CS 6600 Project Report: Unmanned Aircraft System

Matthew Whitesides and Bruce M. McMillin
Department of Computer Science
Missouri University of Science and Technology, Rolla, MO 65409-0350

Abstract—The abstract goes here.

1 Infrastructure

1.1 Infrastructure Description

In its simplest form, an unmanned aircraft system (UAS), is an aircraft system that operates without an onboard pilot. UAS are either controlled remotely through some form of wireless radio communication, semi-autonomously in conjunction with a remote pilot, or fully autonomously using some form of computational intelligence as navigation. These intelligent crewless vehicles have many potential uses in the defense sector, including surveillance, strategic mission execution, aerial sustainability support, and training systems, to name a few. These aircraft fall under the umbrella of cyber-physical systems (CPS), merging the intelligent navigation processes, system health monitoring, and communication with the physical mobile aerial vehicle.

We will design a proposed infrastructure security policy for the Boeing MQ-25 unmanned aircraft system, but it will also apply to similar mission support drones. The MQ-25 is an unmanned aircraft system designed for the U.S. Navy, and it provides autonomous refueling capability for the Boeing F/A-18 Super Hornet, Boeing EA-18G Growler, and Lockheed Martin F-35C fighters. This capability extends the combat range of the supported aircraft, seamlessly and semi-anonymously navigating to the plane, refueling, and returning to base. MQ-25 is the first unmanned aircraft to support aerial refueling another aircraft and is currently in the flight test phase of development [1], making it the perfect system to analyze security impacts for current and future unmanned aircraft systems.

1.1.1 Infrastructure Security Policy

Our infrastructure security policy breaks down the various actions a system user can perform using the following terms.

- *Subject*: Any entity that contains the proper rights can request the UAS perform operations, access objects, or grant rights to another subject.
- *Object*: An entity that is part of the UAS functionality or data that does not have control over another entity.
- *Rights*: A property assigned to a subject that defines its right to access an object or grant permissions to another subject.

Table 1 describes the rights and their associated functionality. Table 2 breaks down the subject roles involved in operating the UAS during a refueling mission. Table 3 contains the access control matrix (ACM) showing each Subject's rights over the objects.

TABLE 1
Description of rights over objects in the UAS.

Right	Description
Owns (O)	The owner of the given object.
Read (R)	Can observe the given object.
Write (W)	Can modify the given object.
Execute (E)	Can execute the functionality of the given object.
Grant (G)	Can grant a given right to another subject.
Control (C)	Can control a given system object.
Delete (C)	Can delete a given object or right.
Create (C)	Can create a new subject or object.

1.2 Rights Leakages

The following basic HRU commands show how given our current ACM, and we can leak integrity right r into a subject that does not contain the right in our initial ACM. We create a simple command sequence that attempts to grant one right from subject p in object o to subject q in o if p has the given right.

These leaks we will attempt to protect from can occur during a standard UAS refueling mission. The primary mission scenario will consist of a mid-air refueling of a F/A-18 on a long-range flight by our UAS. To simply our policy, we group the mission planning and overseeing phase to the *Mission Commander* role, the mission readiness to the *Quality Assurance Evaluators*, the piloting and in mission overseeing of the UAS to the *Pilot Commander and Instructor Pilot*, then the post-flight data analysis and maintenance to the *Flight Data Admin, Maintenance Crew*. From pre-planning to post-mission analysis, this entire mission process will be our example scenario to base our security policy around.

1.2.1 Confidentiality

For our example of a confidentiality attack, we simulate a scenario where our *External Contractor* is a bad actor seeking

TABLE 2
Description of actor subject roles during a UAS refueling mission.

Subject	Description
Pilot Commander (PC)	The primary remote pilot of the UAS during the mission.
Mission Commander (MC)	Responsible for final planning and decision making during the UAS mission.
Instructor Pilot (IP)	Assists the Pilot Commander and can pilot the UAS if given permission from the PC or MC.
Quality Assurance Evaluators (QA)	Responsible for pre-flight operational checks, in-flight analysis, and post-flight evaluation.
Maintenance Crew (MC)	Handles work orders created by the PC, IP, or FDA, responsible for the maintenance of the UAS.
External Contractor (Bad Actor) (EC)	Has a similar job to the MC however only has read rights to the FED.
Flight Data Admin (FDA)	Handles and analyses all mission flight data.

TABLE 3
UAS Refueling Mission Access Control Matrix

	PC	*MC*	*IP*	*QA*	*MC*	*FDA*	*EC*	**RNC**	**ANC**	**NP**	**RO**	**FED**	**RTD**
Pilot Commander (PC)	O,R,W	R,W	R,W	R,W	R,W	R,W	O,R,W,E,G,C	O,R,W,E,G,C	O,R,W,E,G,C	O,R,W,E,G,C	R,W,E,G,C	R,W,E,G,C	
Mission Commander (MC)	R	O,R,W	R,W	R	R	R	R,W,E,G,C	R,W,E,G,C	R,W,E,G,C	R,W,E,G,C	R,W,E,G,C	R,W,E,G,C	
Instructor Pilot (IP)	R	∅	O,R,W	∅	∅	∅	R,W,E,C	R,W,E,C	R,W,E,C	R,W,E,C	R,E	R,E	
Maintenance Crew (MC)	∅	∅	∅	∅	O,R,W	R	∅	∅	∅	∅	R,E	R,E	
Flight Data Admin (FDA)	∅	∅	∅	∅	R	O,R,W	∅	∅	∅	∅	O,R,W,E,G,C	O,R,W,E,G,C	
External Contractor (Bad Actor) (EC)	∅	∅	∅	∅	R	O,R,W	O,R,W	∅	∅	∅	∅	O,R,W,E,G,C	O,R,W,E,G,C
Remote Navigation Control (RNC)	∅	∅	∅	∅	∅	∅	R,W,E,C	R	R	R	R	R	
Autonomous Navigation Control (ANC)	∅	∅	∅	∅	∅	∅	R,W,E,C	R,W,E,C	R,W,E,C	R	R	R	
Navigation Planning (NP)	∅	∅	∅	∅	∅	∅	R	R	R	R	R	R	
Refueling Operation (RO)	∅	∅	∅	∅	∅	∅	∅	∅	R	R,W,E,C	R	R	
Flight Engine Data (FED)	∅	∅	∅	∅	∅	∅	∅	∅	∅	∅	R,W,E,C	∅	

TABLE 4
Read Flight Data ACM After Mission Executing Read Flight Data

	PC	*MC*	*IP*	*QA*	*MC*	*FDA*	*EC*	**RNC**	**ANC**	**NP**	**RO**	**FED**	**RTD**
Pilot Commander (PC)	O,R,W	R,W	R,W	R,W	R,W	R,W	O,R,W,E,G,C	O,R,W,E,G,C	O,R,W,E,G,C	O,R,W,E,G,C	R,W,E,G,C	R,W,E,G,C	
Mission Commander (MC)	R	O,R,W	R,W	R	R	R	R,W,E,G,C	R,W,E,G,C	R,W,E,G,C	R,W,E,G,C	R,W,E,G,C	R,W,E,G,C	
Instructor Pilot (IP)	R	∅	O,R,W	∅	∅	∅	R,W,E,C	R,W,E,C	R,W,E,C	R,W,E,C	R,E	R,E	
Maintenance Crew (MC)	∅	∅	∅	∅	O,R,W	R	∅	∅	∅	∅	R,E	R,E	
Flight Data Admin (FDA)	∅	∅	∅	∅	R	O,R,W	∅	∅	∅	∅	O,R,W,E,G,C	O,R,W,E,G,C	
External Contractor (Bad Actor) (EC)	∅	∅	∅	∅	R	O,R,W	O,R,W	∅	∅	∅	∅	O,R,W,E,G,C	O,R,W,E,G,C
Flight Engine Data (FED)	∅	∅	∅	∅	∅	∅	∅	∅	∅	∅	R,W,E,C	∅	

leaked rights beyond the initial ACM utilizing the following HRU commands available for the UAS.

The following command *Grant r Right* takes in a right *r*, the subject/object *o* the caller wants to grant rights over, the caller subject *p*, and the subject/object the caller is granting rights to *q*. This command verifies the caller has the right themselves over that subject/object to be able able to grant it to another subject, and has the *grant* right over that subject/object. This instills a basic principle of attenuation to our model.

```
command grant_r_right(r, o, p, q)
  if grant in A[p, o] and r in A[p, o]
  then
    enter r into A[q, o];
end
```

Next we have command *Make Owner* allowing a subject *p* to make another subject *q* the owner of a object *o* they currently have owner rights over.

```
command make_owner(p, q, o)
  if owns in A[p, o]
  then
    enter owns into A[q, o];
end
```

When the *PC* and *IP* return from a flight they or a *MC* will read the flight data and create a new object **Flight Record (FR)** holding the flight data, available to subjects who have **FED** access. After running this command our ACM would transition to a new state similar to table 4 which represents the subjects and objects involved in the read flight data procedure.

```
command create_flight_record(p)
  if create in A[p, FED]
  then
    create object FR;
    enter own into A[p, FR];
    enter read into A[p, FR];
    enter grant into A[p, FR];
end
```

```
grant_right_to_subject(Read, FED, PC, QA)
```

1.2.2 Integrity

For our integrity attack scenario, we will simulate a worst-case unauthorized modification to the *Autonomous Navigation Control (ANC)*. This scenario could occur if an unauthorized subject were granted the *write* right by an authorized subject. For example, if the *PC* mistakenly granted the right to a member of the *MC* to fix an issue.

```
grant_right_to_subject(Write, ANC, PC, MC)
```

REFERENCES

[1] A. Erwin, and J. Gibson, Navy, Boeing Make Aviation History with MQ-25 Becoming the First Unmanned Aircraft to Refuel Another Aircraft, Accessed on: Sept. 1, 2021. [Online]. Available: https://www.boeing.com/defense/mq25/

Matthew Whitesides Master's Student at Missouri University of Science and Technology.