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.

Add the bad actors into your Acm and give them appropriate initio/ rights

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 ?

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 will attempt to leak unauthorized post-mission flight data from the UAS to a *Quality Assurance Evaluators*. After the UAS has completed a flight, the *Maintenance Crew* downloads the flight

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 QA and FDA, responsible for the maintenance of the UAS.
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*	**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
Quality Assurance Evaluators (QA)	∅	∅	∅	O,R,W	R	R	∅	∅	∅	∅	∅	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
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	∅
Refueling Tank Data (RTD)	∅	∅	∅	∅	∅	∅	∅	∅	∅	∅	∅	R,W,E,C

data from the onboard UAS computer, then loads it into the base's central flight data system. This leak could occur either at the step of downloading the data from the UAS computer or after it is uploaded to the centralized system. Any subject with grant and *Flight Engine Data* rights could transfer these rights to another member causing a leak.

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

If we were to call this function with the subject *PC* granting right *Read* to *QA* for object *FED*, this would cause a confidentiality right leakage.

```
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.

Why would we give this information away if its a confidentiality leakage? Who is the "bad guy" we are trying to protect from?

How would it do this?

where is the unauthorized subject in your HRU?