drip S. Card
Internet-Draft A. Wiethuechter
Intended status: Informational AX Enterprize
Expires: 26 August 2021 R. Moskowitz
HTT Consulting
S. Zhao (Editor)
Tencent
A. Gurtov
Linkoeping University
22 February 2021

Drone Remote Identification Protocol (DRIP) Architecture
draft-ietf-drip-arch-11

Abstract

This document defines an architecture for protocols and services to support Unmanned Aircraft System Remote Identification and tracking (UAS RID), plus RID-related communications, including required architectural building blocks and their interfaces.

Table of Contents

1. Introduction

This document describes an architecture for protocols and services to support Unmanned Aircraft System Remote Identification and tracking (UAS RID), plus RID-related communications, conforming to proposed and final regulations plus external technical standards, satisfying the requirements listed in the companion requirements document [I-D.ietf-drip-reqs].

1.1. Overview UAS Remote ID (RID) and RID Standardization

UAS Remote Identification (RID) is an application enabler for a UAS to be identified by UTM/USS or third parties entities such as law enforcement. Many safety and other considerations dictate that UAS be remotely identifiable. CAAs worldwide are mandating UAS RID. The European Union Aviation Safety Agency (EASA) has published [Delegated] and [Implementing] Regulations.

CAAs currently promulgate performance-based regulations that do not specify techniques, but rather cite industry consensus technical standards as acceptable means of compliance.

FAA

The FAA published a Notice of Proposed Rule Making [NPRM] in 2019 and whereafter published the Final Rule [FAA_RID] in 2021.

ASTM

ASTM International, Technical Committee F38 (UAS), Subcommittee F38.02 (Aircraft Operations), Work Item WK65041, developed the ASTM [F3411-19] Standard Specification for Remote ID and Tracking.

ASTM defines one set of RID information and two means, MAC-layer broadcast and IP-layer network, of communicating it. If a UAS uses both communication methods, the same information must be provided via both means. The [F3411-19] is cited by FAA in its RID final rule [FAA_RID] as "a potential means of compliance" to a Remote ID rule.

3GPP

With release 16, 3GPP completed the UAS RID requirement study [TS-22.825] and proposed use cases in the mobile network and the services that can be offered based on RID. Release 17 specification works on enhanced UAS service requirements and provides the protocol and application architecture support which is applicable for both 4G and 5G network.

1.2. Overview of Types of UAS Remote ID

1.2.1. Broadcast RID

A set of RID messages are defined for direct, one-way, broadcast transmissions from the UA over Bluetooth or Wi-Fi. These are currently defined as MAC-Layer messages. Internet (or other Wide Area Network) connectivity is only needed for UAS registry information lookup by Observers using the locally directly received UAS RID as a key. Broadcast RID should be functionally usable in situations with no Internet connectivity.

The Broadcast RID is illustrated in Figure 1 below.

```
                  x x  UA
                 xxxxx
                   |
                   |
                   |     app messages directly over
                   |     one-way RF data link (no IP)
                   |
                   |
                   +
                   x
                 xxxxx
                   x
                   x
                   x x   Observer's device (e.g. smartphone)
                 x   x
```

Figure 1

With Broadcast RID, an Observer is limited to their radio "visible" airspace for UAS awareness and information. With Internet queries using harvested RID (see Section 6), the Observer may gain more information about those visible UAS.

1.2.2. Network RID

A RID data dictionary and data flow for Network RID are defined in [F3411-19]. This data flow is from a UAS via unspecified means (but at least in part over the Internet) to a Network Remote ID Service Provider (Net-RID SP). These Net-RID SPs provide the RID data to Network Remote ID Display Providers (Net-RID DP). It is the Net-RID DP that responds to queries from Network Remote ID Observers (expected typically, but not specified exclusively, to be web-based) specifying airspace volumes of interest. Network RID depends upon connectivity, in several segments, via the Internet, from the UAS to the Observer.

The Network RID is illustrated in Figure 2 below:

```
               x x  UA
               xxxxx       ********************
                |   \    *                ------*---+------------+
                |    \   *              /       *  | NET_RID_SP |
                |     \  * ------------/    +---*--+------------+
                | RF   \ */                 |   *
                |        *      INTERNET    |   *  +------------+
                |       /*                  +---*--| NET_RID_DP |
                |      / *                  +---*--+------------+
                +     /   *                 |   *
                 x   /     *****************|***      x
               xxxxx                        |       xxxxx
                 x                          +-------  x
                 x                                    x
                x x   Operator (GCS)      Observer   x x
               x   x                                x   x
```

Figure 2

Command and Control (C2) must flow from the GCS to the UA via some path (ex. a direct RF link, but with increasing BVLOS operations expected often to be wireless links at either end with the Internet between). For all but the simplest hobby aircraft, telemetry (at least position and heading) flows from the UA to the GCS via some path (typically the reverse of the C2 path). Thus RID information pertaining to both the GCS and the UA can be sent by whichever has Internet connectivity to the Net-RID SP (typically the USS managing the UAS operation). The Net-RID SP forwards RID information via the Internet to subscribed Net-RID DP (typically other USS). Subscribed Net-RID DP forward RID information via the Internet to subscribed Observer devices. Regulations require and [F3411-19] describes RID data elements end-to-end. [F3411-19] prescribes the protocol only among Net-RID SP, Net-RID DP, and the Discovery and Synchronization Service (DSS).

> Informative note: Neither link layer protocols nor the use of links (e.g., the link often existing between the GCS and the UA) for any purpose other than carriage of RID information is in the scope of [F3411-19] Network RID..

1.3. Overview of USS Interoperability

Each UAS is registered to at least one USS. With Net-RID, there is direct communication between the UAS and its USS. With Broadcast-RID, the UAS Operator has either pre-filed a 4D space volume for USS operational knowledge and/or Observers can be providing information about observed UA to a USS. USS exchange information via a Discovery and Synchronization Service (DSS) so all USS collectively have knowledge about all activities in a 4D airspace.

The interactions among Observer, UA, and USS are shown in Figure 3.

```
            +----------+
            | Observer |
            +----------+
           /            \
          /              \
   +-----+                +-----+
   | UA1 |                | UA2 |
   +-----+                +-----+
          \              /
           \            /
            +----------+
            | Internet |
            +----------+
           /            \
          /              \
    +-------+           +-------+
    | USS-1 | <-------> | USS-2 |
    +-------+           +-------+
             \         /
              \       /
              +------+
              |  DSS |
              +------+
```

Figure 3

1.4. Overview of DRIP Architecture

The requirements document [I-D.ietf-drip-reqs] also provides an extended introduction to the problem space, use cases, etc. Only a brief summary of that introduction will be restated here as context, with reference to the general UAS RID usage scenarios shown in Figure 4 below.

```
         General      x                           x     Public
         Public     xxxxx                       xxxxx   Safety
         Observer     x                           x     Observer
                      x                           x
                     x x ---------+  +---------- x x
                    x   x         |  |          x   x
                                  |  |
            UA1 x x               |  |  +------------ x x UA2
               xxxxx              |  |  |            xxxxx
                  |               +  +  +              |
                  |            xxxxxxxxxx              |
                  |           x          x             |
                  +----------+x Internet x+------------+
       UA1        |           x          x             |       UA1
      Pilot     x |            xxxxxxxxxx              | x    Pilot
     Operator  xxxxx              + + +                xxxxx Operator
      GCS1      x                 | | |                  x    GCS2
                x                 | | |                  x
               x x                | | |                 x x
              x   x               | | |                x   x
                                  | | |
                +----------+      | | |       +----------+
                |          |------+ | +-------|          |
                | Public   |        |         | Private  |
                | Registry |     +-----+      | Registry |
                |          |     | DNS |      |          |
                +----------+     +-----+      +----------+
```

Figure 4

DRIP will enable leveraging existing Internet resources (standard protocols, services, infrastructure, and business models) to meet UAS RID and closely related needs. DRIP will specify how to apply IETF standards, complementing [F3411-19] and other external standards, to satisfy UAS RID requirements. DRIP will update existing and develop new protocol standards as needed to accomplish the foregoing.

This document will outline the UAS RID architecture into which DRIP must fit and the architecture for DRIP itself. This includes presenting the gaps between the CAAs' Concepts of Operations and [F3411-19] as it relates to the use of Internet technologies and UA direct RF communications. Issues include, but are not limited to:

- Design of trustworthy remote ID and trust in RID messages (Section 4)

- Mechanisms to leverage Domain Name System (DNS: [RFC1034]), Extensible Provisioning Protocol (EPP [RFC5731]) and Registration Data Access Protocol (RDAP) ([RFC7482]) to provide for private (Section 5.2) and public (Section 5.1) Information Registry.

- Harvesting broadcast remote ID messages for UTM inclusion (Section 6)

- Privacy in RID messages (PII protection) (Section 7)

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown above.

3. Definitions and Abbreviations

3.1. Additional Definitions

This document uses terms defined in [I-D.ietf-drip-reqs].

3.2. Abbreviations

ADS-B: Automatic Dependent Surveillance Broadcast

DSS: Discovery & Synchronization Service

EdDSA: Edwards-Curve Digital Signature Algorithm

GCS: Ground Control Station

HHIT: Hierarchical HIT Registries

HIP: Host Identity Protocol

HIT: Host Identity Tag

RID: Remote ID

Net-RID SP: Network RID Service Provider

Net-RID DP: Network RID Display Provider.

PII: Personally Identifiable Information

RF: Radio Frequency

SDSP: Supplemental Data Service Provider

UA: Unmanned Aircraft

UAS: Unmanned Aircraft System

USS: UAS Service Supplier

UTM: UAS Traffic Management

3.3. Claims, Assertions, Attestations, and Certificates

This section introduces the terms "Claims", "Assertions", "Attestations", and "Certificates" as used in DRIP.

This is due to the term "certificate" having significant technological and legal baggage associated with it, specifically around X.509 certificates. These types of certificates and Public Key Infrastructure invoke more legal and public policy considerations than probably any other electronic communication sector. It emerged as a governmental platform for trusted identity management and was pursued in intergovernmental bodies with links into treaty instruments.

Claims:

A claim in DRIP is a predicate (e.g., "X is Y", "X has property Y", and most importantly "X owns Y" or "X is owned by Y"). One basic use case of a claim is an entity using an HHIT as an identifier, e.g., a UAS using an HHIT as a UAS ID.

Assertions:

An assertion in DRIP is a set of claims. This definition is borrowed from JWT/CWT. An HHIT of itself can be seen as an assertion: a claim that the identifier is a handle to an asymmetric keypair owned by the entity, and a claim that the identifier is in the registry specified by the HID embedded in the identifier.

Attestations:

An attestation in DRIP is a signed assertion. The signer may be a claimant or a third party. Under DRIP this is normally used when an entity asserts a relationship with another entity, along with other information, and the asserting entity signs the assertion, thereby making it an attestation.

Certificates:

A certificate in DRIP is an attestation, strictly over identity information, signed by a third party.

4. HHIT for DRIP Entity Identifier

This section describes the basic requirements of a DRIP entity identifier per regulation constrains from ASTM [F3411-19] and explains the use of Hierarchical Host Identity Tags (HHITs) as self-asserting IPv6 addresses and thereby a trustable DRIP identifier for use as the UAS Remote ID. HHITs self-attest to the included explicit hierarchy that provides Registrar discovery for 3rd-party ID attestation.

4.1. UAS Remote Identifiers Problem Space

A DRIP entity identifier needs to be "Trustworthy". This means that within the framework of the RID messages, an Observer can establish that the DRIP identifier used does uniquely belong to the UAS. That the only way for any other UAS to assert this DRIP identifier would be to steal something from within the UAS. The DRIP identifier is self-generated by the UAS (either UA or GCS) and registered with the USS.

The data communication of using Broadcast RID faces extreme challenges due to the limitation of the demanding support for Bluetooth. The ASTM [F3411-19] defines the basic RID message which is expected to contain certain RID data and the Authentication message. The Basic RID message has a maximum payload of 25 bytes and the maximum size allocated by ASTM for the RID is 20 bytes and only 3 bytes are left unused. currently, the authentication maximum payload is defined to be 201 bytes.

Standard approaches like X.509 and PKI will not fit these constraints, even using the new EdDSA [RFC8032] algorithm. An example of a technology that will fit within these limitations is an enhancement of the Host Identity Tag (HIT) of HIPv2 [RFC7401] using Hierarchical HITs (HHITs) for UAS RID is outlined in HHIT based UAS RID [I-D.ietf-drip-rid]. As PKI with X.509 is being used in other systems with which UAS RID must interoperate (e.g. Discovery and Synchronization Service and any other communications involving USS) mappings between the more flexible but larger X.509 certificates and the HHIT-based structures must be devised.

By using the EdDSA HHIT suite, the self-attestations of the RID can be done in as little as 84 bytes. Third-party Certificates can be done in 200 bytes. An Observer would need Internet access to validate a self-attestations claim. A third-party Certificate can be validated via a small credential cache in a disconnected environment. This third-party Certificate is possible when the third-party also uses HHITs for its identity and the UA has the public key and the Certificate for that HHIT.

4.2. HIT as A Trustworthy DRIP Entity Identifier

For a Remote ID to be trustworthy in the Broadcast mode, it is better to have an asymmetric keypair for proof of ID ownership. The common method of using a key signing operation to assert ownership of an ID, does not guarantee name uniqueness. Any entity can sign an ID, claiming ownership. To mitigate spoofing risks, the ID needs to be cryptographically generated from the public key, in such a manner that it is statistically hard for an entity to create a public key that would generate (spoof) the ID. Thus the signing of such an ID becomes an a proof (verifiable attestation, versus mere claim) of ownership.

HITs are statistically unique through the cryptographic hash feature of second-preimage resistance. The cryptographically-bound addition of the Hierarchy and an HHIT registration process (e.g. based on Extensible Provisioning Protocol, [RFC5730]) provide complete, global HHIT uniqueness. This is in contrast to general IDs (e.g. a UUID or device serial number) as the subject in an X.509 certificate.

4.3. HHIT for DRIP Identifier Registration and Lookup

DRIP identifiers need a deterministic lookup mechanism that rapidly provides actionable information about the identified UA. The identifier itself needs to be the inquiry input into the lookup given the constraints imposed by some of the broadcast media. This can best be achieved by an Identifier registration hierarchy cryptographically embedded within the Identifier.

A HHIT itself consists of a registration hierarchy, the hashing crypto suite information, and the hash of these items along with the underlying public key. Additional information, e.g. an IPv6 prefix, can enhance the HHITs use beyond the basic Remote ID function (e.g use in HIP, [RFC7401]).

Therefore, a DRIP identifier can be represented as a HHIT. It can be self-generated by a UAS (either UA or GCS) and registered with the Private Information Registry (More details in Section 5.2) identified in its hierarchy fields. Each DRIP identifier represented as an HHIT can not be used more than once.

A DRIP identifier can be assigned to a UAS as a static HHIT by its manufacturer, such as a single HI and derived HHIT encoded as a hardware serial number per [CTA2063A]. Such a static HHIT can only be used to bind one-time use DRIP identifiers to the unique UA. Depending upon implementation, this may leave a HI private key in the possession of the manufacturer (more details in Section 8).

In another case, a UAS equipped for Broadcast RID can be provisioned not only with its HHIT but also with the HI public key from which the HHIT was derived and the corresponding private key, to enable message signature. A UAS equipped for Network RID can be provisioned likewise; the private key resides only in the ultimate source of Network RID messages (i.e. on the UA itself if the GCS is merely relaying rather than sourcing Network RID messages). Each Observer device can be provisioned either with public keys of the DRIP identifier root registries or certificates for subordinate registries.

The Operators, Private Information Registries as well as other UTM entities can possess UAS ID style HHITs. When present, such HHITs can be used with HIP to strongly mutually authenticate and optionally encrypt communications.

4.4. HHIT for DRIP Identifier Cryptographic

The only (known to the authors of this document at the time of its writing) extant fixed-length ID cryptographically derived from a public key are the Host Identity Tag [RFC7401], HITs, and Cryptographically Generated Addresses [RFC3972], CGAs. However, both HITs and CGAs lack registration/retrieval capability. HHIT, on the other hand, is capable of providing a cryptographic hashing function, along with a registration process to mitigate the probability of a hash collision (first registered, first allowed).

5. DRIP Identifier Registration and Registries

UAS registries can hold both public and private UAS information resulting from the DRIP identifier registration process. Given these different uses, and to improve scalability, security, and simplicity of administration, the public and private information can be stored in different registries. A DRIP identifier is amenable to handling as an Internet domain name (at an arbitrary level in the hierarchy). It also can be registered in at least a pseudo-domain (e.g. .ip6.arpa for reverse lookup), or as a sub-domain (for forward lookup). This section introduces the public and private information registries for DRIP identifiers.

5.1. Public Information Registry

5.1.1. Background

The public registry provides trustable information such as attestations of RID ownership and HDA registration. Optionally, pointers to the repositories for the HDA and RAA implicit in the RID can be included (e.g. for HDA and RAA HHIT|HI used in attestation signing operations). This public information will be principally used by Observers of Broadcast RID messages. Data on UAS that only use Network RID, is only available via an Observer's Net-RID DP that would tend to provide all public registry information directly. The Observer can visually "see" these UAS, but they are silent to the Observer; the Net-RID DP is the only source of information based on a query for an airspace volume.

5.1.2. Proposed Approach

A DRIP public information registry can respond to standard DNS queries, in the definitive public Internet DNS hierarchy. If a DRIP public information registry lists, in a HIP RR, any HIP RVS servers for a given DRIP identifier, those RVS servers can restrict relay services per AAA policy; this requires extensions to [RFC8004]. These public information registries can use secure DNS transport (e.g. DNS over TLS) to deliver public information that is not inherently trustable (e.g. everything other than attestations).

5.2. Private Information Registry

5.2.1. Background

The private information required for DRIP identifiers is similar to that required for Internet domain name registration. A DRIP identifier solution can leverage existing Internet resources: registration protocols, infrastructure and business models, by fitting into an ID structure compatible with DNS names. This implies some sort of hierarchy, for scalability, and management of this hierarchy. It is expected that the private registry function will be provided by the same organizations that run USS, and likely integrated with USS.

5.2.2. Proposed Approach

A DRIP private information registry can support essential Internet domain name registry operations (e.g. add, delete, update, query) using interoperable open standard protocols. It can also support the Extensible Provisioning Protocol (EPP) and the Registry Data Access Protocol (RDAP) with access controls. It might be listed in a DNS: that DNS could be private; but absent any compelling reasons for use of private DNS, a public DNS hierarchy needs to be in place. The DRIP private information registry in which a given UAS is registered needs to be findable, starting from the UAS ID, using the methods specified in [RFC7484]. A DRIP private information registry can also support WebFinger as specified in [RFC7033].

6. Harvesting Broadcast Remote ID messages for UTM Inclusion

ASTM anticipated that regulators would require both Broadcast RID and Network RID for large UAS, but allow RID requirements for small UAS to be satisfied with the operator’s choice of either Broadcast RID or Network RID. The EASA initially specified Broadcast RID for UAS of essentially all UAS and is now also considering Network RID. The FAA RID Final Rules only specifies Broadcast RID for UAS, however, still encourages Network RID for complementary functionality, especially in support of UTM.

One obvious opportunity is to enhance the architecture with gateways from Broadcast RID to Network RID. This provides the best of both and gives regulators and operators flexibility. It offers considerable enhancement over some Network RID options such as only reporting planned 4D operation space by the operator.

These gateways could be pre-positioned (e.g. around airports, public gatherings, and other sensitive areas) and/or crowd-sourced (as nothing more than a smartphone with a suitable app is needed). As Broadcast RID media have limited range, gateways receiving messages claiming locations far from the gateway can alert authorities or a

SDSP to the failed sanity check possibly indicating intent to deceive. Surveillance SDSPs can use messages with precise date/time/position stamps from the gateways to multilaterate UA location, independent of the locations claimed in the messages (which may have a natural time lag as it is), which are entirely operator self-reported in UAS RID and UTM.

Further, gateways with additional sensors (e.g. smartphones with cameras) can provide independent information on the UA type and size, confirming or refuting those claims made in the RID messages. This Crowd Sourced Remote ID (CS-RID) would be a significant enhancement, beyond baseline DRIP functionality; if implemented, it adds two more entity types.

6.1. The CS-RID Finder

A CS-RID Finder is the gateway for Broadcast Remote ID Messages into the UTM. It performs this gateway function via a CS-RID SDSP. A CS-RID Finder could implement, integrate, or accept outputs from, a Broadcast RID receiver. However, it can not interface directly with a GCS, Net-RID SP, Net-RID DP or Network RID client. It would present a TBD interface to a CS-RID SDSP; this interface needs to be based upon but readily distinguishable from that between a GCS and a Net-RID SP.

6.2. The CS-RID SDSP

A CS-RID SDSP would appear (i.e. present the same interface) to a Net-RID SP as a Net-RID DP. A CS-RID SDSP can not present a standard GCS-facing interface as if it were a Net-RID SP. A CS-RID SDSP would present a TBD interface to a CS-RID Finder; this interface can be based upon but readily distinguishable between a GCS and a Net-RID SP.

7. Privacy for Broadcast PII

Broadcast RID messages can contain PII. A viable architecture for PII protection would be symmetric encryption of the PII using a key known to the UAS and its USS. An authorized Observer could send the encrypted PII along with the UAS ID (to entities such as USS of the Observer, or to the UAS in which the UAS ID is registered if that can be determined from the UAS ID itself or to a Public Safety USS) to get the plaintext. Alternatively, the authorized Observer can receive the key to directly decrypt all future PII content from the UA.

PII can be protected unless the UAS is informed otherwise. This could come from operational instructions to even permit flying in a space/time. It can be special instructions at the start or during an operation. PII protection can not be used if the UAS loses connectivity to the USS. The UAS always has the option to abort the operation if PII protection is disallowed.

An authorized Observer can instruct a UAS via the USS that conditions have changed mandating no PII protection or land the UA (abort the operation).

8. Security Considerations

The security provided by asymmetric cryptographic techniques depends upon protection of the private keys. A manufacturer that embeds a private key in an UA may have retained a copy. A manufacturer whose UA are configured by a closed source application on the GCS which communicates over the Internet with the factory may be sending a copy of a UA or GCS self-generated key back to the factory. Keys may be extracted from a GCS or UA. The RID sender of a small harmless UA (or the entire UA) could be carried by a larger dangerous UA as a "false flag." Compromise of a registry private key could do widespread harm. Key revocation procedures are as yet to be determined. These risks are in addition to those involving Operator key management practices.

9. Acknowledgements

The work of the FAA's UAS Identification and Tracking (UAS ID) Aviation Rulemaking Committee (ARC) is the foundation of later ASTM and proposed IETF DRIP WG efforts. The work of ASTM F38.02 in balancing the interests of diverse stakeholders is essential to the necessary rapid and widespread deployment of UAS RID. IETF volunteers who have contributed to this draft include Amelia Andersdotter and Mohamed Boucadair.

10. References

10.1. Normative References

[I-D.ietf-drip-reqs]
Card, S., Wiethuechter, A., Moskowitz, R., and A. Gurtov, "Drone Remote Identification Protocol (DRIP) Requirements", Work in Progress, Internet-Draft, draft-ietf-drip-reqs-06, 1 November 2020, <http://www.ietf.org/internet-drafts/draft-ietf-drip-reqs-06.txt>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Informative References

[CTA2063A] ANSI, "Small Unmanned Aerial Systems Serial Numbers", 2019.

[Delegated] European Union Aviation Safety Agency (EASA), "EU Commission Delegated Regulation 2019/945 of 12 March 2019 on unmanned aircraft systems and on third-country operators of unmanned aircraft systems", 2019.

[F3411-19] ASTM, "Standard Specification for Remote ID and Tracking", 2019.

[FAA_RID] United States Federal Aviation Administration (FAA), "Remote Identification of Unmanned Aircraft", 2021, <https://www.govinfo.gov/content/pkg/FR-2021-01-15/pdf/2020-28948.pdf>.

[FAA_UAS_Concept_Of_Ops] United States Federal Aviation Administration (FAA), "Unmanned Aircraft System (UAS) Traffic Management (UTM) Concept of Operations (V2.0)", 2020, <https://www.faa.gov/uas/research_development/traffic_management/media/UTM_ConOps_v2.pdf>.

[I-D.ietf-drip-rid] Moskowitz, R., Card, S., Wiethuechter, A., and A. Gurtov, "UAS Remote ID", Work in Progress, Internet-Draft, draft-ietf-drip-rid-06, 31 December 2020, <http://www.ietf.org/internet-drafts/draft-ietf-drip-rid-06.txt>.

[Implementing] European Union Aviation Safety Agency (EASA), "EU Commission Implementing Regulation 2019/947 of 24 May 2019 on the rules and procedures for the operation of unmanned aircraft", 2019.

[LAANC] United States Federal Aviation Administration (FAA), "Low Altitude Authorization and Notification Capability", n.d., <https://www.faa.gov/uas/programs_partnerships/data_exchange/>.

[NPRM] United States Federal Aviation Administration (FAA), "Notice of Proposed Rule Making on Remote Identification of Unmanned Aircraft Systems", 2019.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/info/rfc1034>.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, DOI 10.17487/RFC3972, March 2005, <https://www.rfc-editor.org/info/rfc3972>.

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, RFC 5730, DOI 10.17487/RFC5730, August 2009, <https://www.rfc-editor.org/info/rfc5730>.

[RFC5731] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Domain Name Mapping", STD 69, RFC 5731, DOI 10.17487/RFC5731, August 2009, <https://www.rfc-editor.org/info/rfc5731>.

[RFC7033] Jones, P., Salgueiro, G., Jones, M., and J. Smarr, "WebFinger", RFC 7033, DOI 10.17487/RFC7033, September 2013, <https://www.rfc-editor.org/info/rfc7033>.

[RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <https://www.rfc-editor.org/info/rfc7401>.

[RFC7482] Newton, A. and S. Hollenbeck, "Registration Data Access Protocol (RDAP) Query Format", RFC 7482, DOI 10.17487/RFC7482, March 2015, <https://www.rfc-editor.org/info/rfc7482>.

[RFC7484] Blanchet, M., "Finding the Authoritative Registration Data (RDAP) Service", RFC 7484, DOI 10.17487/RFC7484, March 2015, <https://www.rfc-editor.org/info/rfc7484>.

[RFC8004] Laganier, J. and L. Eggert, "Host Identity Protocol (HIP) Rendezvous Extension", RFC 8004, DOI 10.17487/RFC8004, October 2016, <https://www.rfc-editor.org/info/rfc8004>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

[TS-22.825] 3GPP, "UAS RID requirement study", n.d., <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3527>.

[U-Space] European Organization for the Safety of Air Navigation (EUROCONTROL), "U-space Concept of Operations", 2019, <https://www.sesarju.eu/sites/default/files/documents/u-space/CORUS%20ConOps%20vol2.pdf>.

Appendix A. Overview of Unmanned Aircraft Systems (UAS) Traffic Management (UTM)

A.1. Operation Concept

The National Aeronautics and Space Administration (NASA) and FAAs' effort of integrating UAS's operation into the national airspace system (NAS) leads to the development of the concept of UTM and the ecosystem around it. The UTM concept was initially presented in 2013 and version 2.0 is published in 2020 [FAA_UAS_Concept_Of_Ops].

The eventual development and implementation are conducted by the UTM research transition team which is the joint workforce by FAA and NASA. World efforts took place afterward. The Single European Sky ATM Research (SESAR) started the CORUS project to research its UTM counterpart concept, namely [U-Space]. This effort is led by the European Organization for the Safety of Air Navigation (Eurocontrol).

Both NASA and SESAR have published the UTM concept of operations to guide the development of their future air traffic management (ATM) system and make sure safe and efficient integrations of manned and unmanned aircraft into the national airspace.

The UTM composes of UAS operation infrastructure, procedures and local regulation compliance policies to guarantee UAS's safe integration and operation. The main functionality of a UTM includes, but is not limited to, providing means of communication between UAS operators and service providers and a platform to facilitate communication among UAS service providers.

A.2. UAS Service Supplier (USS)

A USS plays an important role to fulfill the key performance indicators (KPIs) that a UTM has to offer. Such Entity acts as a proxy between UAS operators and UTM service providers. It provides services like real-time UAS traffic monitor and planning, aeronautical data archiving, airspace and violation control, interacting with other third-party control entities, etc. A USS can coexist with other USS(s) to build a large service coverage map which can load-balance, relay and share UAS traffic information.

The FAA works with UAS industry shareholders and promotes the Low Altitude Authorization and Notification Capability [LAANC] program which is the first system to realize some of the UTM envisioned functionality. The LAANC program can automate the UAS's flight plan application and approval process for airspace authorization in real-time by checking against multiple aeronautical databases such as airspace classification and fly rules associated with it, FAA UAS facility map, special use airspace, Notice to Airman (NOTAM), and Temporary Flight Rule (TFR).

A.3. UTM Use Cases for UAS Operations

This section illustrates a couple of use case scenarios where UAS participation in UTM has significant safety improvement.

1. For a UAS participating in UTM and takeoff or land in a controlled airspace (e.g., Class Bravo, Charlie, Delta and Echo in United States), the USS where UAS is currently communicating with is responsible for UAS's registration, authenticating the UAS's fly plan by checking against designated UAS fly map database, obtaining the air traffic control (ATC) authorization and monitor the UAS fly path in order to maintain safe boundary and follow the pre-authorized route.

2. For a UAS participating in UTM and take off or land in an uncontrolled airspace (ex. Class Golf in the United States), pre-fly authorization must be obtained from a USS when operating beyond-visual-of-sight (BVLOS) operation. The USS either accepts or rejects received intended fly plan from the UAS. Accepted UAS operation may share its current fly data such as GPS position and altitude to USS. The USS may keep the UAS operation status near real-time and may keep it as a record for overall airspace air traffic monitor.

A.4. Automatic Dependent Surveillance Broadcast (ADS-B)

The ADS-B is the de facto technology used in manned aviation for sharing location information, which is a ground and satellite based system designed in the early 2000s. Broadcast RID is conceptually similar to ADS-B. However, for numerous technical and regulatory reasons, ADS-B itself is not suitable for low-flying small UA. Technical reasons include: needing RF-LOS to large, expensive (hence scarce) ground stations; needing both a satellite receiver and 1090 MHz transceiver onboard CSWaP constrained UA; the limited bandwidth of both uplink and downlink, which are adequate for the current manned aviation traffic volume, but would likely be saturated by large numbers of UAS, endangering manned aviation; etc. Understanding these technical shortcomings, regulators world-wide have ruled out use of ADS-B for the small UAS for which UAS RID and DRIP are intended.

Authors' Addresses

Stuart W. Card
AX Enterprize
4947 Commercial Drive
Yorkville, NY, 13495
United States of America

Email: stu.card@axenterprize.com

Adam Wiethuechter
AX Enterprize
4947 Commercial Drive
Yorkville, NY, 13495
United States of America

Email: adam.wiethuechter@axenterprize.com

Robert Moskowitz
HTT Consulting
Oak Park, MI, 48237
United States of America

Email: rgm@labs.htt-consult.com

Shuai Zhao
Tencent
2747 Park Blvd

Palo Alto, 94588
United States of America

Email: shuai.zhao@ieee.org

Andrei Gurtov
Linkoeping University
IDA
SE-58183 Linkoeping Linkoeping
Sweden

Email: gurtov@acm.org

DRIP Working Group A. Wiethuechter
Internet-Draft S. Card
Intended status: Standards Track AX Enterprize, LLC
Expires: 21 June 2021 R. Moskowitz
HTT Consulting
18 December 2020

DRIP Authentication Formats
draft-ietf-drip-auth-00

Abstract

This document describes how to include trust into the ASTM Remote ID specification defined in ASTM F3411-19 under a Broadcast Remote ID (RID) scenario. It defines a few different message schemes (based on the Authentication Message) that can be used to assure past messages sent by a UA and also act as an assurance for UA trustworthiness in the absence of Internet connectivity at the receiving node.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

UA Systems (UAS) are usually in a volatile environment when it comes to communication. UA are generally small with little computational (or flying) horsepower to carry standard communication equipment. This limits the mediums of communication to few viable options.

Observer systems (e.g. smartphones and tablets) place further constraints on the communication options. The Remote ID Broadcast messages MUST be available to applications on these platforms without modifying the devices.

The ASTM standard [F3411-19] focuses on two ways of communicating to a UAS for RID: Broadcast and Network.

This document will focus on adding trust to Broadcast RID in the current (and an expanded) Authentication Message format.

1.1. DRIP Requirements Addressed

The following [drip-requirements] will be addressed:

GEN 1: Provable Ownership This will be addressed using the Certificate Message type (Section 4.3.1.1).

GEN 2: Provable Binding This requirement is addressed using the Wrapped ASTM Message (Section 4.2.3.1.2), Manifest Message (Section 4.2.3.2) and Message Pack Signature (Section 4.2.3.1.1) types.

GEN 3: Provable Registration This requirement is addressed using the Certificate Message type (Section 4.3.1.1).

2. Terminology

2.1. Required Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. Definitions

See [drip-requirements] for common DRIP terms.

Aircraft: In this document whenever the word Aircraft is used it is referring to an Unmanned Aircraft (UA) not a Manned Aircraft.

3. Background

3.1. Problem Space and Focus

The current standard for Remote ID (RID) does not, in any meaningful capacity, address the concerns of trust in the UA space with communication in the Broadcast RID environment. This is a requirement that will need to be addressed eventually for various different parties that have a stake in the UA industry.

The following subsections will provide a high level reference to the ASTM standard for Authentication Messages and how their current limitations effect trust in the Broadcast RID environment.

3.2. ASTM Authentication Message

```
Page 0:
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+-----------------------------------------------+
| Auth Header   |                                               |
+---------------+  ASTM Authentication Headers  +---------------+
|                                               |               |
+-----------------------------------------------+               |
|                                                               |
|                                                               |
|                                                               |
|                Authentication Data / Signature                |
|                                                               |
|                                                               |
|                                                               |
+---------------------------------------------------------------+

Page 1 - 4:
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+-----------------------------------------------+
| Auth Header   |                                               |
+---------------+                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                Authentication Data / Signature                |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
+---------------------------------------------------------------+
```

Auth Header (1 byte):
    Contains Authentication Type (AuthType) and Page Number. For
    DRIP Authentication AuthType is a value of 0x5.

ASTM Authentication Headers: (6 bytes)
    Contains other header information for the Authentication
    Message from ASTM UAS RID Standard.

Authentication Data / Signature: (109 bytes: 17+23*4)
    Opaque authentication data.

Figure 1: Standard ASTM Authentication Message format

The above diagram is the format defined by ASTM [F3411-19] that is the frame which everything this document fits into. The specific details of the ASTM headers are abstracted away as they are not necessarily required for this document.

There is a 25th byte exclude in the diagrams that comes before the Auth Header. This is the ASTM Header and consists of the Protocol Version and Message Type of the given message frame/page.

4. DRIP Authentication Framing Formats

Currently the ASTM AuthType of 0x5 should be used to denote DRIP based Authentication. The max page count of the Authentication Message is increased to 10, instead of being capped at 5.

To keep consistent formatting across the different mediums (Bluetooth 4, Bluetooth 5 and Wifi NaN) and their independent restrictions the authentication data being sent is REQUIRED to fit within the first 9 pages of the Authentication Message. The final (10th) page of the message is reserved exclusively for Forward Error Correction bytes and is only present on Bluetooth 4.

4.1. DRIP General Frame

```
Page 0:
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+-----------------------------------------------+
|  Auth Header  |                                               |
+---------------+  ASTM Authentication Headers  +---------------+
|                                               |  DRIP Header  |
+-----------------------------------------------+---------------+
|                                                               |
|                                                               |
|                                                               |
|                   DRIP Authentication Data                    |
|                                                               |
|                                                               |
|                                                               |
+---------------------------------------------------------------+

Page 1 - Page N-1:
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+-----------------------------------------------+
|  Auth Header  |                                               |
+---------------+                                               |
|                                                               |
```

```
   |                                                               |
   |                                                               |
   |                    DRIP Authentication Data                   |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   +---------------------------------------------------------------+

   Page N:
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +---------------+-----------------------------------------------+
   |  Auth Header  |                                               |
   +---------------+                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                    Forward Error Correction                   |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   +---------------------------------------------------------------+

   DRIP Header (1 byte):
         7     6     5     4     3     2     1     0
       +-----+-----+-----+-----+-----+-----+-----+-----+
       | FEC |              DRIP AuthType              |
       +-----+-----+-----+-----+-----+-----+-----+-----+

       FEC (1 bit):
           Enabled [1] or Disabled [0]. Signals if Page N is
           filled with XOR FEC.

       DRIP AuthType (7 bits):
           DRIP AuthType                        Values
           -------------                        ------
           0 Wrapped ASTM Message(s)            0
           1 Wrapped ASTM Message(s)            1
           2 Wrapped ASTM Message(s)            2
           3 Wrapped ASTM Message(s)            3
           4 Wrapped ASTM Message(s)            4
           5 Wrapped ASTM Message(s)            5
           8 Byte Manifest                      6
           4 Byte Manifest                      7
```

```
Reserved (Wrapped Messages)          8-15
Certificate: Registry on Aircraft    16
Reserved (Certificates)              17-31
Private Use                          32-63
Reserved                             64-111
Experimental Use                     112-127
```

DRIP Authentication Data (200 bytes):
DRIP Authentication data. 0 to 200 bytes.

Forward Error Correction (23 bytes):
Optional and signaled using DRIP Header. Always last Authentication page.

Figure 2: DRIP General Frame Format

4.1.1. DRIP Header

The DRIP Header is used to signal what kind of Authentication under DRIP that the message is using and consists of two fields.

4.1.1.1. Forward Error Correction (Bit 8)

The Most Significant Bit is used to signal if FEC is present in the final page of the Authentication Message. It MUST be set to 1 if FEC is being used. This is only enabled under Bluetooth 4 and MUST be set to 0 on Bluetooth 5 or Wifi NaN.

4.1.1.2. DRIP AuthType (Bits 1-7)

The lower 7 bits are used as the DRIP AuthType field denoting what Authentication type is being used. There are 5 major areas carved out of the DRIP AuthType defined by the following bitmaps:

```
000 xxxx (0x00-0x0F): Wrapped Messages (16)
001 xxxx (0x10-0x1F): Certificates (16)
01x xxxx (0x20-0x3F): Private Use (32)
1xx xxxx (0x40-0x6F): Reserved (48)
111 xxxx (0x70-0x7F): Experimental Use (16)
```

Figure 3: DRIP Header Bitmasks

4.1.2. DRIP Authentication Data

This field has a maximum size of 200 bytes. If the data is less than the max and a page is only partially filled then the rest of the partially filled page must be null padded.

This section is generally filled with either the Wrapper Frame (Section 4.2) or the Attestation Frame (Section 4.3).

4.1.3. Forward Error Correction

To help Bluetooth (specifically Bluetooth 4) achieve the goal of reliable receipt of paged messages a Forward Error Correction (FEC) scheme is introduced and MUST be used for Legacy Advertising (Bluetooth 4) and MUST NOT be used for Extended Advertising (Bluetooth 5, Wifi NaN) under DRIP.

4.1.3.1. Encoding

A compliant implementation of this standard MUST use XOR for the FEC. When generating the parity the first byte of every Authentication Page MUST be exclude from the XOR operation. For pages 1 through N this leaves the data portion of the page while page 0 will include a number of headers along with 17 bytes of data.

To generate the parity a simple XOR operation using the previous and current page is used. For page 0, a 23 byte null pad is used for the previous page. The resulting 23 bytes of parity is appended in one full page (always the last) allowing for recovery when any single page is lost in transmission.

4.1.3.2. Decoding

Due to the nature of Bluetooth 4 and the existing ASTM paging structure an optimization can be used. If a Bluetooth frame fails its CRC check, then the frame is dropped without notification to the upper protocol layers. From the Remote ID perspective this means the loss of a complete frame/message/page. In Authentication Messages, each page is already numbered so the loss of a page allows the receiving application to build a "dummy" page filling the Authentication Data field (and ASTM Authentication Headers fields if page 0) with nulls.

Using the same methods as encoding, an XOR operation is used between the previous and current page (a 23 byte null pad is used when page 0 is the current page). The resulting 23 bytes is the data of the missing page.

If page 0 is being reconstructed an additional check of the Page Count, to check against how many pages are actually present, MUST be performed for sanity. An additional check on the Data Length field can also be performed, but is not required.

4.1.3.3. Limitations & Recommendations

If more than one page is lost (>1/5 for 5 page messages, >1/10 for 10 page messages) than the error rate of the link is already beyond saving and the application has more issues to deal with.

In theory under Bluetooth 4 up to 15 pages Authentication could be sent (9 pages reserved to Authentication and 6 pages reserved for Forward Error Correction). It is currently recommended however for a max of 10 pages total.

4.2. DRIP Wrapper Frame

This format MUST be encapsulated by the General Frame (Section 4.1) and reside in its data field (Section 4.1.2).

Typically the DRIP Header is set in the range of 0x00 through 0x0F (FEC disabled) or 0x80 through 0x8F (FEC enabled).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+---------------+---------------+---------------+
|                                                               |
|                        UA Hierarchical                        |
|                       Host Identity Tag                       |
|                                                               |
+---------------+---------------+---------------+---------------+
|                        Trust Timestamp                        |
+---------------+---------------+---------------+---------------+
|                                                               |
.                                                               .
.                      Authentication Data                      .
.                                                               .
|                                                               |
+---------------+---------------+---------------+---------------+
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                           Signature                           |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
```

```
|                                                               |
|                                                               |
|                                                               |
+---------------+---------------+---------------+---------------+

UA Hierarchial Host Identity Tag (16 bytes):
    The UAs HHIT in byte form. Hashed from the EdDSA25519
    public key.

Trust Timestamp (4 bytes):
    Timestamp denoting current time plus an offset to trust
    message to.

Authentication Data (116 bytes):
    Opaque authentication data using DRIP format specified in
    the DRIP Header. 0 to 116 bytes.

Signature (64 bytes):
    Signature over preceding fields using the EdDSA25519
    keypair of the UA.
```

Figure 4: DRIP Wrapper Frame Format

4.2.1. UA Hierarchical Host Identity Tag

To avoid needing the UAs HHIT via the ASTM Basic ID in a detached fashion the 16 byte HHIT of the UA is included in the wrapper frame.

The HHIT for the UA (and other entities in the RID and greater UTM system under DRIP) is an enhancement of the Host Identity Tag (HIT) [RFC7401] introducing hierarchy (and how they are used in UAS RID) as defined in [drip-rid].

4.2.2. Trust Timestamp

The Trust Timestamp is of the format defined in [F3411-19]. That is a UNIX timestamp offset by 01/01/2019 00:00:00. An additional offset is then added to push the timestamp a short time into the future to avoid replay attacks.

When wrapping a Vector (Position/Location) Message the payload WILL contain (by ASTM rules) constantly changing data, this includes its own timestamp. This timestamp is only 2 bytes, which is easily attacked and only expresses the 1/10th of seconds since the last hour.

Other ASTM message types, such as Basic ID and Self-ID are static messages with no changing data. To protect a replay of these signed messages the Trust Timestamp is the field during signing to be guaranteed to change.

The offset used against the UNIX timestamp is not defined in this document. Best practices to identify a acceptable offset should be used taking into consideration the UA environment, and propagation characteristics of the messages being sent.

4.2.3. Wrapped Authentication Data

This field has a maximum of 116 bytes in length.

4.2.3.1. Wrapped ASTM Message Formats

When wrapping any ASTM Messages and filling the Wrapped Authentication Data field under DRIP the messages MUST be in Message Type order as defined by ASTM. All message types except Authentication (0x2) and Message Pack (0xF) are allowed.

4.2.3.1.1. 0 Wrapped ASTM Message(s)

This payload type MUST only be used under Extended Advertisement (Bluetooth 5.X and Wifi NaN).

The Wrapped Authentication Data is the concatenation of all messages in the Message Pack (excluding Authentication) in Message Type order. No actual data payload is present in this format as the data is found outside the Authentication Message in the same Message Pack.

The DRIP Header is set to 0x00 (0).

4.2.3.1.2. 1 to 4 Wrapped ASTM Message(s)

This payload type can be used on either Legacy or Extended Advertisements.

The DRIP Header is set to 0x81-0x84 (129-134) when using Legacy Advertisements (FEC is enabled) and 0x01-0x04 (1-4) when using Extended Advertisements (FEC is disabled).

4.2.3.1.3. 5 Wrapped ASTM Message(s)

Editors Note: This payload type does not currently fit in the 116 byte limit of the Wrapper Frame. If the ASTM relaxes the Max Page Count limit for Legacy Advertisements to use all 15 pages then this is possible.

This payload type MUST only be used on Legacy Advertisements (Bluetooth 4.X). It requires 11 pages to complete.

The DRIP Header is set to 0x85 (133).

This payload type allows in Legacy Advertisements to have a pseudo-Message Pack like what is found in Extended Advertisements.

4.2.3.1.4. Limitations

When wrapping a single ASTM Message the 25 byte payload actually causes an inefficiency in the framing format, create a whole page unused except for a single byte. This can be optimized by removing a single byte out of the wrapped message but creates an issue on the receiver of knowing which byte was removed.

When sending a Location Message (Message Type 0x1) a single byte can be removed at the end of the message as it is currently unused. Many other messages in the ASTM Message set however do not have this ability. The first byte can not be removed as it is the key to know how to decode the message.

4.2.3.2. Manifests

Manifests fill the Wrapped Authentication Data field with hashes of previously send messages.

By hashing previously sent messages and signing them we gain trust in UAs previous reports. An observer who has been listening for any considerable length of time can hash received messages and cross check against listed hashes.

4.2.3.2.1. Hash Algorithm and Operation

The hash algorithm used for the Manifest Message is the same hash algorithm used in creation of the HHIT that is signing the Manifest.

A standard HHIT would be using cSHAKE128 from [NIST.SP.800-185]. With cSHAKE128, the hash is computed as follows:

```
cSHAKE128(MAC Address|Message, 8*H-Len, "", "RemoteID Auth Hash")
```

The message MAC Address of the transmitter is prepended to the message, as the MAC Address is the only information that links UA messages from a specific UA.

Editors Note: It should be noted that for Bluetooth mediums this is valid - however Wifi NaN does not give the receiver device the transmitters MAC Address - making this impossible. Either MAC Address should be removed entirely or something different be used in its place to link to a given UA. Thanks Soren Friis for pointing this out.

4.2.3.2.2. 8 Byte

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+---------------+---------------+---------------+
|                   Hash of Previous Manifest                   |
|                                                               |
+---------------+---------------+---------------+---------------+
|                    Hash of Current Manifest                   |
|                                                               |
+---------------+---------------+---------------+---------------+
|                         Message Hash 1                        |
|                                                               |
+---------------+---------------+---------------+---------------+
|                         Message Hash 2                        |
|                                                               |
+---------------+---------------+---------------+---------------+
.                                                               .
.                                                               .
.                                                               .
+---------------+---------------+---------------+---------------+
|                        Message Hash 12                        |
|                                                               |
+---------------+---------------+---------------+---------------+

DRIP Header:
    With FEC: 0x87 [135] (RECOMMENDED)
    Without FEC: 0x07 [7]

Hash of Previous Manifest: (8 bytes)
    A hash of the previously sent Authentication message.

Hash of Current Manifest: (8 bytes)
    A hash of the current Authentication message.

Message Hash: (8 bytes)
    A hash of a previously sent message. 12 max.
```

Figure 5: 4 Byte Manifest

4.2.3.2.3. 4 Byte

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+---------------+---------------+---------------+
|                   Hash of Previous Manifest                   |
+---------------+---------------+---------------+---------------+
|                    Hash of Current Manifest                   |
+---------------+---------------+---------------+---------------+
|                         Message Hash 1                        |
+---------------+---------------+---------------+---------------+
|                         Message Hash 2                        |
+---------------+---------------+---------------+---------------+
.                                                               .
.                                                               .
.                                                               .
+---------------+---------------+---------------+---------------+
|                        Message Hash 27                        |
+---------------+---------------+---------------+---------------+

DRIP Header:
    With FEC: 0x86 [132] (RECOMMENDED)
    Without FEC: 0x06 [6]

Hash of Previous Manifest: (4 bytes)
    A hash of the previously sent Authentication message.

Hash of Current Manifest: (4 bytes)
    A hash of the current Authentication message.

Message Hash: (4 bytes)
    A hash of a previously sent message. 27 max.
```

Figure 6: 4 Byte Manifest

4.2.3.2.4. Pseudo-Blockchain Hashes

Two special hashes are included in all Manifest messages; a previous manifest hash, which links to the previous manifest message, as well as a current manifest hash. This gives a pseudo-blockchain provenance to the manifest message that could be traced back if the observer was present for extended periods of time.

Creation: During creation and signing of this message format this field MUST be set to 0. So the signature will be based on this field being 0, as well as its own hash. It is an open question of if we compute the hash, then sign or sign then compute.

Cycling: There a few different ways to cycle this message. We can "roll up" the hash of 'current' to 'previous' when needed or to completely recompute the hash. This mostly depends on the previous note.

4.2.3.2.5. Manifest Limitation

A potential limitation to this format is dwell time of the UA. If the UA is not sticking to a general area then most likely the Observer will not obtain many (if not all) of the messages in the manifest. Without the original messages received no verification can be done. Examples of such scenarios include delivery or survey UA.

4.2.4. Wrapper Signature

The wrapper signature is generated using the private key half of the the UAs Host Identity (HI) and is done over all preceding data. ASTM/DRIP Headers are exclude from this operation only information within the Wrapper Fame (Section 4.2) is signed.

4.3. DRIP Attestation Frame

This format MUST be encapsulated by the General Frame (Section 4.1) and reside in its data field (Section 4.1.2).

This format is typically used to form a complete certificate using attestation data from a Registry defined in [identity-claims]. The DRIP Header is normally in the range of 0x10 through 0x1F (FEC disable) or 0x90 through 0x9F (FEC enabled).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+---------------+---------------+---------------+
|                                                               |
.                                                               .
.                        Attestation Data                       .
.                                                               .
|                                                               |
+---------------+---------------+---------------+---------------+
|                      Expiration Timestamp                     |
+---------------+---------------+---------------+---------------+
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                            Signature                          |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
+---------------+---------------+---------------+---------------+

Attestation Data: (up to 132 bytes):
    Data the UA asserts claim to.
    Up to 132 bytes in length.

Expiration Timestamp (4 bytes):
    Generated by the UA to protect against replay attacks.

Signature (64 bytes):
    Signature over preceding fields using the EdDSA25519
    keypair of the UA.
```

Figure 7: DRIP Attestation Format

4.3.1. Attestation Data

Any data up to 132 bytes in length that the UA wishes to assert truth to.

4.3.1.1. DRIP Certificate

This payload type can be used in either Legacy or Extended Advertising. It is used to grant the ability to authenticate UA Remote ID when the receiving device of the observer (e.g. a smartphone with a dedicated RID application) has no Internet service (e.g. LTE signal).

The DRIP Header is set to 0x90 (144) when used for Legacy Advertisements and 0x10 (16) for Extended Advertisements.

The Attestation Data field is filled with the Attestation: Registry on Aircraft (Section 3.2.2 Attestation: X on Y (Offline Form) from [identity-claims]). This is binding claim between the Registry and the Aircraft, asserting the relationship between the two entities. It also provides the UA Host Identity to allow signature verification of messages signed by the UA. Also included in its structure is the HHIT of the Registry to check the local shortlist of Registries that the Observer device trusts (mapping HHITs to HIs).

More details about this Attestation and other certificates and the provisioning process can be found in [identity-claims].

4.3.2. Expiration Timestamp

Generated by the UA during the creation of the Authentication message. It is set a short time into the future to protect against replay attacks of this DRIP format.

It shares the same format as the Trust Timestamp (Section 4.2.2).

4.3.3. Attestation Signature

Performed by the UA using the onboard keypair which matches the HHIT in the Basic ID Message (0x0).

5. Transport Methods & Recommendations

5.1. Legacy Advertisements (Bluetooth 4.X)

With Legacy Advertisements the goal is to attempt to bring reliable receipt of the paged Authentication Message. Forward Error Correction (Section 4.1.3) MUST be enabled when using Legacy Advertising methods (such as Bluetooth 4.X).

Under ASTM Bluetooth 4.X rules, transmission of dynamic messages are at least every 1 second while static messages (which is what

Authentication is classified under) are sent at least every 3 seconds.

Under DRIP the Certificate Message MUST be transmitted to properly meet the GEN 1 and GEN 3 requirement.

The ASTM Message Wrapper and Manifest both satisfy the GEN 2 requirement. At least one MUST be implemented to comply with the GEN 2 requirement.

A single Manifest can carry at most (using the full 10 page limit and 8 byte hashes) 12 unique hashes of previously sent messages (of any type). This results in a total of 22 (12 + 10) frames of Bluetooth data being transmitted over Bluetooth.

In comparison the Message Wrapper sends 6 pages (each a single frame) for each wrapped message. For backwards compatibility the implementation should also send the standard ASTM message that was wrapped for non-DRIP compliant receivers to obtain. This method results in 84 total Bluetooth frames (12 + (12 * 6)) sent.

The question of which is better suited is up to the implementation.

5.2. Extended Advertisements (Bluetooth 5.X and Wifi NaN)

Under the ASTM specification, Bluetooth 5 or Wifi NaN transport of Remote ID is to use the Message Pack (Type 0xF) format for all transmissions. Under Message Pack all messages are sent together (in Message Type order) in a single Bluetooth frame (up to 9 single frame equivalent messages). Message Packs are required by ASTM to be sent at a rate of 1 per second (like dynamic messages).

Without any fragmentation or loss of pages with transmission Forward Error Correction (Section 4.1.3) MUST NOT be used as it is impractical.

6. ASTM Considerations

* Increase Authentication Max Page Count from 5 to 10. Legacy Advertising can use all 10 while Extended Advertising has a maximum of 9 due to Bluetooth 5 limitations.

* Allocate Authentication Type 0x5 for DRIP from ASTM AuthType field.

7. IANA Considerations

This document does not require any actions by IANA.

8. Security Considerations

TODO

(Ed. Note: Hash lengths (length vs strength/collision rate); replay attacks with timestamps; static Cra (issue but nulled if UA signing other stuff dynamically meaning signatures will fail as HI won't match - this is probably a deeper discussion topic for provisioning security considerations when we get to there))

9. Acknowledgments

Ryan Quigley and James Mussi of AX Enterprize, LLC for early prototyping to find holes in the draft specifications.

10. Appendix A: Thoughts on ASTM Authentication Message

The format standardized by the ASTM is designed with a few major considerations in mind, which the authors of this document feel put significant limitations on the expansion of the standard.

The primary consideration (in this context) is the use of the Bluetooth 5.X Extended Frame format. This method allows for a 255 byte payload to be sent in what the ASTM refers to as a "Message Pack".

The idea is to include up to five standard ASTM Broadcast RID messages (each of which are 25 bytes) plus a single authentication message (5 pages of 25 bytes each) in the Message Pack. The reasoning is then the Authentication Message is for the entire Message Pack.

The authors have no issues with this proposed approach; this is a valid format to use for the Authentication Message provided by the ASTM. However, by limiting the Authentication Message to ONLY five pages in the standard it ignores the possibility of other formatting options to be created and used.

Another issue with this format, not fully addressed in this document is fragmentation. Under Bluetooth 4.X, each page is sent separately which can result in lose of pages on the receiver. This is disastrous as the loss of even a single page means any signature is incomplete.

With the current limitation of 5 pages, Forward Error Correction (FEC) is nearly impossible without sacrificing the amount of data sent. More pages would allow FEC to be performed on the Authentication Message pages so loss of pages can be mitigated.

All these problems are further amplified by the speed at which UA fly and the Observer's position to receive transmissions. There is no guarantee that the Observer will receive all the pages of even a 5 page Authentication Message in the time it takes a UA to traverse across their line of sight. Worse still is that is not including other UA in the area, which congests the spectrum and could cause further confusion attempting to collate messages from various UA. This specific problem is out of scope for this document and our solutions in general, but should be noted as a design consideration.

11. References

11.1. Normative References

[F3411-19] "Standard Specification for Remote ID and Tracking", February 2020.

[NIST.SP.800-185] Kelsey, J., Change, S., and R. Perlner, "SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash and ParallelHash", DOI 10.6028/nist.sp.800-185, NIST Special Publication SP 800-185, December 2016, <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-185.pdf>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

11.2. Informative References

[drip-requirements] Card, S., Wiethuechter, A., Moskowitz, R., and A. Gurtov, "Drone Remote Identification Protocol (DRIP) Requirements", Work in Progress, Internet-Draft, draft-ietf-drip-reqs-06, 1 November 2020, <http://www.ietf.org/internet-drafts/draft-ietf-drip-reqs-06.txt>.

[drip-rid] Moskowitz, R., Card, S., Wiethuechter, A., and A. Gurtov, "UAS Remote ID", Work in Progress, Internet-Draft, draft-ietf-drip-uas-rid-01, 9 September 2020, <http://www.ietf.org/internet-drafts/draft-ietf-drip-uas-rid-01.txt>.

[identity-claims]
Wiethuechter, A., Card, S., and R. Moskowitz, "DRIP Identity Claims", Work in Progress, Internet-Draft, draft-wiethuechter-drip-identity-claims-03, 2 November 2020, <http://www.ietf.org/internet-drafts/draft-wiethuechter-drip-identity-claims-03.txt>.

[RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <https://www.rfc-editor.org/info/rfc7401>.

Authors' Addresses

Adam Wiethuechter
AX Enterprize, LLC
4947 Commercial Drive
Yorkville, NY 13495
United States of America

Email: adam.wiethuechter@axenterprize.com

Stuart Card
AX Enterprize, LLC
4947 Commercial Drive
Yorkville, NY 13495
United States of America

Email: stu.card@axenterprize.com

Robert Moskowitz
HTT Consulting
Oak Park, MI 48237
United States of America

Email: rgm@labs.htt-consult.com

DRIP R. Moskowitz
Internet-Draft HTT Consulting
Updates: 7401, 7343 (if approved) S. Card
Intended status: Standards Track A. Wiethuechter
Expires: 1 August 2021 AX Enterprize, LLC
A. Gurtov
Linköping University
28 January 2021

UAS Remote ID
draft-ietf-drip-rid-07

Abstract

This document describes the use of Hierarchical Host Identity Tags (HHITs) as self-asserting IPv6 addresses and thereby a trustable Identifier for use as the UAS Remote ID. HHITs self-attest to the included explicit hierarchy that provides Registrar discovery for 3rd-party ID attestation.

Table of Contents

1. Introduction

[drip-requirements] describes a UAS ID as a "unique (ID-4), non-spoofable (ID-5), and identify a registry where the ID is listed (ID-2)"; all within a 20 character Identifier (ID-1).

This document describes the use of Hierarchical HITs (HHITs) (Appendix B) as self-asserting IPv6 addresses and thereby a trustable Identifier for use as the UAS Remote ID. HHITs include explicit hierarchy to provide Registrar discovery for 3rd-party ID attestation.

HITs are statistically unique through the cryptographic hash feature of second-preimage resistance. The cryptographically-bound addition of the Hierarchy and a HHIT registration process (TBD; e.g. based on Extensible Provisioning Protocol, [RFC5730]) provide complete, global HHIT uniqueness. This is in contrast to general IDs (e.g. a UUID or device serial number) as the subject in an X.509 certificate.

In a multi-CA PKI, a subject can occur in multiple CAs, possibly fraudulently. CAs within the PKI would need to implement an approach to enforce assurance of uniqueness.

Hierarchical HITs provide self attestation of the HHIT registry. A HHIT can only be in a single registry within a registry system (e.g. EPP and DNS).

Hierarchical HITs are valid, though non-routable, IPv6 addresses. As such, they fit in many ways within various IETF technologies.

1.1. Nontransferablity of HHITs

HIs and its HHITs SHOULD NOT be transferable between UA or even between replacement electronics for a UA. The private key for the HI SHOULD be held in a cryptographically secure component.

2. Terms and Definitions

2.1. Requirements Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. Notation

| Signifies concatenation of information - e.g., X | Y is the concatenation of X and Y.

Claim(X,Y):
Form of a predicate (X is Y, X has property Y, and most importantly X owns Y).

Assertion({X...}):
A set of one or more claims. This definition is borrowed from JWT/CWT.

Attestation(X,Y):
A signed claim. X attests to Y.

Certificate(X,Y):
A claim or attestation, Y, signed exclusively by a third party, X, and are only over identities.

2.3. Definitions

See [drip-requirements] for common DRIP terms.

cSHAKE (The customizable SHAKE function):
Extends the SHAKE scheme to allow users to customize their use of the function.

HDA (Hierarchical HIT Domain Authority):
The 16 bit field identifying the HHIT Domain Authority under an RAA.

HHIT
Hierarchical Host Identity Tag. A HIT with extra hierarchical information not found in a standard HIT.

HI
Host Identity. The public key portion of an asymmetric keypair used in HIP.

HID (Hierarchy ID):
The 32 bit field providing the HIT Hierarchy ID.

HIP
Host Identity Protocol. The origin of HI, HIT, and HHIT, required for DRIP. Optional full use of HIP enables additional DRIP functionality.

HIT
Host Identity Tag. A 128 bit handle on the HI. HITs are valid IPv6 addresses.

Keccak (KECCAK Message Authentication Code):
The family of all sponge functions with a KECCAK-f permutation as the underlying function and multi-rate padding as the padding rule.

RAA (Registered Assigning Authority):
The 16 bit field identifying the business or organization that manages a registry of HDAs.

RVS (Rendezvous Server):
The HIP Rendezvous Server for enabling mobility, as defined in [RFC8004].

SHAKE (Secure Hash Algorithm KECCAK):
A secure hash that allows for an arbitrary output length.

XOF (eXtendable-Output Function):
A function on bit strings (also called messages) in which the output can be extended to any desired length.

3. Hierarchical HITs as Remote ID

Hierarchical HITs are a refinement on the Host Identity Tag (HIT) of HIPv2 [RFC7401]. HHITs require a new ORCHID mechanism as described in Appendix C. HHITs for UAS ID also use the new EdDSA/SHAKE128 HIT suite defined in Appendix D (requirements GEN-2). This hierarchy, cryptographically embedded within the HHIT, provides the information for finding the UA's HHIT registry (ID-3).

The current ASTM [F3411-19] specifies three UAS ID types:

TYPE-1 A static, manufacturer assigned, hardware serial number per ANSI/CTA-2063-A "Small Unmanned Aerial System Serial Numbers" [CTA2063A].

TYPE-2 A CAA assigned (presumably static) ID.

TYPE-3 A UTM system assigned UUID [RFC4122], which can but need not be dynamic.

For HHITs to be used effectively as UAS IDs, F3411-19 SHOULD add UAS ID type 4 as HHIT.

3.1. Hierarchical HITs encoded as CTA-2063-A Serial Numbers

In some cases it is advantageous to encode HHITs as a CTA 2063-A Serial Number [CTA2063A]. For example, readings of the FAA Remote ID Rules [FAA_RID] seem to state that a Remote ID Module (i.e. not integrated with UA controller) must only use "the serial number of the unmanned aircraft"; CTA 2063-A meets this requirement. The encoding rules are defined in Appendix B.4.

3.2. Remote ID as one class of Hierarchical HITs

UAS Remote ID may be one of a number of uses of HHITs. As such these follow-on uses need to be considered in allocating the RAAs Appendix B.3.1 or HHIT prefix assignments Section 8.

3.3. Hierarchy in ORCHID Generation

ORCHIDS, as defined in [RFC7343], do not cryptographically bind the IPv6 prefix nor the Orchid Generation Algorithm (OGA) ID (the HIT Suite ID) to the hash of the HI. The justification then was attacks against these fields are DoS attacks against protocols using them.

HHITs, as defined in Appendix C, cryptographically bind all content in the ORCHID through the hashing function. Thus a recipient of a HHIT that has the underlying HI can directly act on all content in the HHIT. This provides a strong, self attestation for using the hierarchy to find the HHIT Registry.

3.4. Hierarchical HIT Registry

HHITs are registered to Hierarchical HIT Domain Authorities (HDAs). A registration process (TBD) ensures UAS ID global uniqueness (ID-4). It also provides the mechanism to create UAS Public/Private data associated with the HHIT UAS ID (REG-1 and REG-2).

The 2 levels of hierarchy within the HHIT allows for CAAs to have their own Registered Assigning Authority (RAA) for their National Air Space (NAS). Within the RAA, the CAAs can delegate HDAs as needed. There may be other RAAs allowed to operate within a given NAS; this is a policy decision by the CAA.

3.5. Remote ID Authentication using HHITs

The EdDSA25519 Host Identity (HI) [Appendix D] underlying the HHIT can be used in an 84 byte self proof attestation as shown in Appendix E to provide proof of Remote ID ownership (requirements GEN-1). An Internet lookup service like DNS can provide the HI and registration proof (requirements GEN-3).

Similarly the 200 byte offline self attestation shown in Appendix E.1 provide the same proofs without Internet access and with a small cache that contains the HDA's HI/HHIT and HDA meta-data. These self attestations are carried in the ASTM Authentication Message (Msg Type 0x2).

Hashes of previously sent ASTM messages can be placed in a signed "Manifest" Authentication Message (requirements GEN-2). This can be either a standalone Authentication Message, or an enhanced self attestation Authentication Message. Alternatively the ASTM Message Pack (Msg Type 0xF) can provide this feature, but only over Bluetooth 5 or WiFi NAN broadcasts.

4. UAS ID HHIT in DNS

There are 2 approaches for storing and retrieving the HHIT from DNS. These are:

* As FQDNs in the .aero TLD.

* Reverse DNS lookups as IPv6 addresses per [RFC8005].

The HHIT can be used to construct an FQDN that points to the USS that has the Public/Private information for the UA (REG-1 and REG-2). For example the USS for the HHIT could be found via the following. Assume the RAA is 100 and the HDA is 50. The PTR record is constructed as:

```
100.50.hhit.uas.aero   IN PTR      foo.uss.aero.
```

The individual HHITs are potentially too numerous (e.g. 60 - 600M) and dynamic to actually store in a signed, DNS zone. The HDA SHOULD provide DNS service for its zone and provide the HHIT detail response.

The HHIT reverse lookup can be a standard IPv6 reverse look up, or it can leverage off the HHIT structure. Assume a Prefix of 2001:30::/28, the RAA is 10 and the HDA is 20 and the HHIT is:

```
2001:30:a0:145:a3ad:1952:ad0:a69e
```

An HHIT reverse lookup could be to:

```
a69e.ad0.1952.a3ad.145.a0.30.2001.20.10.hhit.arpa.
```

A 'standard' ip6.arpa RR has the advantage of only one Registry service supported.

```
$ORIGIN  5.4.1.0.0.a.0.0.0.3.0.0.1.0.0.2.ip6.arpa.
e.9.6.a.0.d.a.0.2.5.9.1.d.a.3.a    IN   PTR
```

5. Other UTM uses of HHITs

HHITs can be used extensively within the UTM architecture beyond UA ID (and USS in UA ID registration and authentication). This includes a GCS HHIT ID. The GCS could use its HIIT if it is the source of Network Remote ID for securing the transport and for secure C2 transport [drip-secure-nrid-c2].

Observers SHOULD have HHITs to facilitate UAS information retrieval (e.g., for authorization to private UAS data). They could also use their HHIT for establishing a HIP connection with the UA Pilot for direct communications per authorization. Further, they can be used by FINDER observers, [crowd-sourced-rid].

6. DRIP Requirements addressed

This document provides solutions to GEN 1 - 3, ID 1 - 5, and REG 1 - 2.

7. ASTM Considerations

ASTM will need to make the following changes to the "UA ID" in the Basic Message (Msg Type 0x0):

Type 4:
   This document UA ID of Hierarchical HITs (see Section 3).

8. IANA Considerations

IANA will need to make the following changes to the "Host Identity Protocol (HIP) Parameters" registries:

Host ID:
This document defines the new EdDSA Host ID (see Appendix D.1).

HIT Suite ID:
This document defines the new HIT Suite of EdDSA/cSHAKE (see Appendix D.2).

HIT Suite ID:
This document defines two new HDA domain HIT Suites (see Appendix B.2.1).

Because HHIT format is not compatible with [RFC7343], IANA is requested to allocated a new 28-bit prefix out of the IANA IPv6 Special Purpose Address Block, namely 2001:0000::/23, as per [RFC6890].

9. Security Considerations

A 64 bit hash space presents a real risk of second pre-image attacks Section 9.2. The HHIT Registry services effectively block attempts to "take over" a HHIT. It does not stop a rogue attempting to impersonate a known HHIT. This attack can be mitigated by the receiver of the HHIT using DNS to find the HI for the HHIT.

Another mitigation of HHIT hijacking is if the HI owner (UA) supplies an object containing the HHIT and signed by the HI private key of the HDA such as Appendix E.1 as shown in Section 3.5.

The two risks with hierarchical HITs are the use of an invalid HID and forced HIT collisions. The use of a DNS zone (e.g. "hhit.arpa.") is a strong protection against invalid HIDs. Querying an HDA's RVS for a HIT under the HDA protects against talking to unregistered clients. The Registry service has direct protection against forced or accidental HIT hash collisions.

Cryptographically Generated Addresses (CGAs) provide a unique assurance of uniqueness. This is two-fold. The address (in this case the UAS ID) is a hash of a public key and a Registry hierarchy naming. Collision resistance (more important that it implied second-preimage resistance) makes it statistically challenging to attacks. A registration process (TBD) within the HDA provides a level of assured uniqueness unattainable without mirroring this approach.

The second aspect of assured uniqueness is the digital signing (attestation) process of the HHIT by the HI private key and the further signing (attestation) of the HI public key by the Registry's key. This completes the ownership process. The observer at this point does not know WHAT owns the HHIT, but is assured, other than the risk of theft of the HI private key, that this UAS ID is owned by something and is properly registered.

9.1. Hierarchical HIT Trust

The HHIT UAS RID in the ASTM Basic Message (Msg Type 0x0, the actual Remote ID message) does not provide any assertion of trust. The best that might be done within this Basic Message is 4 bytes truncated from a HI signing of the HHIT (the UA ID field is 20 bytes and a HHIT is 16). This is not trustable. Minimally, it takes 84 bytes, Appendix E, to prove ownership of a HHIT.

The ASTM Authentication Messages (Msg Type 0x2) as shown in Section 3.5 can provide practical actual ownership proofs. These attestations include timestamps to defend against replay attacks. But in themselves, they do not prove which UA actually sent the message. They could have been sent by a dog running down the street with a Broadcast Remote ID device strapped to its back.

Proof of UA transmission comes when the Authentication Message includes proofs for the ASTM Location/Vector Message (Msg Type 0x1) and the observer can see the UA or that information is validated by ground multilateration [crowd-sourced-rid]. Only then does an observer gain full trust in the HHIT Remote ID.

HHIT Remote IDs obtained via the Network Remote ID path provides a different approach to trust. Here the UAS SHOULD be securely communicating to the USS (see [drip-secure-nrid-c2]), thus asserting HHIT RID trust.

9.2. Collision risks with Hierarchical HITs

The 64 bit hash size does have an increased risk of collisions over the 96 bit hash size used for the other HIT Suites. There is a 0.01% probability of a collision in a population of 66 million. The probability goes up to 1% for a population of 663 million. See Appendix G for the collision probability formula.

However, this risk of collision is within a single "Additional Information" value, i.e. a RAA/HDA domain. The UAS/USS registration process should include registering the HHIT and MUST reject a collision, forcing the UAS to generate a new HI and thus HHIT and reapplying to the registration process.

9.3. Proofs Considerations

A major consideration is the optimization done in Certificate: X on Y (Concise Form) to get its length down to 200 bytes. The truncation of Certificate: HDA on HDA down to just its HHIT is one that could be used against the system to act as a false Registry. For this to occur an attacker would need to find a hash collision on that Registry HHIT and then manage to spoof all of DNS being used in the system.

The authors believe that the probability of such an attack is low when Registry operators are using best practices in security. If such an attack can occur (especially in the time frame of "one-time use IDs") then there are more serious issues present in the system.

10. References

10.1. Normative References

[F3411-19] ASTM International, "Standard Specification for Remote ID and Tracking", February 2020, <http://www.astm.org/cgi-bin/resolver.cgi?F3411>.

[NIST.SP.800-185] Kelsey, J., Change, S., and R. Perlner, "SHA-3 derived functions: cSHAKE, KMAC, TupleHash and ParallelHash", National Institute of Standards and Technology report, DOI 10.6028/nist.sp.800-185, December 2016, <https://doi.org/10.6028/nist.sp.800-185>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC6890] Cotton, M., Vegoda, L., Bonica, R., Ed., and B. Haberman, "Special-Purpose IP Address Registries", BCP 153, RFC 6890, DOI 10.17487/RFC6890, April 2013, <https://www.rfc-editor.org/info/rfc6890>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Informative References

[corus] CORUS, "U-space Concept of Operations", September 2019, <https://www.sesarju.eu/node/3411>.

[crowd-sourced-rid] Moskowitz, R., Card, S., Wiethuechter, A., Zhao, S., and H. Birkholz, "Crowd Sourced Remote ID", Work in Progress, Internet-Draft, draft-moskowitz-drip-crowd-sourced-rid-05, 15 November 2020, <https://tools.ietf.org/html/draft-moskowitz-drip-crowd-sourced-rid-05>.

[CTA2063A] ANSI/CTA, "Small Unmanned Aerial Systems Serial Numbers", September 2019, <https://shop.cta.tech/products/small-unmanned-aerial-systems-serial-numbers>.

[drip-requirements] Card, S., Wiethuechter, A., Moskowitz, R., and A. Gurtov, "Drone Remote Identification Protocol (DRIP) Requirements", Work in Progress, Internet-Draft, draft-ietf-drip-reqs-06, 1 November 2020, <https://tools.ietf.org/html/draft-ietf-drip-reqs-06>.

[drip-secure-nrid-c2] Moskowitz, R., Card, S., Wiethuechter, A., and A. Gurtov, "Secure UAS Network RID and C2 Transport", Work in Progress, Internet-Draft, draft-moskowitz-drip-secure-nrid-c2-02, 25 December 2020, <https://tools.ietf.org/html/draft-moskowitz-drip-secure-nrid-c2-02>.

[FAA_RID] United States Federal Aviation Administration (FAA), "Remote Identification of Unmanned Aircraft", 2020, <https://www.govinfo.gov/content/pkg/FR-2021-01-15/pdf/2020-28948.pdf>.

[Keccak] Bertoni, G., Daemen, J., Peeters, M., Van Assche, G., and R. Van Keer, "The Keccak Function", <https://keccak.team/index.html>.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <https://www.rfc-editor.org/info/rfc4122>.

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, RFC 5730, DOI 10.17487/RFC5730, August 2009, <https://www.rfc-editor.org/info/rfc5730>.

[RFC7343] Laganier, J. and F. Dupont, "An IPv6 Prefix for Overlay Routable Cryptographic Hash Identifiers Version 2 (ORCHIDv2)", RFC 7343, DOI 10.17487/RFC7343, September 2014, <https://www.rfc-editor.org/info/rfc7343>.

[RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <https://www.rfc-editor.org/info/rfc7401>.

[RFC8004] Laganier, J. and L. Eggert, "Host Identity Protocol (HIP) Rendezvous Extension", RFC 8004, DOI 10.17487/RFC8004, October 2016, <https://www.rfc-editor.org/info/rfc8004>.

[RFC8005] Laganier, J., "Host Identity Protocol (HIP) Domain Name System (DNS) Extension", RFC 8005, DOI 10.17487/RFC8005, October 2016, <https://www.rfc-editor.org/info/rfc8005>.

Appendix A. EU U-Space RID Privacy Considerations

EU is defining a future of airspace management known as U-space within the Single European Sky ATM Research (SESAR) undertaking. Concept of Operation for EuRopean UTM Systems (CORUS) project proposed low-level Concept of Operations [corus] for UAS in EU. It introduces strong requirements for UAS privacy based on European GDPR regulations. It suggests that UAs are identified with agnostic IDs, with no information about UA type, the operators or flight trajectory. Only authorized persons should be able to query the details of the flight with a record of access.

Due to the high privacy requirements, a casual observer can only query U-space if it is aware of a UA seen in a certain area. A general observer can use a public U-space portal to query UA details based on the UA transmitted "Remote identification" signal. Direct remote identification (DRID) is based on a signal transmitted by the UA directly. Network remote identification (NRID) is only possible for UAs being tracked by U-Space and is based on the matching the current UA position to one of the tracks.

The project lists "E-Identification" and "E-Registrations" services as to be developed. These services can follow the privacy mechanism proposed in this document. If an "agnostic ID" above refers to a completely random identifier, it creates a problem with identity resolution and detection of misuse. On the other hand, a classical HIT has a flat structure which makes its resolution difficult. The Hierarchical HITs provide a balanced solution by associating a registry with the UA identifier. This is not likely to cause a major conflict with U-space privacy requirements, as the registries are typically few at a country level (e.g. civil personal, military, law enforcement, or commercial).

Appendix B. The Hierarchical Host Identity Tag (HHIT)

The Hierarchical HIT (HHIT) is a small but important enhancement over the flat HIT space. By adding two levels of hierarchical administration control, the HHIT provides for device registration/ownership, thereby enhancing the trust framework for HITs.

HHITs represent the HI in only a 64 bit hash and uses the other 32 bits to create a hierarchical administration organization for HIT domains. Hierarchical HIT construction is defined in Appendix C. The input values for the Encoding rules are in Appendix C.1.

A HHIT is built from the following fields:

* IANA prefix (max 28 bit)

* 32 bit Hierarchy ID (HID)

* 4 (or 8) bit HIT Suite ID

* ORCHID hash (96 - prefix length - Suite ID length bits, e.g. 64) See Appendix C

The Context ID for the ORCHID hash is:

```
Context ID :=  0x00B5 A69C 795D F5D5 F008 7F56 843F 2C40
```

B.1. HHIT prefix

A unique IANA IPv6 prefix, no larger than 28 bit, for HHITs is recommended. It clearly separates the flat-space HIT processing from HHIT processing per Appendix C.

Without a unique prefix, the first 4 bits of the RRA would be interpreted as the HIT Suite ID per HIPv2 [RFC7401].

B.2. HHIT Suite IDs

The HIT Suite IDs specifies the HI and hash algorithms. Any HIT Suite ID can be used for HHITs. The 8 bit format is supported (only when the first 4 bits are ZERO), but this reduces the ORCHID hash length.

B.2.1. 8 bit HIT Suite IDs

Support for 8 bit HIT Suite IDs is allowed in Sec 5.2.10, [RFC7401], but not specified in how ORCHIDs are generated with these longer OGAs. Appendix C provides the algorithmic flexiblity, allowing for HDA custom HIT Suite IDs as follows:

HIT Suite	Four-bit ID	Eight-bit encoding
HDA Assigned 1	NA	0x0E
HDA Assigned 2	NA	0x0F

This feature may be used for large-scale experimenting with post quantum computing hashes or similar domain specific needs. Note that currently there is no support for domain specific HI algorithms.

B.3. The Hierarchy ID (HID)

The Hierarchy ID (HID) provides the structure to organize HITs into administrative domains. HIDs are further divided into 2 fields:

* 16 bit Registered Assigning Authority (RAA)

* 16 bit Hierarchical HIT Domain Authority (HDA)

B.3.1. The Registered Assigning Authority (RAA)

An RAA is a business or organization that manages a registry of HDAs. For example, the Federal Aviation Authority (FAA) could be an RAA.

The RAA is a 16 bit field (65,536 RAAs) assigned by a numbers management organization, perhaps ICANN's IANA service. An RAA must provide a set of services to allocate HDAs to organizations. It must have a public policy on what is necessary to obtain an HDA. The RAA need not maintain any HIP related services. It must maintain a DNS zone minimally for discovering HID RVS servers.

As HHITs may be used in many different domains, RAA should be allocated in blocks with consideration on the likely size of a particular usage. Alternatively, different Prefixes can be used to separate different domains of use of HHTs.

This DNS zone may be a PTR for its RAA. It may be a zone in a HHIT specific DNS zone. Assume that the RAA is 100. The PTR record could be constructed:

```
100.hhit.arpa   IN PTR      raa.bar.com.
```

B.3.2. The Hierarchical HIT Domain Authority (HDA)

An HDA may be an ISP or any third party that takes on the business to provide RVS and other needed services for HIP enabled devices.

The HDA is an 16 bit field (65,536 HDAs per RAA) assigned by an RAA. An HDA should maintain a set of RVS servers that its client HIP-enabled customers use. How this is done and scales to the potentially millions of customers is outside the scope of this document. This service should be discoverable through the DNS zone maintained by the HDA's RAA.

An RAA may assign a block of values to an individual organization. This is completely up to the individual RAA's published policy for delegation.

B.4. Encoding HHITs in CTA 2063-A Serial Numbers

In some cases it is advantageous to encode HHITs as a CTA 2063-A Serial Number [CTA2063A]. For example, readings of the FAA Remote ID Rules [FAA_RID] seem to state that a Remote ID Module (i.e. not integrated with UA controller) must only use "the serial number of the unmanned aircraft"; CTA 2063-A meets this requirement.

Encoding a HHIT within the 2063-A format is not simple. There is no place for the HID; there will need to be a mapping service from Manufacturer Code to HID. The HIT Suite ID and ORCHID hash will take 14 characters (see below), leaving only 1 character for the Manufacturer's use of other information.

A character in a CTA 2063-A Serial Number "shall include any combination of digits and uppercase letters, except the letters O and I, but may include all digits". This would allow for a Base34 encoding of the binary HIT Suite ID and ORCHID hash. Although, programatically, such a conversion is not hard, other technologies (e.g. credit card payment systems) that have used such odd base encoding have had performance challenges. Thus here a Base32 encoding will be used by also excluding the letters Z and S (too similar to the digits 2 and 5).

The low-order 68 bits (HIT Suite ID | ORCHID hash) of the HHIT SHALL be left-padded with 2 bits of ZERO. This 70 bit number will be encoded into 14 characters using the digit/letters above. The Manufacturer MAY use a Length Code of 14 or 15. If 15, the first character after the Length Code is set by the Manufacturer with the low order 14 characters for the encoded HIT Suite ID and ORCHID hash.

A mapping service (e.g. DNS) MUST provide a trusted (e.g. via DNSSEC) conversion of the 4 character Manufacturer Code to high-order 60 bits (Prefix | HID) of the HHIT. Definition of this mapping service is currently out of scope of this document.

Appendix C. ORCHIDs for Hierarchical HITs

This section improves on ORCHIDv2 [RFC7343] with three enhancements:

* Optional Info field between the Prefix and OGA ID.

* Increased flexibility on the length of each component in the ORCHID construction, provided the resulting ORCHID is 128 bits.

* Use of cSHAKE, NIST SP 800-185 [NIST.SP.800-185], for the hashing function.

The Keccak [Keccak] based cSHAKE XOF hash function is a variable output length hash function. As such it does not use the truncation operation that other hashes need. The invocation of cSHAKE specifies the desired number of bits in the hash output. Further, cSHAKE has a parameter 'S' as a customization bit string. This parameter will be used for including the ORCHID Context Identifier in a standard fashion.

This ORCHID construction includes the fields in the ORCHID in the hash to protect them against substitution attacks. It also provides for inclusion of additional information, in particular the hierarchical bits of the Hierarchical HIT, in the ORCHID generation. This should be viewed as an addendum to ORCHIDv2 [RFC7343], as it can produce ORCHIDv2 output.

C.1. Adding additional information to the ORCHID

ORCHIDv2 [RFC7343] is currently defined as consisting of three components:

```
ORCHID     :=  Prefix | OGA ID | Encode_96( Hash )

where:

Prefix          : A constant 28-bit-long bitstring value
                  (IANA IPv6 assigned).

OGA ID          : A 4-bit long identifier for the Hash_function
                  in use within the specific usage context.  When
                  used for HIT generation this is the HIT Suite ID.

Encode_96( )    : An extraction function in which output is obtained
                  by extracting the middle 96-bit-long bitstring
                  from the argument bitstring.
```

This addendum will be constructed as follows:

```
ORCHID     :=  Prefix (p) | Info (n) | OGA ID (o) | Hash (m)

where:

Prefix (p)      : An IANA IPv6 assigned prefix (max 28-bit-long).

Info (n)        : n bits of information that define a use of the
                  ORCHID.  n can be zero, that is no additional
                  information.

OGA ID (o)      : A 4 or 8 bit long identifier for the Hash_function
                  in use within the specific usage context.  When
                  used for HIT generation this is the HIT Suite ID.

Hash (m)        : An extraction function in which output is m bits.

p + n + o + m = 128 bits
```

With a 28 bit IPv6 Prefix, the remaining 100 bits can be divided in any manner between the additional information, OGA ID, and the hash output. Care must be taken in determining the size of the hash portion, taking into account risks like pre-image attacks. Thus 64 bits as used in Hierarchical HITs may be as small as is acceptable.

C.2. ORCHID Encoding

This addendum adds a different encoding process to that currently used in ORCHIDv2. The input to the hash function explicitly includes all the header content plus the Context ID. The header content consists of the Prefix, the Additional Information, and OGA ID (HIT Suite ID). Secondly, the length of the resulting hash is set by sum of the length of the ORCHID header fields. For example, a 28 bit Prefix with 32 bits for the HID and 4 bits for the OGA ID leaves 64 bits for the hash length.

To achieve the variable length output in a consistent manner, the cSHAKE hash is used. For this purpose, cSHAKE128 is appropriate. The the cSHAKE function call for this addendum is:

```
cSHAKE128(Input, L, "", Context ID)

Input      :=  Prefix | Additional Information | OGA ID | HOST_ID
L          :=  Length in bits of hash portion of ORCHID
```

For full Suite ID support (those that use fixed length hashes like SHA256), the following hashing can be used (Note: this does NOT produce output Identical to ORCHIDv2 for Prefix of /28 and Additional Information of ZERO length):

```
Hash[L](Context ID | Input)

Input      :=  Prefix | Additional Information | OGA ID | HOST_ID
L          :=  Length in bits of hash portion of ORCHID

Hash[L]    :=  An extraction function in which output is obtained
               by extracting the middle L-bit-long bitstring
               from the argument bitstring.
```

Hierarchical HIT uses the same context as all other HIPv2 HIT Suites as they are clearly separated by the distinct HIT Suite ID.

C.2.1. Encoding ORCHIDs for HITv2

This section is included to provide backwards compatibility for ORCHIDv2 [RFC7343] as used for HITv2 [RFC7401].

For HITv2s, the Prefix MUST be 2001:20::/28. Info is length ZERO (not included), and OGA ID is length 4. Thus the HI Hash is length 96. Further the Prefix and OGA ID are NOT included in the hash calculation. Thus the following ORCHID calculations for fixed output length hashes are used:

```
Hash[L](Context ID | Input)

Input      :=  HOST_ID
L          :=  96
Context ID :=  0xF0EF F02F BFF4 3D0F E793 0C3C 6E61 74EA

Hash[L]    :=  An extraction function in which output is obtained
               by extracting the middle L-bit-long bitstring
               from the argument bitstring.
```

For variable output length hashes use:

```
Hash[L](Context ID | Input)

Input      :=  HOST_ID
L          :=  96
Context ID :=  0xF0EF F02F BFF4 3D0F E793 0C3C 6E61 74EA

Hash[L]    :=  The L bit output from the hash function
```

Then the ORCHID is constructed as follows:

```
Prefix | OGA ID | Hash Output
```

C.3. ORCHID Decoding

With this addendum, the decoding of an ORCHID is determined by the Prefix and OGA ID (HIT Suite ID). ORCHIDv2 [RFC7343] decoding is selected when the Prefix is: 2001:20::/28.

For Hierarchical HITs, the decoding is determined by the presence of the HHIT Prefix as specified in the HHIT document.

C.4. Decoding ORCHIDs for HITv2

This section is included to provide backwards compatibility for ORCHIDv2 [RFC7343] as used for HITv2 [RFC7401].

HITv2s are identified by a Prefix of 2001:20::/28. The next 4 bits are the OGA ID. is length 4. The remaining 96 bits are the HI Hash.

Appendix D. Edward Digital Signature Algorithm for HITs

Edwards-Curve Digital Signature Algorithm (EdDSA) [RFC8032] are specified here for use as Host Identities (HIs) per HIPv2 [RFC7401]. Further the HIT_SUITE_LIST is specified as used in [RFC7343].

See Appendix B.2 for use of the HIT Suite for this document.

D.1. HOST_ID

The HOST_ID parameter specifies the public key algorithm, and for elliptic curves, a name. The HOST_ID parameter is defined in Section 5.2.19 of [RFC7401].

Algorithm profiles	Values	
EdDSA	13 [RFC8032]	(RECOMMENDED)

For hosts that implement EdDSA as the algorithm, the following ECC curves are available:

Algorithm	Curve	Values
EdDSA	RESERVED	0
EdDSA	EdDSA25519	1 [RFC8032]
EdDSA	EdDSA25519ph	2 [RFC8032]
EdDSA	EdDSA448	3 [RFC8032]
EdDSA	EdDSA448ph	4 [RFC8032]

D.2. HIT_SUITE_LIST

The HIT_SUITE_LIST parameter contains a list of the supported HIT suite IDs of the Responder. Based on the HIT_SUITE_LIST, the Initiator can determine which source HIT Suite IDs are supported by the Responder. The HIT_SUITE_LIST parameter is defined in Section 5.2.10 of [RFC7401].

The following HIT Suite ID is defined, and the relationship between the four-bit ID value used in the OGA ID field and the eight-bit encoding within the HIT_SUITE_LIST ID field is clarified:

HIT Suite	Four-bit ID	Eight-bit encoding	
RESERVED	0	0x00	
EdDSA/cSHAKE128	5	0x50	(RECOMMENDED)

The following table provides more detail on the above HIT Suite combinations. The input for each generation algorithm is the encoding of the HI as defined in this Appendix.

The output of cSHAKE128 is variable per the needs of a specific ORCHID construction. It is at most 96 bits long and is directly used in the ORCHID (without truncation).

Index	Hash function	HMAC	Signature algorithm family	Description
5	cSHAKE128	KMAC128	EdDSA	EdDSA HI hashed with cSHAKE128, output is variable

Table 1: HIT Suites

Appendix E. Example HHIT Self Attestation

This section shows example uses of HHIT RID to prove trustworthiness of the RID and attestation of registration to the RAA|HDA. These are examples only and other documents will provide fully specified attestations. Care has been taken in the example design to minimize the risk of replay attacks.

This ownership/attestation of a HHIT can be proved in 84 bytes via the following HHIT Self Attestation following Appendix F.2.1 format:

* 4 byte Signing Timestamp

* 16 byte HHIT

* 64 byte Signature (EdDSA25519 signature)

The Timestamp MAY be the standard UNIX time at the time of signing. A protocol specific timestamp may be used to avoid programming complexities. For example, [F3411-19] uses a 00:00:00 01/01/2019 offset.

To minimize the risk of replay, the UA SHOULD create a new Self Attestation, with a new timestamp, at least once a minute. The UA MAY precompute these attestations and transmit during the appropriate 1 minute window. 1 minute is chosen as a balance between attestation compute time against risk. A shorter window of use lessens the risk of replay.

The signature is over the 20 byte Timestamp + HHIT.

The receiver of such an attestation would need access to the underlying public key (HI) to validate the signature. This may be obtained via a DNS query using the HHIT. A larger (116 bytes) Self Attestation could include the EdDSA25519 HI. This larger 116 attestation allows for signature validation before HHIT lookup to prove registration attestation.

E.1. HHIT Offline Self Attestation

Ownership and RAA|HDA registration of a HHIT can be proved in 200 bytes without Internet access and a small cache via the following HHIT Offline Self Attestation Appendix F.2 format:

* 16 byte UA HHIT

* 32 byte UA EdDSA25519 HI

* 4 byte HDA Signing Expiry Timestamp

* 16 byte HDA HHIT

* 64 byte HDA Signature (EdDSA25519 signature)

* 4 byte UA Signing Timestamp

* 64 byte UA Signature (EdDSA25519 signature)

The Timestamps MAY be the standard UNIX time at the time of signing. A protocol specific timestamp may be used to avoid programming complexities. For example, [F3411-19] uses a 00:00:00 01/01/2019 offset.

The HDA signature is over the 68 byte UA HHIT + UA HI + HDA Expiry Timestamp + HDA HHIT. During the UA Registration process, the UA would provide a Self Attestation to the HDA. The HDA would construct its attestation of registry with an Expiry Timestamp, its own HHIT, and its signature, returning a 132 byte HDA Registry Attestation to the UA. The UA would use this much the same way as its HHIT only in the Self Attestation above, creating a 200 byte Offline Self Attestation.

The receiver of such an attestation would need a cache of RAA ID, HDA ID, HDA HHIT, and HDA HI (min 80 bytes per RAA/HDA).

Appendix F. DRIP Proofs

The DRIP Proofs are a set of custom objects to be used in the USS/UTM system. They are created during the enrollment of an Operator and the provisioning of an Aircraft and are tied to the Operator ID and UAS RID.

These structures, when chained together, create two distinct roots of trust. One back to the UAS manufacturer, back to the initial production of a given Aircraft. The other back to the authorizing CAA. These chains can also be used by authorized entities to trace an Aircraft through all owners and flights in the Aircraft's lifetime (something of interest to ICAO).

The rest of this section will define the formats of proofs in DRIP as forms of certificates and attestations and their common uses.

F.1. Claim / Assertion: HHIT

The HHIT can be taken in its entirety as a single claim or broken into various claims and thus be classified as an assertion.

There are a number of different claims that an HHIT can be broken into:

* Valid ORCHID construction. To validate would require the Host Identity used.

* Ownership of the asymmetric keypair used to generate the hash.

* Being a member of a specified Registry. This is defined by the RAA and HDA pairing encoded. This is a baseless claim on its own that is attested to by the Registry.

F.2. Self-Attestation: Attestation(X,X)

This DRIP Proof is a self-signed attestation (by an entity known as 'X') staking an unverified claim on a HHIT/HI pairing until an expiration date/time.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +---------------+---------------+---------------+---------------+
    |                                                               |
    |                          Hierarchical                         |
    |                       Host Identity Tag                       |
    |                                                               |
    +---------------+---------------+---------------+---------------+
    |                                                               |
    |                                                               |
    |                                                               |
    |                              Host                             |
    |                            Identity                           |
    |                                                               |
    |                                                               |
    |                                                               |
    +---------------+---------------+---------------+---------------+
    |                      Expiration Timestamp                     |
    +---------------+---------------+---------------+---------------+
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                            Signature                          |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    +---------------+---------------+---------------+---------------+
```

HHIT The HHIT of the entity, derived from the HI and other information.

HI The HI of the entity. This is the public half of an EdDSA25519 asymmetric keypair.

Expiration Timestamp A timestamp signaling the expiration of the attestation.

Signature Generated using the asymmetric keypair of the entity.

Figure 1: Self-Attestation: Attestation(X,X)

This Self-Attestation is 116 bytes attesting to a number of claims and assertions. Overall the entire structure creates an assertion of the ownership of this first two claims (HHIT and HI), a binding (between HHIT and HI) and an upper time bound of relevance (the Expiration Timestamp).

The offset of the Expiration Timestamp (ETS) SHOULD be of significant length (possibly years).

These are 5 (five) Self-Attestations that can be created in a standard DRIP UAS RID system:

* Attestation(Manufacturer, Manufacturer)

* Attestation(RAA, RAA)

* Attestation(HDA, HDA) or Attestation(Registry, Registry)

* Attestation(Operator, Operator)

* Attestation(Aircraft, Aircraft)

This is not an exhaustive list as any entity with the DRIP UAS system SHOULD have a Self-Attestation for itself.

The Timestamp formatting is covered in Appendix F.5.

F.2.1. Concise Self-Attestation: Attestation(X, ConciseX)

A smaller version of Attestation(X, X) exists where the Host Identity is removed allowing a claim to be made in 84 bytes.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +---------------+---------------+---------------+---------------+
   |                                                               |
   |                         Hierarchical                          |
   |                      Host Identity Tag                        |
   |                                                               |
   +---------------+---------------+---------------+---------------+
   |                    Expiration Timestamp                       |
   +---------------+---------------+---------------+---------------+
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                          Signature                            |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   |                                                               |
   +---------------+---------------+---------------+---------------+
```

HHIT — The HHIT of the entity, derived from the HI and other information.

Expiration Timestamp — A timestamp signaling the expiration of the attestation.

Signature — Generated using the asymmetric keypair of the entity.

Figure 2: Concise Self-Attestation: Attestation(X, ConciseX)

This form would require that the Host Identity associated with the HHIT be in a public Registry to be requested (nominally with a DNS lookup using a HIP RR type) and checked against.

The Timestamp formatting is covered in Appendix F.5.

F.3. Certificate(X, Y)

This DRIP Proof is an attestation where Entity X asserts trust in the binding claimed by Entity Y (in Assertion Y) and signs this asserting with a timestamp and an expiration of when the binding is no longer asserted by Entity X.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+---------------+---------------+---------------+
|           Length Ax           |           Length Ay           |
+---------------+---------------+---------------+---------------+
|                                                               |
.                                                               .
.                          Assertion X                          .
.                                                               .
|                                                               |
+---------------+---------------+---------------+---------------+
|                                                               |
.                                                               .
.                          Assertion Y                          .
.                                                               .
|                                                               |
+---------------+---------------+---------------+---------------+
|                           Timestamp                           |
+---------------+---------------+---------------+---------------+
|                     Expiration Timestamp                      |
+---------------+---------------+---------------+---------------+
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                           Signature                           |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
+---------------+---------------+---------------+---------------+
```

Length Ax — Length in bytes of Assertion(X). Encoded as an unsigned integer.

Length Ay	Length in bytes of Assertion(Y). Encoded as an unsigned integer.
Assertion(X)	The attestation/certificate of entity X.
Assertion(Y)	The attestation/certificate of entity Y.
Timestamp	A timestamp signalling the current time at signing of the certificate.
Expiration Timestamp	A timestamp signaling the expiration of the attestation.
Signature	Generated using the asymmetric keypair of the entity.

Figure 3: Certificate(X, Y)

Cxy Form wraps both Self-Attestations of the entities and is signed by Entity X. Two timestamps, one taken at the time of signing and one as an expiration time are used to set boundaries to the assertion. Care should be given to how far into the future the Expiration Timestamp is set, but is left up to system policy.

Most attestations of this form have a length of 304 bytes; some may be 84 or 116 bytes. Certificate(Registry, Certificate(Operator,Aircraft)) is unique in that is 680 bytes long, binding of two Cxy forms (in this specific case Certificate(Registry, Operator) with Certificate(Operator, Aircraft).

The Timestamp formatting is covered in Appendix F.5.

F.3.1. Concise Certificate(X, Concise Y)

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +---------------+---------------+---------------+---------------+
    |                                                               |
    |                 Hierarchical Host Identity Tag                |
    |                          of Entity X                          |
    |                                                               |
    +---------------+---------------+---------------+---------------+
    |                                                               |
    .                                                               .
    .                              Ayy                              .
    .                                                               .
    |                                                               |
    +---------------+---------------+---------------+---------------+
    |                      Expiration Timestamp                     |
    +---------------+---------------+---------------+---------------+
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                            Signature                          |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    |                                                               |
    +---------------+---------------+---------------+---------------+
```

Figure 4: Concise Certificate(X, Concise Y)

The short form of the Cxy this attestation is 200 bytes long and is designed to fit inside the framing of the ASTM F3411 Authentication Message. The HHIT of Entity X is used in place of the full Axx (see Section 9.3 for comments). The timestamp is removed and only an expiration timestamp is present. Ayy MUST NOT be the in Concise Form.

During creation the Expiration Timestamp MUST be no later than the Expiration Timestamp found in Ayy.

F.4. Offline Broadcast Attestation: Attestation(X, Offline Y)

A special attestation that is the basis for a certificate finalized onboard the aircraft during flight. It is used in Broadcast RID to provide the trustworthiness of the Aircraft without the need of the Observer to be connected to the Internet.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+---------------+---------------+---------------+
|                                                               |
|                 Hierarchical Host Identity Tag                |
|                          of Entity X                          |
|                                                               |
+---------------+---------------+---------------+---------------+
|                                                               |
|                 Hierarchical Host Identity Tag                |
|                          of Entity Y                          |
|                                                               |
+---------------+---------------+---------------+---------------+
|                                                               |
|                                                               |
|                                                               |
|                   Host Identity of Entity Y                   |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
+---------------+---------------+---------------+---------------+
|                     Expiration Timestamp                      |
+---------------+---------------+---------------+---------------+
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                           Signature                           |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
+---------------+---------------+---------------+---------------+
```

Figure 5: Offline Form: Attestation(X, Offline Y)

The signature is generated using Entity X's keypair.

F.5. Timestamps

Timestamps MAY be the standard UNIX time or a protocol specific timestamp, to avoid programming complexities. For example [F3411-19] uses a 00:00:00 01/01/2019 offset. When a Expiration Timestamp is required a desired offset is added, setting the timestamp into the future. The amount of offset for specific timestamps is left to best practice.

F.6. Signatures

Signatures are ALWAYS taken over the preceding fields in the certificate/attestation. For DRIP the EdDSA25519 algorithm from [RFC8032] is used.

Appendix G. Calculating Collision Probabilities

The accepted formula for calculating the probability of a collision is:

$$p = 1 - e^{-k^2/(2n)}$$

P Collision Probability
n Total possible population
k Actual population

The following table provides the approximate population size for a collision for a given total population.

Total Population	Deployed Population With Collision Risk of .01%	1%
2^96	4T	42T
2^72	1B	10B
2^68	250M	2.5B
2^64	66M	663M
2^60	16M	160M

Acknowledgments

Dr. Gurtov is an adviser on Cybersecurity to the Swedish Civil Aviation Administration.

Quynh Dang of NIST gave considerable guidance on using Keccak and the NIST supporting documents. Joan Deamen of the Keccak team was especially helpful in many aspects of using Keccak.

Authors' Addresses

Robert Moskowitz
HTT Consulting
Oak Park, MI 48237
United States of America

Email: rgm@labs.htt-consult.com

Stuart W. Card
AX Enterprize, LLC
4947 Commercial Drive
Yorkville, NY 13495
United States of America

Email: stu.card@axenterprize.com

Adam Wiethuechter
AX Enterprize, LLC
4947 Commercial Drive
Yorkville, NY 13495
United States of America

Email: adam.wiethuechter@axenterprize.com

Andrei Gurtov
Linköping University
IDA
SE-58183 Linköping
Sweden

Email: gurtov@acm.org

drip Working Group
Internet-Draft
Intended status: Standards Track
Expires: 26 August 2021

A. Wiethuechter
S. Card
AX Enterprize, LLC
R. Moskowitz
HTT Consulting

22 February 2021

DRIP Registries
draft-wiethuechter-drip-registries-00

Abstract

TODO

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

TODO

2. Terminology

2.1. Required Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. Definitions

See [drip-requirements] for common DRIP terms.

HDA: Hierarchial HIT Domain Authority. The 16 bit field identifying the HIT Domain Authority under a RAA.

HID: Hierarchy ID. The 32 bit field providing the HIT Hierarchy ID.

RAA: Registered Assigning Authority. The 16 bit field identifying the Hierarchical HIT Assigning Authority.

3. Provisioning

Under DRIP UAS RID a special provisioning procedure is required to properly generate and distribute the certificates and attestations to all parties in the USS/UTM ecosystem using DRIP RID.

Keypairs are expected to be generated on the device hardware it will be used on. Due to hardware limitations (see Section 4) and connectivity it is acceptable under DRIP RID to generate keypairs for the Aircraft on Operator devices and later securely inject them into the Aircraft (as defined in Section 3.6.2). The methods to securely inject and store keypair information in a "secure element" of the Aircraft is out of scope of this document.

3.1. Overview of Transactions

In DRIP, each Operator MUST generate a Host Identity of the Operator (HIo) and derived Hierarchical HIT of the Operator (HHITo). These are registered with a Private Information Registry along with whatever Operator data (inc. PII) is required by the cognizant CAA and the registry. In response, the Operator will obtain a Certificate from the Registry, an Operator (Cro), signed with the Host Identity of the Registry private key (HIr(priv)) proving such registration.

An Operator may now add a UA.

* An Operator MUST generate a Host Identity of the Aircraft (HIa) and derived Hierarchical HIT of the Aircraft (HHITa)

* Create a Certificate from the Operator on the Aircraft (Coa) signed with the Host Identity of the Operator private key (HIo(priv)) to associate the UA with its Operator

* Register them with a Private Information Registry along with whatever UAS data is required by the cognizant CAA and the registry

* Obtain a Certificate from the Registry on the Operator and Aircraft ("Croa") signed with the HIr(priv) proving such registration

* And obtain a Certificate from the Registry on the Aircraft (Cra) signed with HIr(priv) proving UA registration in that specific registry while preserving Operator privacy.

The operator then MUST provision the UA with HIa, HIa(priv), HHITa and Cra.

* UA engaging in Broadcast RID MUST use HIa(priv) to sign Auth Messages and MUST periodically broadcast Cra.

* UAS engaging in Network RID MUST use HIa(priv) to sign Auth Messages.

* Observers MUST use HIa from received Cra to verify received Broadcast RID Auth messages.

* Observers without Internet connectivity MAY use Cra to identify the trust class of the UAS based on known registry vetting.

* Observers with Internet connectivity MAY use HHITa to perform lookups in the Public Information Registry and MAY then query the Private Information Registry which MUST enforce AAA policy on Operator PII and other sensitive information

3.2. HHIT Delegation

Under the FAA [NPRM], it is expecting that IDs for UAS are assigned by the UTM and are generally one-time use. The methods for this however are unspecified leaving two options.

1 The entity generates its own HHIT, discovering and using thr RAA and HDA for the target Registry. The method for discovering a Registry's RAA and HDA is out of scope here. This allows for the device to generate an HHIT to send to the Registry to be accepted (thus generating the required Host Identity Claim) or denied.

2 The entity sends to the Registry its HI for it to be hashed and result in the HHIT. The Registry would then either accept (returning the HHIT to the device) or deny this pairing.

In either case the Registry must decide on if the HI/HHIT pairing is valid. This in its simplest form is checking the current Registry for a collision on the HHIT.

Upon accepting a HI/HHIT pair the Registry MUST populate the required the DNS serving the HDA with the HIP RR and other relevant RR types (such as TXT and CERT). The Registry MUST also generate the appropriate Host Identity Claim for the given operation.

If the Registry denied the HI/HHIT pair, because there was a HHIT collision or any other reason, the Registry MUST signal back to the device being provisioned that a new HI needs to be generated.

3.3. Manufacturer

```
              +--------------+      Ca0a0 +-----------------+
              | Manufacturer | <--------> | Manufacturer CA |
              +--------------+ Ama0       +-----------------+
                 ^    |
                 |    |
                 |    |
         Ca0a0   |    |   Ama0
                 |    |
                 |    v
              +----------+
              | Aircraft |
              +----------+
```

During the initial configuration and production at the factory the Aircraft MUST be configured to have a serial number. ASTM defines this to be an ANSI/CTA-2063A. Under DRIP a HHIT can be encoded as such to be able to convert back and forth between them. This is out of scope for this document.

Under DRIP the Manufacturer SHOULD be using HHITs and have their own keypair and Cxx (Certificate: Manufacturer on Manufacturer). (Ed. Note: some words on aircraft keypair and certs here?).

Certificate: Aircraft 0 on Aircraft 0 (Ca0a0) is extracted by the manufacturer and send to their Certificate Authority (CA) to be verified and added. A resulting certificate (Attestation: Manufacturer on Aircraft 0) SHOULD be a DRIP Attestation in the Axy Form - however this could be a X.509 certificate binding the serial number to the manufacturer.

3.4. Registry

TODO

DRIP UAS RID defines two levels of hierarchy maintained by the Registration Assigning Authority (RAA) and HHIT Domain Authority (HDA). The authors anticipate that an RAA is owned and operated by a regional CAA (or a delegated party by an CAA in a specific airspace region) with HDAs being contracted out. As such a chain of trust for registries is required to ensure trustworthiness is not compromised. More information on the registries can be found in [hhit-registries].

Both the RAA and HDA generate their own keypairs and self-signed certificates (Certificate: RAA on RAA and Certificate: HDA on HDA respectively). The HDA sends to the RAA its self-signed certificate to be added into the RAA DNS.

The RAA confirms the certificate received is valid and that no HHIT collisions occur before added a HIP RR to its DNS for the new HDA. An Attestation: RAA on HDA is sent as a confirmation that provisioning was successful.

The HDA is now a valid "Registry" and uses its keypair and Certificate: HDA on HDA with all provisioning requests from downstream.

3.5. Operator

```
            +----------+            +---------+
            | Registry | ---------> | HDA DNS |
            +----------+  [HIP RR]  +---------+
               ^    |
               |    |
               |    |
         Coo   |    |   Aro
               |    |
               |    v
            +----------+
            | Operator |
            +----------+
```

The Operator generates a keypair and HHIT as specified in DRIP UAS RID. A self-signed certificate (Certificate: Operator on Operator) is generated and sent to the desired Registry (HDA). Other relevant information and possibly personally identifiable information needed may also be required to be sent to the Registry (all over a secure channel - the method of which is out of scope for this document).

The Registry cross checks any personally identifiable information as required. Certificate: Operator on Operator is verified (both using the expiration timestamp and signature). The HHIT is searched in the Registries database to confirm that no collision occurs. A new attestation is generated (Attestation: Registry on Operator) and sent securely back to the Operator. Optionally the HHIT/HI pairing can be added to the Registries DNS in to form of a HIP Resource Record (RR). Other RRs, such as CERT and TXT, may also be used to hold public information.

With the receipt of Attestation: Registry on Operator the provisioning of an Operator is complete.

3.6. Aircraft

3.6.1. Standard Provisioning

Under standard provisioning the Aircraft has its own connectivity to the Registry, the method which is out of scope for this document.

```
+----------+
| Registry |
+----------+
     ^
     |
     |
     |  Cro, CoaN
     |
     |
+----------+                        +----------+
| Operator | <--------------------- | Aircraft |
+----------+          Ca0aN         +----------+
```

Figure 1: Standard Provision: Step 1

Through mechanisms not specified in this document the Aircraft should have methods to instruct the Aircrafts onboard systems to generate a keypair and certificate. This certificate is chained to the factory provisioned certificate (Certificate: Aircraft 0 on Aircraft 0). This new attestation (Attestation: Aircraft 0 on Aircraft N) is securely extracted by the Operator.

With Attestation: Aircraft 0 on Aircraft N the sub certificate (Certificate: Aircraft N on Aircraft N) is used by the Operator to generate Attestation: Operator on Aircraft N. This along with Attestation: Registry on Operator is sent to the Registry.

```
+----------+
| Registry |
+----------+
     |
     |
     |
     |  Token
     |
     v
+----------+                        +----------+
| Operator | ---------------------> | Aircraft |
+----------+        Token           +----------+
```

Figure 2: Standard Provision: Step 2

On the Registry, Attestation: Registry on Operator is verified and used as confirmation that the Operator is already registered. Attestation: Operator on Aircraft N also undergoes a validation check and used to generate a token to return to the Operator to continue provisioning.

Upon receipt of this token, the Operator injects it into the Aircraft and its used to form a secure connection to the Registry. The Aircraft then sends Attestation: Manufacturer on Aircraft 0 and Attestation: Aircraft 0 to Aircraft N.

```
+---------+
| HDA DNS |
+---------+
    ^
    |
    | HIP RR
    |
    |
    |
+----------+ <-----------------------------+
| Registry |                               |
+----------+ ------------------------+     |
    |                                 |     |
    |                                 |     |  Token,
    |  CroaN                    CraN  |     |  Cma0, Ca0aN
    |                                 |     |
    |                                 |     |
    v                                 v     |
+----------+                      +----------+
| Operator |                      | Aircraft |
+----------+                      +----------+
```

Figure 3: Standard Provision: Step 3

The Registry uses Attestation: Manufacturer on Aircraft 0 (with an external database if supported) to confirm the validity of the Aircraft. Attestation: Aircraft 0 on Aircraft N is correlated with Attestation: Operator on Aircraft N and Attestation: Manufacturer on Aircraft 0 to see the chain of ownership. The new HHIT tied to Aircraft N is then checked for collisions in the HDA. With the information the Registry generates two certificates: Attestation: Registry on Operator on Aircraft N and Attestation: Registry on Aircraft N (Offline Form). A HIP RR (and other RR types as needed) are generated and inserted into the HDA.

Attestation: Registry on Operator on Aircraft N is sent via a secure channel back to the Operator to be stored. Attestation: Registry on

Aircraft N (Offline Form) is sent to the Aircraft to be used in Broadcast RID.

3.6.2. Operator Assisted Provisioning

This provisioning scheme is for when the Aircraft is unable to connect to the Registry itself or does not have the hardware required to generate keypairs and certificates.

```
+----------+
| Registry |
+----------+






+----------+                        +----------+
| Operator | ---------------------> | Aircraft |
+----------+       aN, CaNaN        +----------+
```

Figure 4: Operator Assisted Provision: Step 1

To start the Operator generates on behalf of the Aircraft a new keypair and Certificate: Aircraft N on Aircraft N. This keypair and certificate are injected into the Aircraft for it to generate Attestation: Aircraft 0 on Aircraft N. After injecting the keypair and certificate, the Operator MUST destroy all copies of the keypair.

```
+----------+
| Registry |
+----------+
    ^
    |
    |
    |  Cro, Cma0, Ca0aN, CoaN
    |
    |
+----------+                        +----------+
| Operator | <--------------------- | Aircraft |
+----------+        Cma0, Ca0aN     +----------+
```

Figure 5: Operator Assisted Provision: Step 2

Attestation: Manufacturer on Aircraft 0 and Attestation: Aircraft 0 on Aircraft N is extracted by the Operator and the following data items are sent to the Registry; Attestation: Registry on Operator, Attestation: Manufacturer on Aircraft 0, Attestation: Aircraft 0 on Aircraft N, Attestation: Operator on Aircraft N.

```
+----------+            +---------+
| Registry | ---------> | HDA DNS |
+----------+   HIP RR   +---------+
     |
     |
     |
     |  CroaN, CraN
     |
     v
+----------+                        +----------+
| Operator | ---------------------> | Aircraft |
+----------+          CraN          +----------+
```

Figure 6: Operator Assisted Provision: Step 3

On the Registry validation checks are done on all attestations as per the previous sections. Once complete then the Registry checks for a HHIT collision, adding to the HDA if clear and generates Attestation: Registry on Operator on Aircraft N and Attestation: Registry on Aircraft N (Offline Form). Both are sent back to the Operator.

The Operator securely inject Attestation: Registry on Aircraft N (Offline Form) and securely stores Attestation: Registry on Operator on Aircraft N.

3.6.3. Initial Provisioning

A special form of provisioning is used when the Aircraft is first sold to an Operator. Instead of generating a new keypair, the built in keypair and certificate done by the Manufacturer is used to provision and register the aircraft to the owner.

For this either Standard or Operator Assisted methods can be used.

4. Security Considerations

TODO

5. References

5.1. Normative References

[F3411-19] "Standard Specification for Remote ID and Tracking", February 2020.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

5.2. Informative References

[drip-requirements] Card, S., Wiethuechter, A., Moskowitz, R., and A. Gurtov, "Drone Remote Identification Protocol (DRIP) Requirements", Work in Progress, Internet-Draft, draft-ietf-drip-reqs-06, 1 November 2020, <http://www.ietf.org/internet-drafts/draft-ietf-drip-reqs-06.txt>.

[drip-rid] Moskowitz, R., Card, S., Wiethuechter, A., and A. Gurtov, "UAS Remote ID", Work in Progress, Internet-Draft, draft-ietf-drip-uas-rid-01, 9 September 2020, <http://www.ietf.org/internet-drafts/draft-ietf-drip-uas-rid-01.txt>.

[hhit-registries] Moskowitz, R., Card, S., and A. Wiethuechter, "Hierarchical HIT Registries", Work in Progress, Internet-Draft, draft-moskowitz-hip-hhit-registries-02, 9 March 2020, <http://www.ietf.org/internet-drafts/draft-moskowitz-hip-hhit-registries-02.txt>.

[NPRM] "Notice of Proposed Rule Making on Remote Identification of Unmanned Aircraft Systems", December 2019.

Authors' Addresses

Adam Wiethuechter
AX Enterprize, LLC
4947 Commercial Drive
Yorkville, NY 13495
United States of America

Email: adam.wiethuechter@axenterprize.com

Stuart Card
AX Enterprize, LLC
4947 Commercial Drive

Yorkville, NY 13495
United States of America

Email: stu.card@axenterprize.com

Robert Moskowitz
HTT Consulting
Oak Park, MI 48237
United States of America

Email: rgm@labs.htt-consult.com