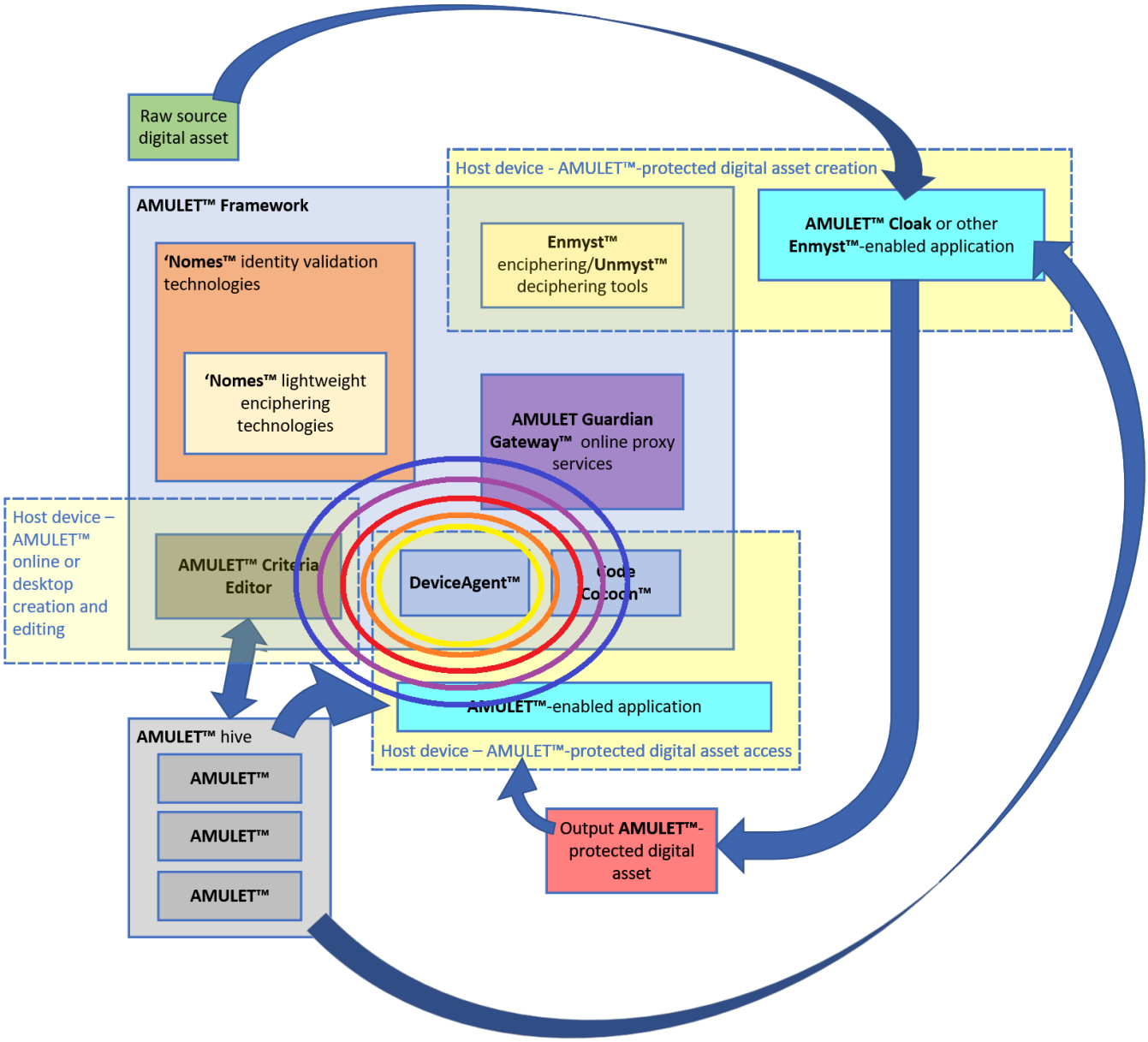


AMULET™ Framework's workhorse: The DeviceAgent™

Version 1.0

F. Scott Deaver
CTO, Certitude Digital, Inc.
September 30, 2020

Table of contents

[The technologies described here are exclusive and proprietary to F. Scott Deaver and Certitude Digital, Inc. They are represented by issued patents, patents pending, copyrights, trademarks and underlying trade secrets protected by intellectual property law. We offer them here in the spirit of transparency and education as a public service to enhance awareness with the intention to help prevent crime and abuse, and with the expectation that our rights will be respected. We intend to enforce those rights as required.]

The AMULET™ Framework and DeviceAgent™

The AMULET™ Framework is a system of independent, Internet, and/or enterprise components that impenetrably and unfailingly provides lock-down security for a protected digital asset (a file or block of memory) and all copies of it, no matter where they go, how they get there, or what they do, in strict accordance with the wishes of the digital asset's content provider(s) and/or owner(s). No other system, service, or application on other can do the same - the ideas, methods, and tools that accomplish these things are uniquely proprietary unto myself and Certitude Digital.

That having been said, it is our intention that as many people as possible derive benefit from our technologies, irrespective of whether they pay anything for them or not (we provide significant value-add in optional products and services that we can profit fairly, while ensuring that it not only costs the vast majority of consumers of protected digital asset absolutely nothing, they also benefit is a wide variety of ways beyond just the security they achieve.

This system as a whole is described in lay terms in our document entitled "The AMULET™ Framework technology map", available from the Certitude Digital website at https://www.certitudedigital.com/public_docs/articlesdownload/AMULETFrameworkTechnologyMap.pdf.

The DeviceAgent™'s role

Of all of the components of the AMULET™ Framework, the DeviceAgent™ may very well be the most important. Each and every AMULET™ device is installed with one, and only one, instance of the DeviceAgent™, which will launch whenever the device is started. The DeviceAgent™ performs many functions, but its most critical role is to decipher (our unique form of that process is called "Unmyst™") data from a protected digital asset into a secured space (called a Code Cocoon™) where an authorized AMULET™-enabled application or services can then receive the enciphered contents streamed to them directly on a time-limited basis.

Before we get into the more-technical presentation, this is the description of the DeviceAgent™ borrowed from our "The AMULET™ Framework technology map" article:

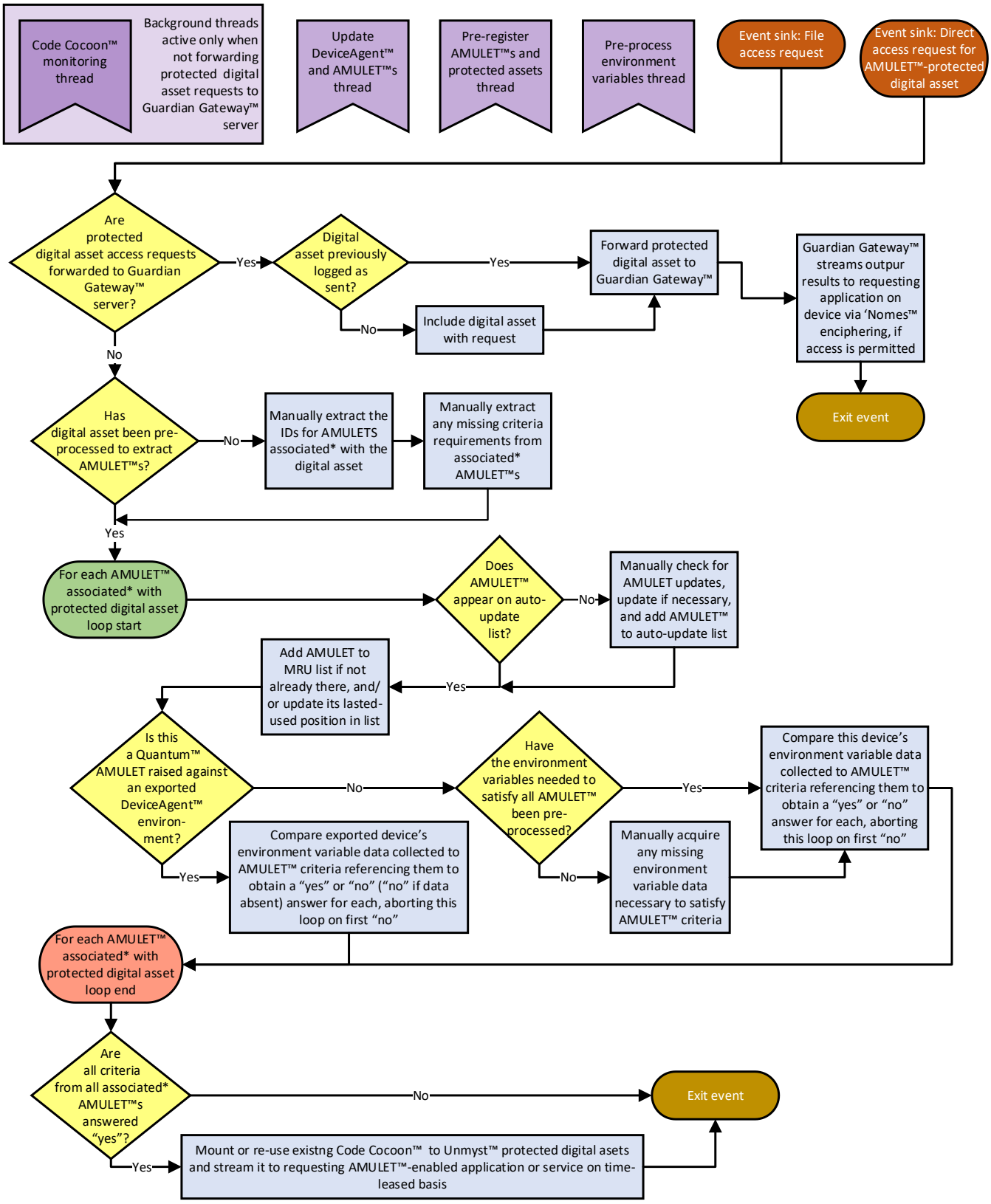
> "This is the component, one to each device and usually loaded at the startup of the device, which does all of the heavy lifting when it comes to determining whether access is to be permitted to an AMULET™-protected digital asset. There are two forms of the DeviceAgent™ - in the standard form, the DeviceAgent™ ensures version currency of the DeviceAgent™ itself as well as any AMULET™s it comes in contact with when online, gathers device environment information in the background that may be needed to satisfy AMULET™ criteria, prequalifies AMULET™ criteria for AMULET™s that have been registered with it for impending use. It then reacts to protected digital asset access requests from AMULET™-enabled applications and services, verifying that the current device environment satisfies all of the criteria in AMULET™s associated with the digital asset. If so, digital asset is temporarily Unmysted™ into a secure Code Cocoon™ isolation area, from which it is then streamed to the requesting authorized app or service in segments on a time-leased basis.

> "In the AMULET Guardian Gateway™-dependent form of the DeviceAgent™, intended for device with slower CPUs, less memory, less storage capacity, or multi-tasking limitations, the DeviceAgent™ collects the device's environment as before, but when an AMULET™-protected digital asset access request arrives, the environment and the request are instead shipped to an AMULET Guardian Gateway™ server via 'Nomes™ lightweight useridentity certified enciphering. The AMULET Guardian Gateway™ server analyzes the AMULET™s associated with the protected digital asset against the enciphered environment data shipped up, and if appropriate Unmysts™ the digital asset into its own secure Code Cocoon™ isolation area, streaming the Unmysted™ results back to the original requesting AMULET™-enabled application or service on the device on a time-leased basis using 'Nomes™ lightweight enciphering."

Our goal with this document is to present a more detailed description of how the DeviceAgent™ functions, as a means to demonstrate the engineering behind the AMULET™ Framework system and the DeviceAgent™'s relationships with other important system components.

I've elected to do this primarily with flowcharts, which most people who deal with digital systems on any technical level can easily understand and follow. I will include some brief textual comments following each, as appropriate. With that, off we go!

DeviceAgent™ process flow



* Relationships between AMULET™s associated with a protected digital asset can be peer-to-peer, or wrapped like layers of an onion (with implied precedence between layers), or a combination of both

The DeviceAgent™ is comprised of a number of background threads along with at least two event handlers mounted in a primary process thread a primary. Other processes and threads exist - for example, external processes monitor the main process thread (and the main process thread monitors those processes in return) to detect and mitigate any external gaming attempts using proprietary techniques we invented). These are not shown, for purposes of clarity and to avoid tipping our hand to potential hackers. Also not shown are various direct entry points AMULET™-enabled applications can leverage via our licensed Application Programming Interface (API) and Software Development Kits (SDKs).

AMULET Guardian Gateway™ influences

The DeviceAgent™'s behaviors and services vary in accordance with whether the hosting device is forwarding some of its responsibilities to an AMULET Guardian Gateway™ proxy server, or whether the host device is itself a Guardian Gateway™ server. An AMULET Guardian Gateway™ proxy server is a patent-protected suite of a software and services which can do the heavy lifting with respect to Unmysting™ and streaming results from Code Cocoons™ to the requesting AMULET™-enabled applications when a host device is online and has too little CPU power, memory, speed, or multi-threading capability to do the work itself.

In that case where the hosting device is forwarding some of its responsibilities to an AMULET Guardian Gateway™ proxy server, there will be no Code Cocoon™s to monitor with a background thread. When that happens, the host will still need to collect local environment data relevant to any AMULET™s associated, but some of the duties of those background processing threads as well as the main event processing thread will be reduced.

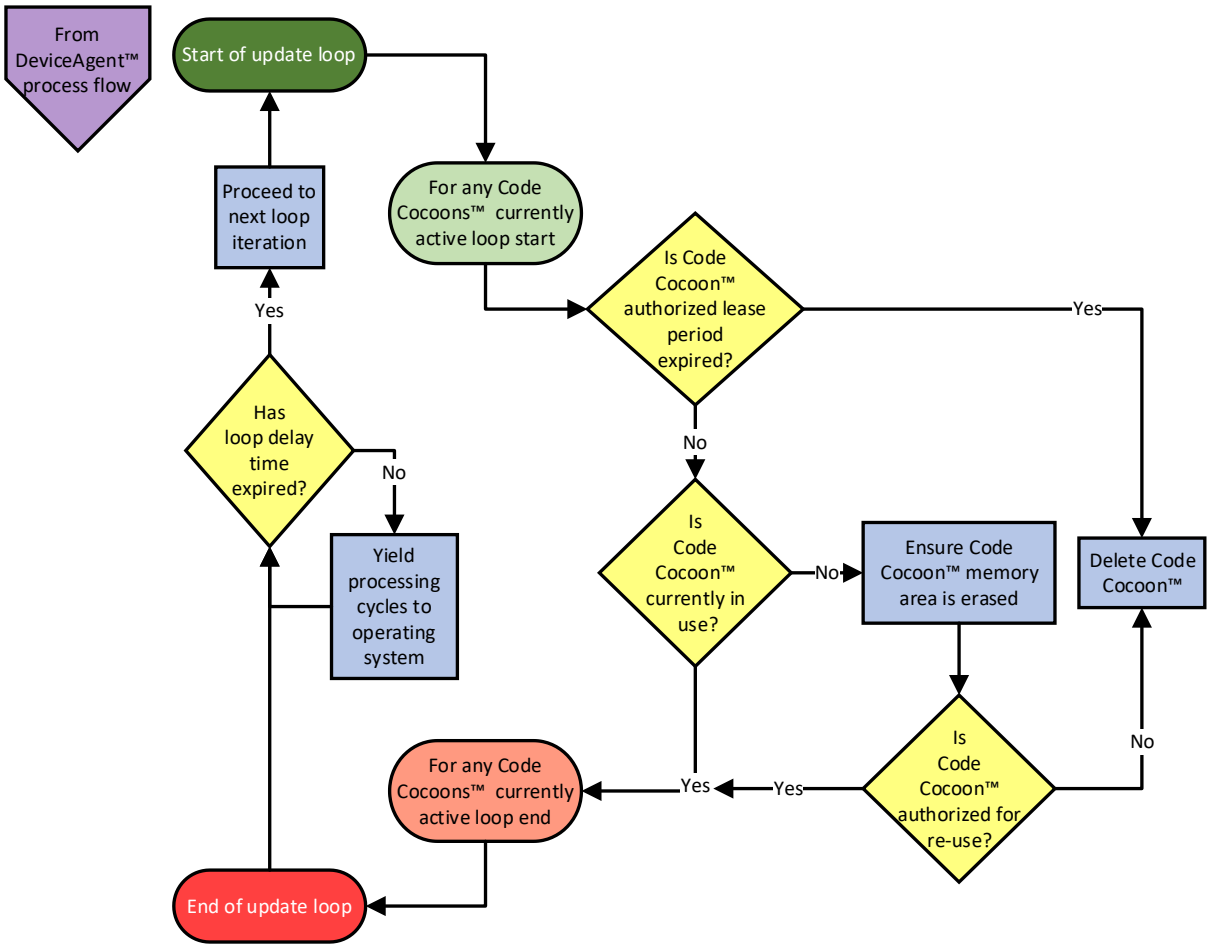
AMULET Quantum™ influences

Another factor that can influence some of the work to be done at the host device is whether or not the device is publishing its environment to an AMULET Quantum™ shared environment server, or whether the host device itself is an AMULET Quantum™ shared environment server. An AMULET Quantum™ shared environment server is part of a patented system that allows an AMULET™-protected digital asset's AMULET™ criteria to be assessed, not against the current host device's environment (which might be nothing more than a generic database server or Kubernetes-managed virtual machine or container), but instead assessed against an exported remote host device's real-time, runtime exported environment data. For example, your SSN or other private date residing in the cell of a faraway database you know nothing about can be dynamically protected based on your activities on your personal device, without those activities being revealed to anyone (hence "Quantum™" - quantum entanglement is the means in of quantum mechanics in physics by which the states of a particle, or "qbit" in computer parlance, at one location are non-independent from the states of a qbit in another place entirely).

AMULET Quantum™ capabilities, when enabled, can potentially add the task of exporting a device's environment data to a shared environment server to a background process thread, or possibly cahnge the main even processing thread task from comparing an AMULET™s criteria to exported environment data rather than host device environment data.

In light of these influences, the flowchart above should be reasonably self-explanatory.

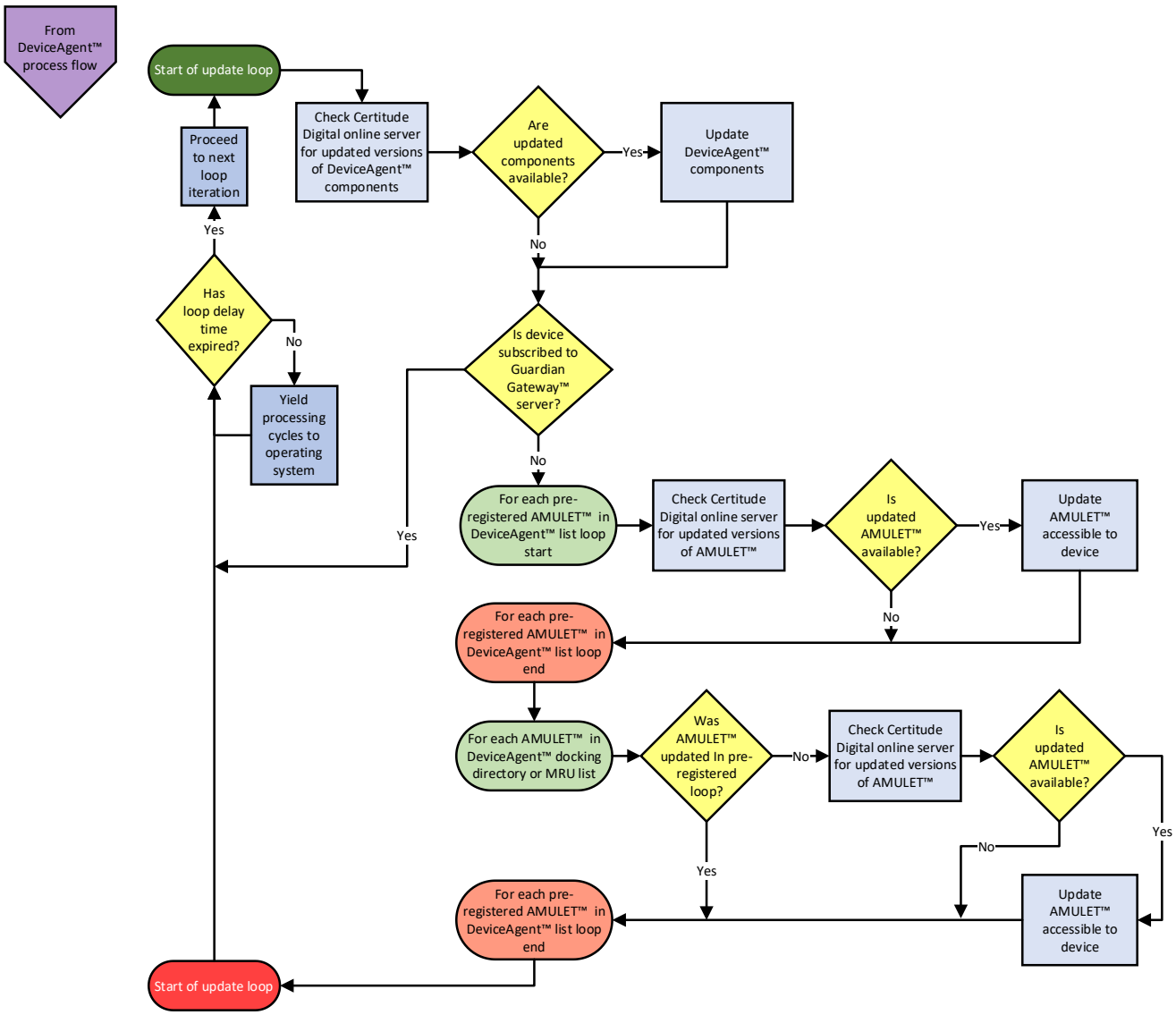
Code Cocoon™ monitoring process



For host devices that are not reliant upon an AMULET Guardian Gateway™ proxy server, this background process thread ensures that no Code Cocoon™s get accidentally abandoned (due to an AMULET™-enabled third-party application failure or abort, for instance) or linger beyond their lease or usefulness, or are being subjected to external manipulation. A Code Cocoon™ is the secure temporary disposable enciphered isolation environment to which AMULET™-protected digital assets are briefly Unmysted™ and then streamed on a time-leased basis to a requesting AMULET™-enabled application or service, after which the Code Cocoon™ and any Unmysted™ digital asset remnants within it are destroyed.

The flowchart above describes the primary functions of the Code Cocoon™ monitoring background processing thread.

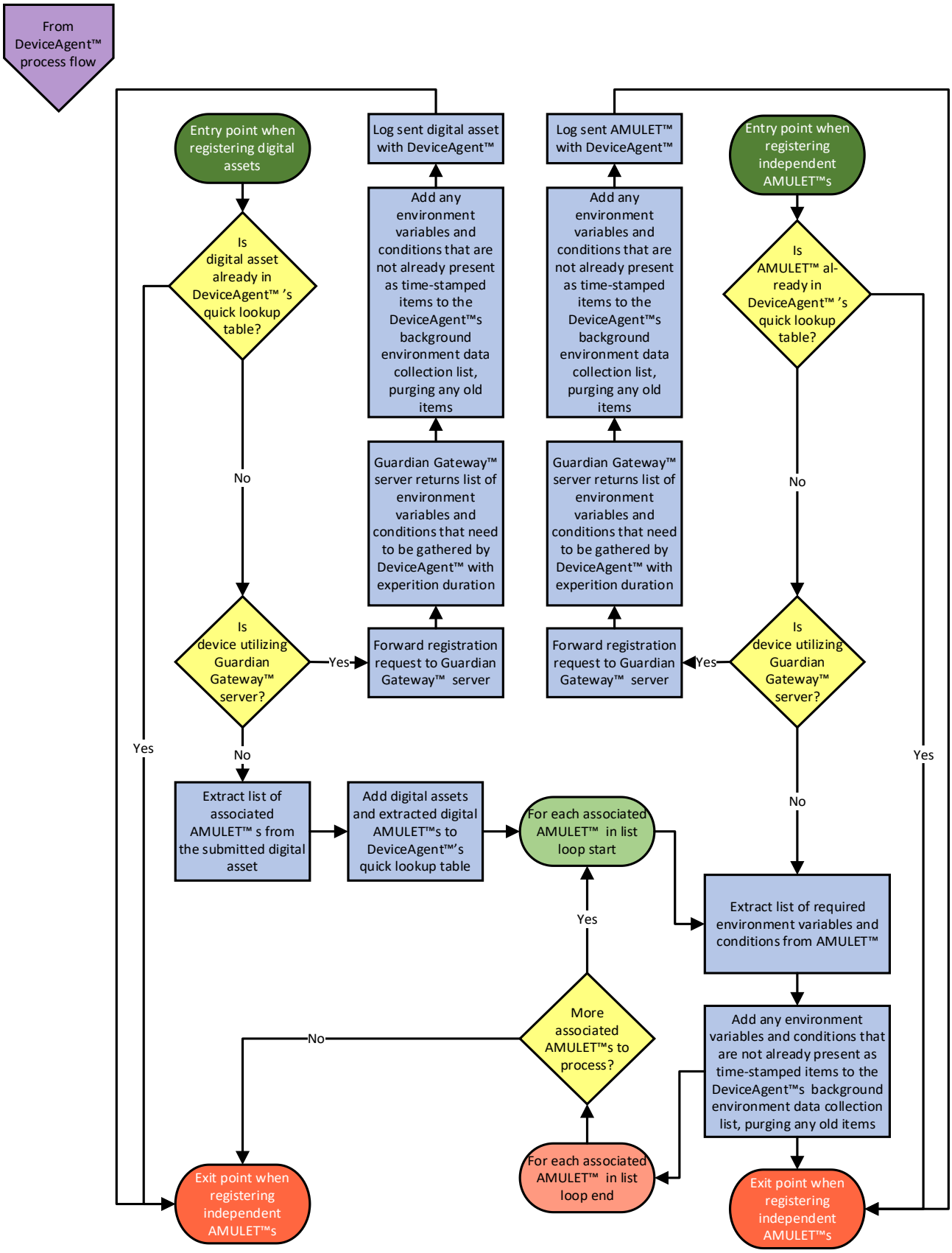
Updating the AMULET DeviceAgent™ and docked AMULET™s when online



Whenever any AMULET™-enabled host device is online, this background process thread keeps any AMULET™ Framework system components and AMULET™s within its purview updated.

The flowchart above describes the primary functions of the AMULET™ Framework system and AMULET™s online update background processing thread.

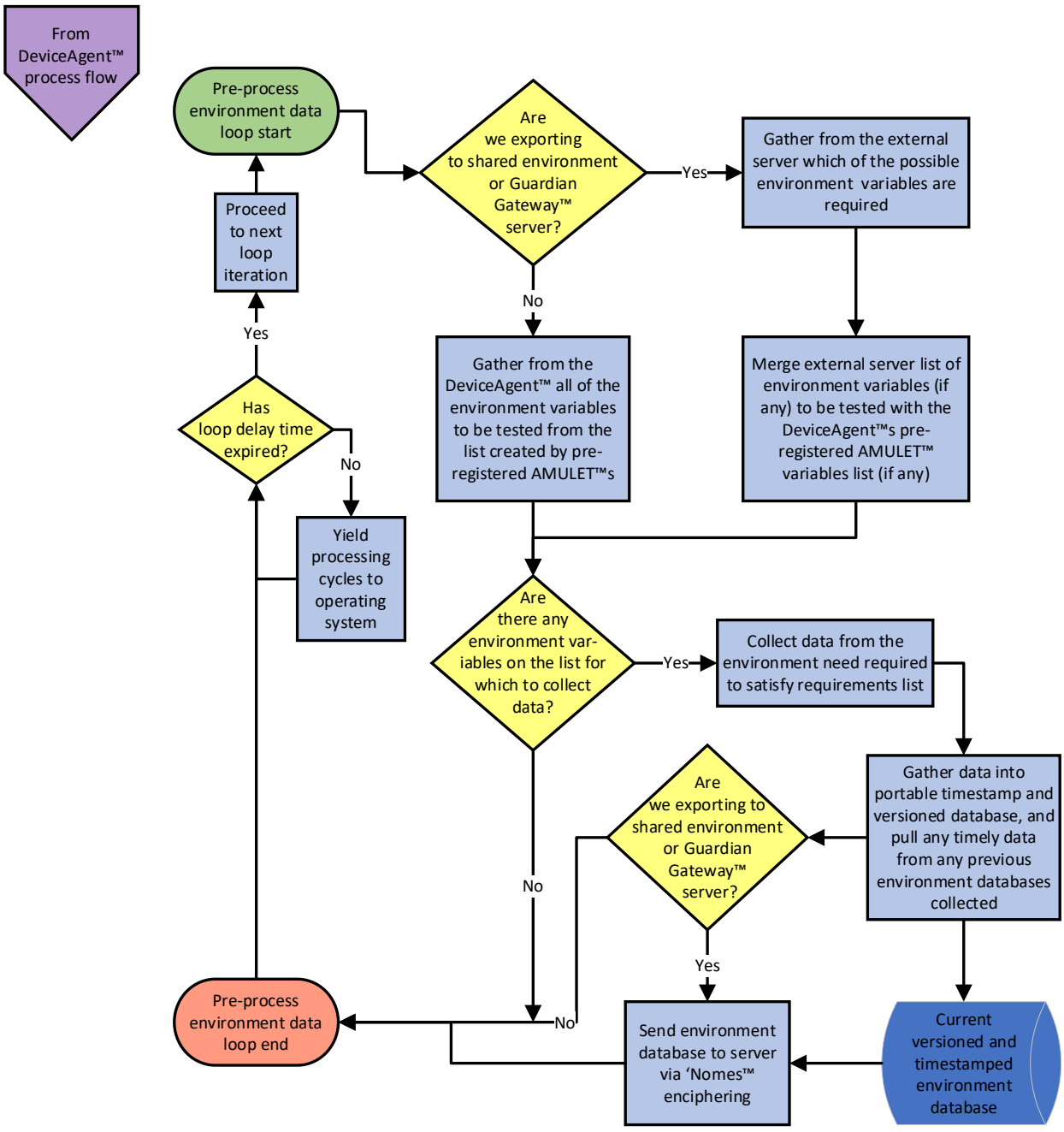
Pre-register AMULET™-protected digital assets and independent AMULET™s



Among the many unique efficiencies in the AMULET™ Framework system is that both AMULET™- protected digital assets, as well as individual independent AMULET™s, can be registered with the DeviceAgent™ prior to use by an AMULET™-enabled application or service, through a licensed API or SDK. When pre-registered, the crtieria in the pre-registered AMULET™s can be pre-assessed against the host device's environment data in another background thread prior to actual use. When pre-assessed, subsequent access requests for AMULET™-protected digital assets can be answered instantaneously.

The flowchart above describes the primary functions of the AMULET™-protected digital assets and independent AMULET™s pre-registration update background processing thread.

Pre-processing environment variables loop



*When supporting both external shared environment and Guardian Gateway™ servers, the Guardian Gateway™ server will be re-directed to retrieve its environment data from the external shared environment. Devices will at most share their environment data with only one server.

Another of the many unique efficiencies in the AMULET™ Framework system is that the environment variables needed to satisfy pre-registered AMULET™-criteria can be collected in advance (and optionally forwarded to an AMULET Guardian Gateway™ proxy server or AMULET Quantum™ server). When pre-collected the crtieria in the pre-registered AMULET™s can be pre-assessed against the host device's environment data in another

background thread prior to actual use. When pre-assessed, subsequent access requests for AMULET™-protected digital assets can be answered instantaneously.

The flowchart above describes the primary functions of the host device environment data advance collection background processing thread.

Pre-comparison of AMULET™ criteria to environment variables loop

There is, as we have referenced and you probably would have surmised anyway, another background processing thread which pre-compares registered AMULET™ criteria to collected environment date so that the answers to all known AMULET™ criteria are already tabled when an AMULET™-protected digital assets request is made, permitting instantaneous access decisions. That's a pretty trivial thing to do on a computer, so you can safely assume we have the means to do that.

But if you don’t already know, doing things the traditional way when they could be improved (and not coincidentally, made more secure at the same time) is never good enough for us. So, we have invented a machine-language parallelism compiler language between the cores of a multi-core processor on a host device that happens to be really good, and really fast, at these kinds of comparisons… and also (conveniently for us) inscrutable to external observers. We haven’t the resources at the moment to properly protect these innovations (and they have value beyond cybersecurity) as formal intellectual property, so we are not able to expose them here for this document.

For the moment, you can simply assume a big loop that merely compares each unique AMULET criteria in turn to its corresponding environment data, and tables the "yes/no" answer and timestamp for quick look-up later, which ordinarily would be plenty "good enough". Look to later revisions of this document posted online at the Certitude Digital website for descriptions of the actual method we use once we are permitted to do so.