “Chromium's multi-process architecture is a radical departure from other web browsers.” [[2]](#footnote-1) This multi-process architecture makes each tab its own process managed by the Chromium Central Process Manager. Our approach is to leverage this architecture to create dynamic policy enforcement while preserving the user experience and performance.

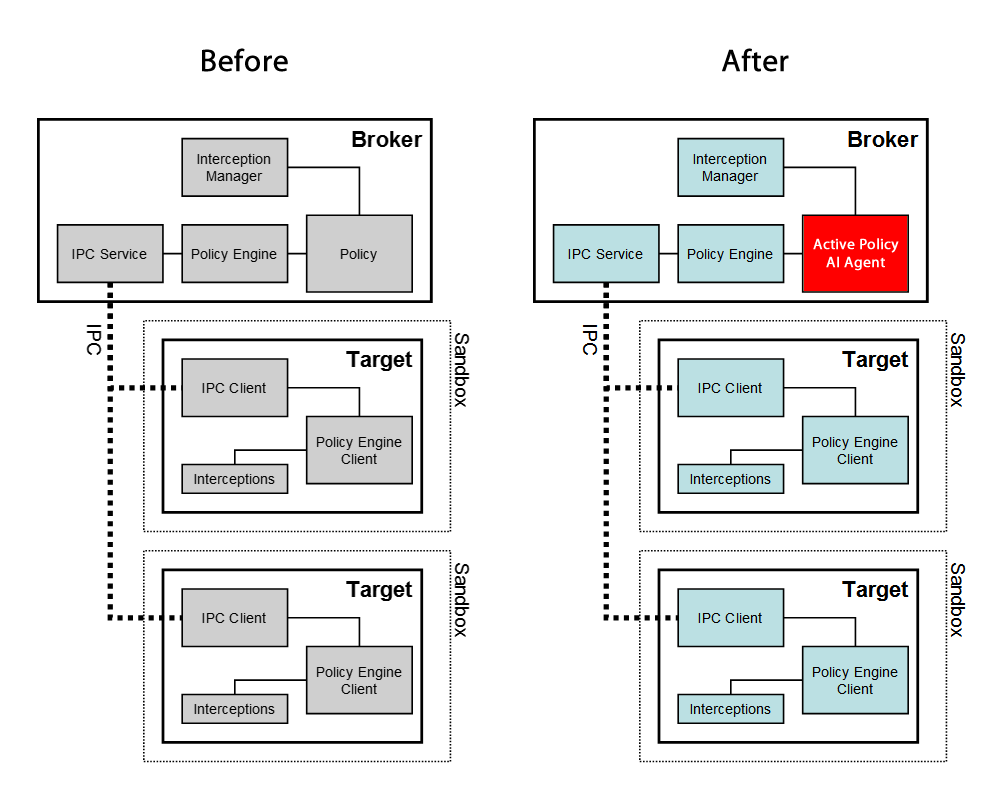
###### *Chromium Central Process Manager*

Separate processes for each tab helps achieve a number of amazing innovations. First, the code running in that tab’s process also has its own thread and its own memory block. Here is an overview of the sandboxing architecture :

*“Given the renderer is running in a separate process, we have the opportunity to restrict its access to system resources via sandboxing. For example, we can ensure that the renderer's only access to the network is via its parent browser process. Likewise, we can restrict its access to the filesystem using the host operating system's built-in permissions.  
In addition to restricting the renderer's access to the filesystem and network, we can also place limitations on its access to the user's display and related objects. We run each render process on a separate Windows "Desktop" which is not visible to the user. This prevents a compromised renderer from opening new windows or capturing keystrokes.”[[3]](#footnote-2)*

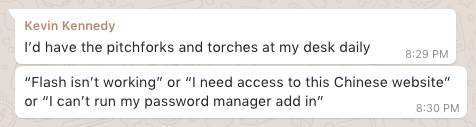
Chromium’s architecture mimics the type of isolation that processes are given when running in a modern Operating System. This type of architecture is ideal for creating and enforcing custom security policies. If we can actively control the policies used to make decisions in the browsers and dynamically grant permissions to processes running inside tabs, we can do a better job at protecting the users from harming themselves or the organization without impeding access to legitimate sites or slowing down performance.

Google built Chromium with security in mind, however, the policy management features are a little outdated, slow, and relatively weak. Google pointed out this weakness in an [online comic](https://www.google.com/googlebooks/chrome/big_32.html) they used to announce the Chromium project in 2008. The security API is a public, however, and they did this with the idea that the community would improve on the design. Our idea is to create an **Active Policy Agent AI Agent** in the Broker module, one ideally updated by private blockchain like the one in BETA at Oasis Labs. The policies in the browser could be updated from a reliable source in real time while the AI module makes decisions about safety based on derived variants of the threats before they are found by threat researchers.



[[4]](#footnote-3)

The Active Policy AI Agent operates under the basic principles of [Threat Hunting](https://en.wikipedia.org/wiki/Cyber_threat_hunting), is updated by a trusted source using Private BlockChain [Smart Contracts](https://en.wikipedia.org/wiki/Smart_contract) and relay Threat Intelligence using BlockChain back to the security team for analysis, alerts, and actions. When I described some early ideas of this system to a classmate their initial response was :



With the Active Policy AI Agent and a system that relays the behavior of users back to the Security Team there would be no need for “pitchforks and torches.” Making decisions about a particular Chinese website, or a certain version of the Flash Player based on real-time threat intelligence and the actual processes running in the tabbed sandbox of a user’s browser eliminates the need for most analog “permission granting.”

Using Oasis Labs Testnet and a fork of the Chromium Project, I’ve started development of an Active Policy Agent AI.

###### *Passwords*

The Active Policy AI Agent doesn’t just need to scan for malicious code or suspicious websites, it can be configured to ensure that passwords are strong and memorable. The best passwords are long, include several different types of characters and most importantly **can be remembered**. Forcing users to pick long passwords that they can’t remember leads to a much worse CyberSecurity no-no, saving them in a file called “passwords.” Quoting myself:

*“We all remember lyrics from songs, or lines from movies, or famous quotes. If we intersperse these with numbers and special characters we can make a pretty strong password. For instance, many people know this quote: "Hello, my name is Inigo Montoya, you killed my father, prepare to die." What if we converted that to HmniIMykmfPTD747#! ... Here I just used the first letter of each word in the phrase and capitalized the Hello, Inigo Montoya and Prepare To Die, then added 747, hash, bang.”[[5]](#footnote-4)*

The Active Policy AI Agent will sense when the user is picking or using a password and can provide prompts that help the user create a strong password using the method above, almost like an Alexa service.

**Policy Agent**: *“Looks like you are choosing a password, would you like some help?”*

**User**: Sure.

**Policy Agent**: *“Great, think of a line from your favorite movie or song. Don’t write it down, just think of it. Choose something you know like the back of your hand”*

**User**: Ok, got it.

**Policy Agent**: *“Ok, now use the first letter of each word in that phrase to create your password, capitalizing words like names or key points … then add a small number and a special character that you can remember. For example: ‘Show me the Money!’ -Jerry Maguire … would be* ‘**SmtM!JM747#!**’ ”

## Access Control

##### High-Level Abstract Concepts

We have many different ways of calculating trust and safety outside of computers. In the real world, we rely on relationships to measure trust. We know that a title alone does not entitle someone to access. For example, a child’s mother has authority over them as opposed to just any random person with the title mother. The title “mother” alone does not grant access. However, a mother may pass authority to another family member or even a stranger, who will consequently have authority over the child, even though they don’t have the title. This type of permission grant is nuanced, but we can distill the transfer of permission to a few key points. The subject ( the mother ) is the creator of the object ( the child ) and therefore inherits the natural authority to grant permission. The mother also has a relationship with the child, a history of interactions. This is relevant because a mother with no history of caring for their child would not hold the same authority ( in the case the mother had given the child up for adoption ). Lastly, we can assume the mother has some interest in the security of the child, so the mother is checking any person that she grants permission to for some level of trust. To summarize, we need to check three things when granting permission, hierarchy ( is the subject a parent to the object? ), relationship ( is the subject connected to the object? ) and trust ( can the subject trust the grantee of the permissions with the object, and what is the range of that trust ).

There are no actual children in a file system, but let’s examine the analogy and see if the ideas of hierarchy, relationship, and trust hold true. A user creates a file ( the user is the parent of the file ), the user frequently updates the file ( a history of interactions ) and then the user may assign some importance to the file ( shows a concern for the file’s security ). That last step rarely happens. We have too many “children” in the digital space and rarely make time to assign a level of importance to each document, picture or file we make. And it follows since there is rarely detailed categorization of security for each document, file, and picture, they all default to the same level of security, which is usually a function of referencing predefined access list. Furthermore, it is rare that a user who is granted permission to a file has time to go through the same level of vetting that a mother would conduct with a potential babysitter. So, we can conclude that habits surrounding file creation and the time associated with proper vetting may prevent us from treating our files with the same level of care a mother would give their children. But, what if the process of categorization and vetting could be automated with technology?

##### Access Calculations

Role-Based Access Controls, Attribute Based Access Controls, and Lattice-Based Access Controls all give us some notion of control over an aspect of systems that users interact with.

In **RBAC** hierarchy is represented by the sets U, R, P, S (users, roles, permissions, and sessions, respectively) and defined by these base formulas.  
• UA ⊆ U × R (user assignment)  
• PA ⊆ R × P (permission assignment)  
• RH ⊆ R × R is a partial order called the role hierarchy or role dominance relation written as ≤.[[6]](#footnote-5)

**ABAC** is very similar but switches the concept of roles with attributes.

**LBAC** popular among engineering who create REST Services and API’s describes security “in terms of the lattice (a partial order set) where each object and subject have a greatest lower bound (meet) and least upper bound (join) of access rights.” [[7]](#footnote-6)

**By using machine learning to perform curation on data objects** which feeds a dynamic policy engine, a document with sensitive conversations between one nuclear scientist and military intelligence about HEARTBEAT, can be hidden from the other nuclear scientist based on the content in the document not on a policy attached to a folder, a predetermined uncommon attribute or a complex set of security bounds. The curation engine recognizes the conversation as privileged, prompts the user for guidance, and feeds the curation into a policy engine that determines access based on the relation of the subject to the document. The rule created in the engine is not transitive to any nuclear scientist with the same clearance, but only those that have a relationship with the topic and a history in the conversation. In a sense the control access threats each piece of content like a sensitive social media post, allowing the user to fine-tune the audience and usage, prompting them with curation ques based on similar bits of content. Here is a sample interaction from an email implementation of this new model:

( User is sending an email with sensitive information about HEARTBEAT to several recipients )

**Email Client:** “It looks like this email contains a link to confidential information about a program you have access to called: HEARTBEAT. Are you sure you want to include John Shirley, Richie Valentine, and Lionel Richie? There is no record they have authorization”

( User clicks an option to ADD John Shirley, and redact the mentions of HEARTBEAT for the other users )

**Email**  “Sent a request to add John Shirley to Patrick Stewart, information will be redacted until access it authorized”

This is not just good for filtering documents in a folder, but it can also be used to prevent unauthorized access. The system can restrict access when the user or creator behaves in a strange way or seems likely to be an imposter since the automated curation also includes a range of acceptable/normal access patterns. Again, creating this is RBAC would be next to impossible, but machine learning can render an analysis when the assertion of access or authority looks strange. When does the assertion of authority look strange? When a new subject, with a new status, asserts authority over an object it has never had a relationship with before. But not only that :

Asserting authority over an object after entering from a strange/unknown location

*( the Mom climbing in the window )*

Asserting a type of authority that has ultimate consequences for the object

*( the Mom tries to kill the child )*

*Giving authority to a new subject without consensus*

*( the Mom giving a child away to a stranger )*

**And notice what all of these examples have in common.**  *They all involve a risk to the object aka the child.* If we ask a smart question about the trust the child object requires we might be able to understand the necessary controls required for access and what that means.

##### Scoring Trust by First Measuring Distrust

In this new access control system, we will measure trust inversely by measuring distrust. If distrust is 0 than the subject is completely trusted. If distrust is 100 then the subject is completely untrusted. With a lower distrust score, the subject is granted more freedom of usage. However, we maintain that a distrust score of 0, still should be calculated in real time base on activity and usage. A user signing in for the first time from a Russian IP address at 1 am Pacific Standard Time on a Sunday my now violate the login access policy, but may not have enough trust to view documents with highly classified information that requires 0 distrust. This model allows the system to make intelligent decisions about access in real time without forcing all or nothing scenarios.

##### Creating Nuanced Layers of Access

**Each level of access involves many different layers.** Before you can read something you have to be able to see it. And usually seeing it involves the ability to look at its details. Seeing something and looking at details may seem similar but there are some nuances. Let’s exchange some terms. Is the object visible? Can you see the details? Visibility without details could let us know there is *something* there, without knowing what it is. Visibility with the ability to see details means knowing it is there and being able to see details like file name, file size, file type, etc. A subject could have visibility and details and still not be able to read. The same levels of access exist when trying to discern whether a subject copy an object or move it or destroy it.

##### Implementation

At NASA permission scheme/process are presently pretty antiquated. Gaining access to a resource, whether that resource is computational, physical or access related, involves submitting a formal request that must be checked by a sponsor and a supervisor. The request may also involve required training, security clearances and even US citizenship.

This cybersecurity plan introduces **Dynamic Relational Trust-Based Access Control ( DRTBAC ).**  This keeps track of the content that is produced by the user and aids in the curation of content for the use in a real-time policy engine. Much like the AI policy engine in the Chromium project this engine uses the analyzes the content and tries to come up with a policy based on the sensitivity of the content. In many ways, the NASA model has the same controls, but DRTBAC model automates the process and allows for the possibility of security within documents.

Implementation of DRTBAC will mean creating a special policy module in Jemison that include authentication methods and robust backend to train the machine learning model and retain policy data.

##### Assigning Roles

Assigning roles in DRTBAC is much different than previous models. DRTBAC is more interested in the projects and groups the user associated with instead of their hierarchy or title. Once a user is assigned to a group they relative access to documents with users that share their security clearance and work descriptions. Continuing with the pair of Nuclear Scientist, once Richard Sherman gets his NASA ID and laptop he still doesn’t have access to anything outside of his machine. To get access he must be associated with a group working on a project and as user start interacting with Richard and sharing content access is granted by the owner of the content. In some ways, this mimics the Discretionary Access Control Model ( the model currently used at NASA ), with the key difference that the system is curating and monitoring the way Richard interacts with co-workers and is generating suggestions for access as he works, verifying this access with the owners of the system against a record of his security clearance.

Works Cited

Caltagirone, Sergio, et al. “The Diamond Model of Intrusion Analysis.” 2013, www.dtic.mil/dtic/tr/fulltext/u2/a586960.pdf.

GoogleDevelopers. “Chromium's Multi-Process Architecture.” *YouTube*, YouTube, 29 Apr. 2009, www.youtube.com/watch?v=A0Z0ybTCHKs.

“Endpoint Security.” *Wikipedia*, Wikimedia Foundation, 4 Oct. 2018, en.wikipedia.org/wiki/Endpoint\_security#Corporate\_network\_security.

Goodrich, Michael T., and Roberto Tamassia. *Introduction to Computer Security*. Pearson, 2014.

Kassner, Michael. “Cyber Threat Hunting: How This Vulnerability Detection Strategy Gives Analysts an Edge.” *TechRepublic*, TechRepublic, 29 Apr. 2016, www.techrepublic.com/article/cyber-threat-hunting-why-this-active-strategy-gives-analysts-an-edge/.

“Multi-Process Architecture.” *The Chromium Projects*, Google, www.chromium.org/developers/design-documents/multi-process-architecture.

“The Broker Process.” *Sandbox*, Google, chromium.googlesource.com/chromium/src/ /master/docs/design/sandbox.md#the-broker-process.

“The Target Process.” *Sandbox*, Google, chromium.googlesource.com/chromium/src/ /master/docs/design/sandbox.md#the-target-process.

1. “Endpoint Security.” *Wikipedia*, Wikimedia Foundation, 4 Oct. 2018, en.wikipedia.org/wiki/Endpoint\_security#Corporate\_network\_security. [↑](#footnote-ref-0)
2. GoogleDevelopers. “Chromium's Multi-Process Architecture.” *YouTube*, YouTube, 29 Apr. 2009, www.youtube.com/watch?v=A0Z0ybTCHKs. [↑](#footnote-ref-1)
3. “Multi-Process Architecture.” *The Chromium Projects*, Google, www.chromium.org/developers/design-documents/multi-process-architecture. [↑](#footnote-ref-2)
4. “The Target Process.” *Sandbox*, Google, chromium.googlesource.com/chromium/src/ /master/docs/design/sandbox.md#the-target-process. [↑](#footnote-ref-3)
5. Davis, Brian. “Post-Work: Discussion - Recovering After a Password Breach!” *Post-Work: Discussion - Recovering After a Password Breach!*, Brian Russel Davis, 8 Oct. 2018, canvas.brown.edu/courses/1076326/discussion\_topics/3564539. [↑](#footnote-ref-4)
6. Sohr, Karsten & Drouineaud, Michael & Ahn, G.-J & Gogolla, Martin. (2008). Analyzing and Managing Role-Based Access Control Policies. Knowledge and Data Engineering, IEEE Transactions on. 20. 924-939. 10.1109/TKDE.2008.28. [↑](#footnote-ref-5)
7. “Lattice-Based Access Control.” *Wikipedia*, Wikimedia Foundation, 14 May 2018, en.wikipedia.org/wiki/Lattice-based\_access\_control. [↑](#footnote-ref-6)