REPÜLÉSTUDOMÁNYI KÖZLEMÉNYEK
31. évfolyam (2019) 3. szám 153–167. • DOI: 10.32560/rk.2019.3.265

Vránics Dávid Ferenc, Palik Mátyás

Mission as a Service – Egy felhőalapú UAS megvalósítása

A pilóta nélküli légi járművek – drónok – és a felhőalapú rendszerek napjainkra közismert, széles körben alkalmazott ágazatai a modern technológiának. A Mission as a Service – „szolgáltatásként nyújtott küldetés" már nem csak egy utópista jövőkép, a jelen technológiája lehetőséget ad UAV[1]-ok interneten át történő, akár tömeges távirányítására. Kutatásunk részét képezi egy felhőből irányított, pilóta nélküli légijármű-rendszer prototípusának kialakítása, funkcionális és biztonsági tesztek végrehajtása, illetve a munkálatok során szerzett tapasztalatok megosztása. Az alkalmazott technológiák között szerepel többek között az OpenStack felhőarchitektúra, a MapBox térképmegjelenítő keretrendszer, a MAVLink kommunikációs protokoll, a MAV Downlink mobilalkalmazás és az APM robotpilóta. Jelen publikáció az eddig elért eredményekbe ad betekintést, a rendszer jelenlegi állapotában az első éles repülésre készen áll.

Kulcsszavak: *UAS[2], drón, felhő, MAVLink, OpenStack*

Áttekintés

A kitűzött cél egy felhőalapú rendszerből irányított, pilóta nélküli légijármű-rendszer megvalósítása, nyílt technológiák lehető legnagyobb arányban történő felhasználásával.

Az elkészült prototípussal végrehajtható kísérletek eredményeként és a megvalósítás tapasztalatai alapján a későbbiekben ajánlásokat fogalmazhatunk meg a hasonló rendszerek megvalósítási, illetve tesztelési kérdéseiben, felhívva a figyelmet az esetleges biztonsági kockázatokra, illetve iránymutatást adva azok mérséklésére.

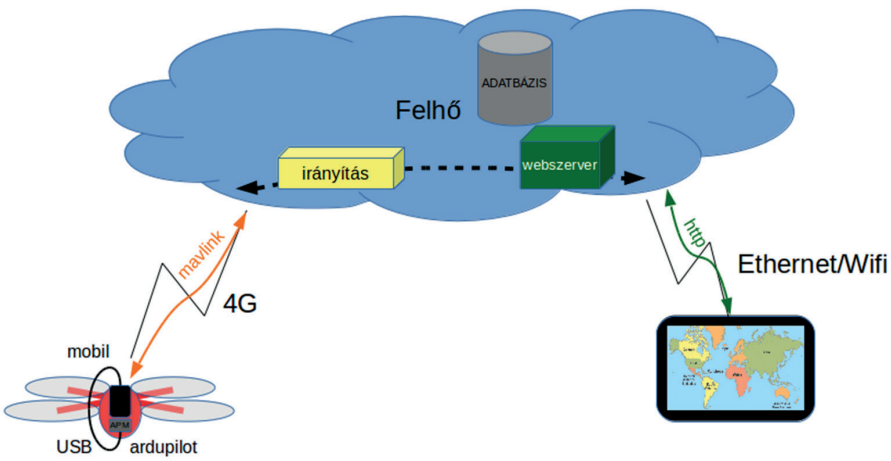
Jelen esetben az autonóm működésű, kereskedelmi és védelmi célú UAS-ek reprezentálása a cél. Ennek oka, hogy a hobbi UAV-k „feladatköre" miatt az egyszerű rádiós összeköttetést nem életszerű felhőalapú vezérléssel kiváltani. A hobbi UAV-k repülési adatainak felhőbe integrálása kézzel, a repülés végrehajtását megelőzően történhet meg, ahogy azt a jogszabálymódosítási tervezet [1] is előírja. Az eseti légtérigénylés és -aktiválás megkönnyítésére a HungaroControl által kifejlesztett mobilalkalmazás, a MyDroneSpace a hírek szerint hamarosan elérhető lesz [2].

[1] UAV – Unmanned Aerial Vehicle.
[2] UAS – Unmanned Aircraft System.

Az első szerző korábbi, témához kapcsolódó diplomamunkájában [3] az UAV-ok számára felhőalapon küldetéstervezést és -végrehajtást kiszolgáló rendszereket – a felhő számítási rendszerek szolgáltatási szintjeinek nevezéktanához illeszkedve – *Mission as a Service*-nek, „szolgáltatásként nyújtott küldetésnek" nevezte, helyénvalóbb ismert gyűjtőfogalom híján.

A prototípus rendszermagját egy OpenStack alapú felhőrendszer adja. Az UAV-k irányába TCP-alapú MAVLink szerver szolgálja ki a kapcsolatot, illetve menti adatbázisba a kapott adatokat, miközben az UAV felé küldött parancsokat is továbbítja.

A felhasználói oldalon egy térképes felületet szolgál ki, a HungaroControl Légtér.hu weboldalához hasonlóan MapBox keretrendszerre alapozva. A tényleges megjelenítendő adatokat a korábban említett adatbázisból nyeri a honlap, illetve a felhasználónak lehetősége van az UAV-ok számára küldetések megtervezésére/törlésére, a repülőeszközök élesítésére/biztosítására, fel/leszállítására, illetve autonóm vagy kézi irányítású módba kapcsolására.



1. ábra
A korábban tervezett architektúra sematikus ábrája [4]

A levegő-föld adatátvitel digitális (csomagkapcsolt) kapcsolattal valósul meg, bár a repülőeszköz a hagyományos, 2,4 GHz frekvenciájú távirányító egységgel is fel van szerelve, amellyel vészhelyzet esetén átvehető, felülírható az irányítás. A teszteszköz forgószárnyas, kvadrokopter felépítésű, bár elviekben a fedélzeti vezérlőre töltött firmware frissítésével merevszárnyas eszközzel is kompatibilissé tehető a rendszer.

2. ábra
A prototípus UAV [a szerzők szerkesztése]

A repülőeszköz alapját egy DJI F450 „Flame Wheel" jellegű keret adja. A kiegészítő lábak lehetővé teszik különböző kiegészítő függesztmények (szenzorok, nyomkövető jeladók, kamera) felszerelését a keret alá. A keret belsejében kap helyet az Ardupilot APM 2.6 típusú robotpilóta-egység és a 2,4 GHz másodlagos távirányító modul, illetve a nagy kapacitású lítiumakkumulátor. A keret tetején lett elhelyezve a GPS és külső iránytű modul, egy akkumulátorfeszültség-figyelő modul, illetve az elsődleges adatkapcsolatért felelős mobiltelefon. A mobiltelefon USB OTG-kábellel kapcsolódik az APM-hez, amelyen keresztül MAVLink protokoll segítségével tudja utasítani a robotpilótát, illetve adatfolyamokhoz fér hozzá a GPS, gyorsulásmérő, iránytű, giroszkóp és egyéb szenzorok irányából. A GPS-antenna, illetve külső iránytű modul egy dönthető árbocon kap helyet az elektromágneses interferencia kiküszöbölése végett.

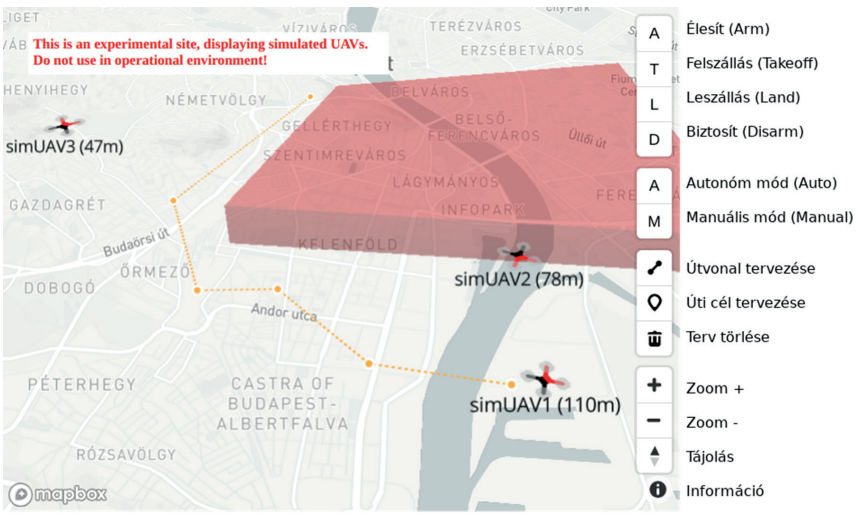
A teljes rendszer akkumulátorral és mobiltelefonnal együtt nagyjából másfél kilogramm, így kiegészítő függesztményeknek körülbelül fél kilogramm marad az UAV-ra kötött biztosításkonstrukció által maximált két kilogrammos felszállótömegből.

Első kérdésként felmerülhet a kézenfekvő kérdés, a mobilhálózat lefedettsége földfelszín felett. Egy, a 2019-es VFR Repülésbiztonsági Fórum rendezvényen elhangzott előadás [5] részben érinti a témát, az elhangzottak alapján sikeres méréseket végeztek 1000 lábig egy hasonló, de mobilhálózat-alapú helymeghatározással kísérletező projekt keretein belül. Az előadó nem hivatalos álláspontjában kimondja, hogy korábbi tapasztalatai alapján 4500 láb magasságban is képes volt mobilnetkapcsolaton át kommunikálni. Emellett megemlíti későbbi lehetőségét, hogy egy telefonos operátorral együttműködve „kinyittatnak" felfelé nagyjából egy tucat bázisállomást, ezzel az LTE-lefedettséget legalább FL660-ig kiterjesztik. Megjegyzi, hogy földközelben sokkal nagyobb a mobilhálózaton a rádióhullámok szórása, hiszen jóval több a zavaró jelforrás és tereptárgy, mint magasabban a légtérben.

A felhasználói oldalon a térképes megjelenítést MapBox keretrendszerrel valósítottuk meg. A felületen egyedi vezérlőelemek is helyet kaptak, az egyes akciók (például fel- és leszállás)

végrehajtására. Lehetőség van útvonalak rajzolására vagy úti cél kijelölésére. A megjelenés alapesetben felülnézetes 2D, jobb egérgombbal 3 dimenzióban is forgatható.

A vörös színű test egy no-fly zónát hivatott reprezentálni, egyelőre gyakorlati funkciója nincs, csak a megjelenítést teszteltük. Nagyobb mértékű zoom esetén az épületek is kiemelkednek a felszínből, segítve a tereptárgyak, akadályok áttekintését.



3. ábra
Képernyőkép a térképes felületről, egy tervezett útvonallal [a szerzők szerkesztése]

Földi alrendszer

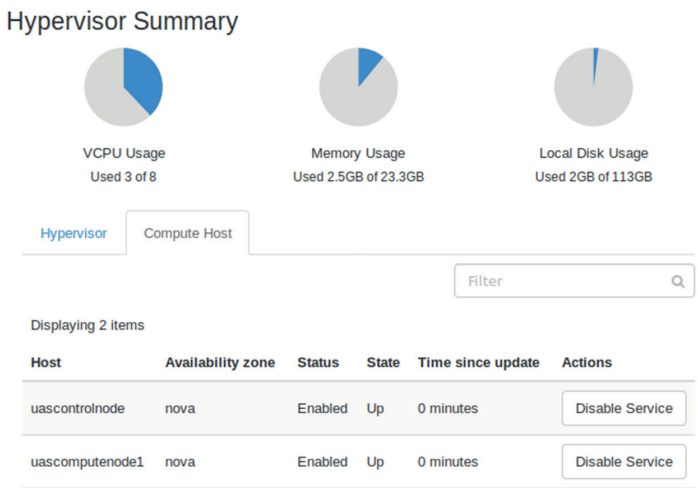
Felhő infrastruktúra

Felhő

Röviden összefoglalva, felhőalapú rendszereken több fizikai számítógépre virtualizált számítási teljesítményt vagy szolgáltatást értünk. A legismertebb ilyen szolgáltatások internetes tárhelyet, közösségimédia-oldalakat, videómegosztó oldalakat, keresőoldalakat szolgálnak ki. A koncepció lényegét talán ott lehet megfogni, hogy ezen oldalak felhasználóinak nem érdeke tudni, hogy a szolgáltatás hátterében mekkora vagy mennyi szerverszámítógép van, a fontos csak az, hogy éjjel-nappal elérhető legyen a szolgáltatás, és folyamatos legyen a felhasználói élmény (például ne akadozzon a megtekintett videó).

Felhőarchitektúrát telepíthetünk akár otthon a számítógépünkre, számítógépeinkre is, hiszen a virtualizációs technológiának köszönhetően a hardverplatform maga kevéssé lényeges. Jelen megvalósításban két különböző típusú notebook számítógépre lett kiterjesztve a felhő.

Így összességében 8 virtuális processzorszál és 24 GB memória áll rendelkezésünkre. Igény szerint bármikor több további számítógép csatlakoztatható a hálózathoz, illetve a rendszerhez.



4. ábra
Több számítógép, egy irányító és egy tisztán számítási feladatú elem a rendszerben [képernyőkép az OpenStack Horizon erőforrás-menedzsment felületéről, a szerzők szerkesztése]

Openstack

Az OpenStack az egyik legelterjedtebb nyílt forrású felhőmegoldás, amely különböző fantázianevekkel illetett „projektekből" épül fel [6]. Ezek közül a következőket alkalmaztuk a fejlesztés és telepítés során:

- Automatizáció („Heat")
- Portál („Horizon")
- Számítási erőforrások („Nova")
- Blokkos tároló („Cinder")
- Objektumorientált tároló („Swift")
- Hálózat („Neutron")
- Képfájlok („Glance")
- Azonosítás („Keystone")
- Monitorozás („Ceilometer")
- Munkamenet („Mistral")
- Metrika, statisztika („Gnocchi")
- Riasztás („Aodh")
- Terheléselosztás („Octavia")
- Eseménysor („Zaqar")

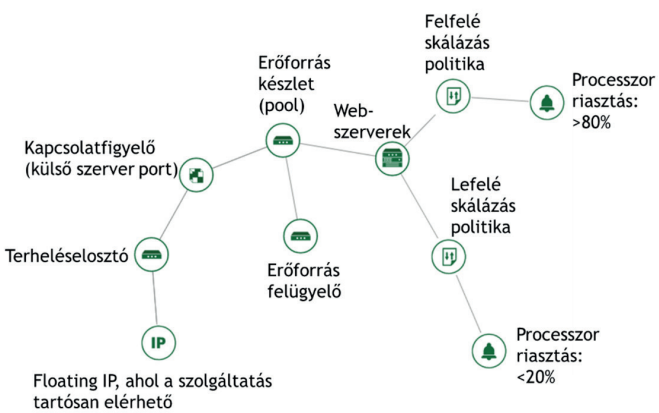
HA és LB

A felhőrendszerek egyik legkívánatosabb tulajdonsága a magas fokú rendelkezésre állás, vagy HA[3]. Ehhez szorosan kapcsolódó fogalom az LB,[4] vagyis terheléselosztás. A szolgáltatás külső hozzáférési pontjait valamilyen HA-proxymegoldás mögé szokás rejteni, ami alapjaiban egy robusztus átjárót jelent a külvilág felé. A proxy mögött több párhuzamosan dolgozó folyamat között osztjuk el a számítási feladatokat, amelyek képesek hatékonyan osztozni a közösen használt erőforrásokon. Ez a felhőben futó szolgáltatások tervezésénél az egyik legfontosabb szempont. Ezzel a módszerrel a rendelkezésre állást olyan magas szinten garantálni lehet, hogy általában a felhőkiszolgálók „ötkilences", vagyis 99,999% rendelkezésre állást vállalnak a szerződésekben. Ez évente mindössze 5 perc megengedhető szolgáltatás-kiesést jelent.

Jelen rendszerünkben egyik oldalon a térképes felületet és a hozzá tartozó REST API-t szolgáljuk ki, a másik oldalon pedig egy pymavlink alapú (Python MAVLink) szervert.

Önskálázás

A párhuzamos futásra optimalizált dolgozószálakat tetszés szerint indíthatjuk vagy leállíthatjuk a leterheltség függvényében. Ezt nevezzük horizontális skálázásnak. Jelen megvalósítást az OpenStack Heat orchesztrációs projektjét használva állítottuk össze. Egy előre elkészített sablonnal konfigurálhatjuk az erőforrásainkat. A térképes felület alatt futó, automatikusan skálázódó erőforrások logikai kapcsolata az 5. ábrán figyelhető meg.



5. ábra
Önskálázó OpenStack konfiguráció [a szerzők szerkesztése]

[3] HA – High Availability.
[4] LB – Load Balancing.

Első komolyabb kihívásunkkal a processzormetrikák elérése során találkoztunk: a legfrissebb Ceilometer-verziókban már nem támogatott a cpu_util metrika (és egyéb aggregált metrikák sem), ami a processzorhasználatot adná vissza százalékos formában. Kizárólag a processzoridő érhető el szigorúan monoton növekvő formában. Ezt a metrikát nem sikerült a Gnocchi-komponenssel reprodukálni, így egyelőre egy korábbi Ceilometer-verzióra álltunk vissza.

Önjavítás

Hasonlóan kívánatos tulajdonság lehet a párhuzamosan dolgozó virtuális gépek önjavítása, vagyis hiba esetén érzékeljék az incidenst, és indítsák újra a szolgáltatásukat „tiszta lappal". Erre is sikerült egy konfigurációt összeállítanunk. Viszont amikor ötvözni próbáltuk az önjavítást az önskálázással, versenyhelyzet alakult ki skálázás esetén: lefelé skálázáskor, amikor a fölöslegesen dolgozó virtuális gép törlődik, a törlés hatására a gép újraindítja magát, hiszen az önjavításnak ez lenne az egyik célja. Ennek következtében a skálázó újfent próbálkozik a virtuális gép felszabadításával, amire az újra megjavítja magát. A jelenlegi megvalósításban ezért egyszerű skálázást hagytunk meg, hiba esetén a többi gép terhelése megnő, és a következő kiértékelési időpontban (jelenleg 5 percenként) indul egy újabb példány. Sajnos a hibás virtuális gép a kézzel történő eltávolításig beragadt állapotban marad.

DevStack

Az OpenStack-es infrastruktúra telepítéséhez DevStack-et alkalmaztunk. Ennek sajnos limitációja, hogy a fizikai gépek újraindítása esetén a Neutron hálózatkezelő szolgáltatás nem képes újra helyesen elindulni. A két notebookot energiagazdálkodási okokból nem tartjuk 24/7 bekapcsolva, így a tesztelési időszakokban mindig újratelepítjük az egész OpenStack-környezetet, ez nagyjából gépenként 20 percet vesz igénybe.

További lehetőségek

Jelenleg az adatbázist MySQL szolgálja ki egy virtuális gépen. Ezt bevett gyakorlat szerint később érdemes lehet átmozgatni az OpenStack adatbázis szolgáltatásába („Trove"). Megfontolandó, hogy a későbbiekben SQL helyett a pozíció- és szenzoradatok naplózását noSQL-adatbázisba mozgassuk, idősoros adatok tárolására ugyanis ezek sok esetben jobb megoldást adnak az SQL-alapú adatbázisoknál.

Később dinamikus (járművek körüli vagy időjárásfüggő) és statikus no-fly[5] zónák (NFZ) kiszámítására is alkalmas lehet a rendszer, hiszen a bejelentkezett járművek adatai szinte valós időben jelen vannak a rendszerben, emellett a felhőben elméletileg végtelen számítási kapacitás[6] áll rendelkezésre. Ütközésveszélyes helyzet észlelése esetén a felhő automatikusan be is avatkozhat a küldetésbe, hiszen a bejelentkezett légi jármű közvetlenül fogad parancsokat

[5] Repülés elől elzárt.
[6] Hiszen bármennyi fizikai számítógép lehet a háttérben, mindig amennyi szükséges – elvileg.

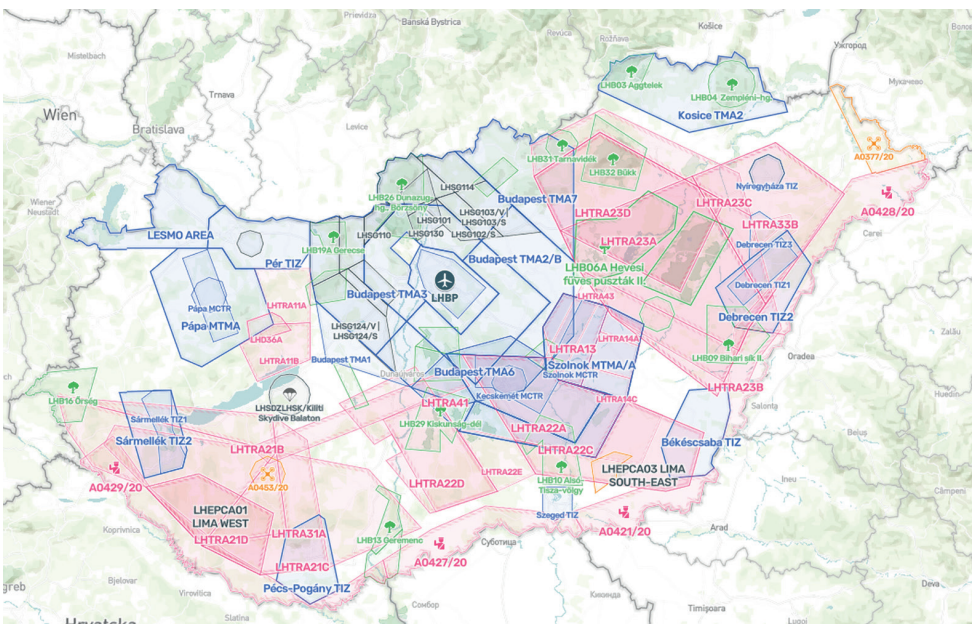
a felhőből. Jelvesztés esetén az UAV-k folytatják a küldetést a tervezett útvonalon haladva, így legtöbb esetben csak offline eszközök között lehetséges ütközéses baleset.

Jelenleg a szerver állandó jelleggel egy (bár felhőalapú, de csak egy virtuális gépből álló) interneten elérhető címen fut. A szerver konfigurációját első körben egyszerű szkript injektálással oldottuk meg, későbbiekben képfájlba telepítve vagy szoftverkonténer-technológiával lehet ezt továbbfejleszteni.

Térképes felület

Célja egy szemmel könnyen áttekinthető, térképes megjelenítés, amely egyéb repüléstervezést segítő rétegekkel is kiegészíthető (például no-fly zónák, időjárási előrejelzés, épületek), integrált küldetéstervezési funkcióval, akár több UAV együttes kijelölésével. Mindezt okostelefonon vagy számítógépen, platformfüggetlen módon.

A térképes felületet a MapBox-keretrendszerrel valósítottuk meg. Ennek oka, hogy a HungaroControl Légtér.hu térképes megjelenítője is erre alapul, esetleges keretrendszerbeli hibák feltárása esetén közvetlen hozzá tudunk járulni ennek a rendszernek a jobbá és biztonságosabbá tételéhez is.



6. ábra
Légtér.hu térképes felület [7]

Érdemes megjegyezni, hogy a MapBox legtöbb függvényhívása földrajzi hosszúság-szélesség párokkal dolgozik, amíg a MAVLink-protokoll szélesség-hosszúság párokkal, vagyis pont

fordított a sorrend. Ezek felcserélése esetén az UAV úti célja Magyarország helyett leginkább Szaúd-Arábiába fog esni.

Az UAV-k utolsó ismert pozíciójának térképes megjelenítése segít az eltűnt vagy lezuhant eszközök felkutatásában is.

További lehetőségek

A MapBox támogatást nyújt különböző AR[7]- és VR[8]-megoldások integrálására, akár amolyan virtuális terepasztal megvalósítására is (7. ábra).

7. ábra
Képernyőkép egy MapBox Unity alapú kiterjesztettvalóság-alkalmazásból [8]

A térképen tetszőleges számú réteg megjeleníthető, így akár ultrarövidtávú meteorológiai előrejelzésekkel is segíthetjük a küldetések tervezését ugyanazon a felületen.

[7] AR – Augmented Reality, kiterjesztett valóság.
[8] VR – Virtual Reality, virtuális valóság.

Légi alrendszer

A MAVLink-protokoll

A MAVLink bináris protokoll eredetileg kisméretű légi járművek rádiós kapcsolaton történő irányítására lett optimalizálva. Napjainkra kiegészítették szárazföldi és víz alatti járművek irányítására szolgáló funkciókkal is.

MAVLink 1.0

1. táblázat
A MAVLink 1.0 csomagszerkezete (a szerzők fordítása [9] alapján)

Bájtindex	Tartalom	Érték	Leírás
0	Csomag kezdete jelzés	0xFE	Protokollspecifikus jelzés. MAVLink rendszerek, amelyek nem ismerik ezt a verziót, eldobják a csomagot.
1	Hordozott adat hossza	0–255	A hordozott adat hosszát (n) jelöli.
2	Csomag sorszáma	0–255	Az elveszett csomagok detektálását segíti.
3	Rendszerazonosító	1–255	A küldő rendszerazonosítója a csatornán. A csatornán kommunikáló egyedi rendszerek megkülönböztetésére szolgál.
4	Komponensazonosító	0–255	A küldő komponensazonosítója a rendszeren. A rendszeren belüli komponensek megkülönböztetésére szolgál (például robotpilóta, kamera).
5	Üzenettípus	0–255	Az üzenettípus azonosítója. A hordozott adatstruktúra deszerializálását segíti.
6-tól (n+6)-ig	Hordozott adat		Az üzenettípus által meghatározott adatstruktúra.
(n+7)-től (n+8)-ig	Ellenőrző összeg		Az üzenetre vonatkozó X.25 CRC[9] (a csomag kezdete jelzést kivéve). CRC extra bájtot tartalmaz.

A rendszerazonosító (sysid) tartománya láthatóan elég szűkös, a kitüntetett nullás azonosító szórt üzeneteknek van fenntartva, a tartomány vége pedig általában a földi állomások azonosítására szolgál. Így egy kapcsolaton maximum 254 légi járművel tartható fenn kommunikáció, ez elég erős limitáció a regisztrált, pilóta nélküli járművek számára vonatkozóan. A komponensazonosító (compid) az egy UAV-n helyet kapó alrendszerek, például MAVLink-képes robotpilóta, kamera, gimbal vagy GPS azonosítására szolgál.

Az Amerikai Légierő berkein belül született egy dolgozat, ami a MAVLink 1.0 biztonsági hiányosságaira alapozva mutat be támadási módszereket [10].

A szerző a helyi és távoli hozzáférésre, közbeékelődéses támadásokra (lehallgatás, eltérítés) fókuszál, illetve szolgáltatásmegtagadás jellegű támadások lehetőségeire.

Egy másik, belga kutatócsapat fuzzing technikával keresett és talált hibákat a protokoll megvalósításában [11]. A cikk kiadásáig nem végeztek mélyebb behatolástesztelést a talált hibák mentén.

[9] CRC – Ciklikusredundancia-vizsgálat (Cyclic Redundancy Check).

MAVLink 2.0

2. táblázat
A MAVLink 2.0 csomagszerkezete (a szerzők fordítása [9] alapján)

Bájt index	Tartalom	Érték	Leírás
0	Csomag kezdete jelzés	0xFD	Protokollspecifikus jelzés. MAVLink rendszerek, amelyek nem ismerik ezt a verziót, eldobják a csomagot.
1	Hordozott adat hossza	0–255	A hordozott adat hosszát (n) jelöli.
2	Inkompatibilitásjelzők		Funkcionalitásjelzők, amelyeket a fogadó komponensnek kötelezően képesnek kell lennie értelmezni. A csomagot eldobja, ha a rendszer nem felel meg a feltételeknek.
3	Kompatibilitásjelzők		Funkcionalitásjelzők, amelyeket a fogadó komponens opcionálisan képes lehet értelmezni. A csomagot feldolgozhatja akkor is, ha a rendszer nem képes teljes mértékben értelmezni.
4	Csomag sorszáma	0–255	Az elveszett csomagok detektálását segíti.
5	Rendszerazonosító	1–255	A küldő rendszerazonosítója a csatornán. A csatornán kommunikáló egyedi rendszerek megkülönböztetésére szolgál.
6	Komponensazonosító	0–255	A küldő komponensazonosítója a rendszeren. A rendszeren belüli komponensek megkülönböztetésére szolgál (például robotpilóta, kamera).
7 to 9	Üzenettípus-azonosító	0–16777215	Az üzenettípus azonosítója. A hordozott adatstruktúra deszerializálását segíti.
10-től (n + 10)-ig	Hordozott adat		Az üzenettípus által meghatározott adatstruktúra.
(n + 11)-től (n + 12)-ig	Ellenőrző összeg		Az üzenetre vonatkozó X.25 CRC (a csomag kezdete jelzést kivéve). CRC extra bájtot tartalmaz.
(n + 12)-től (n + 26)-ig	Aláírás		(Opcionális) A csomagátvitel közbeni módosításának megelőzésére szolgál.

Kutatásunk szempontjából talán a legfontosabb különbség az új (opcionális) aláírásmező az üzenet végén.

3. táblázat
A MAVLink 2.0 csomagok aláírássorának felépítése (a szerzők fordítása [9] alapján)

Mező	Leírás
Kapcsolatazonosító (8 bit)	A kapcsolat azonosítója, amelyen a csomag közlekedik.
Időbélyeg (48 bit)	Időbélyeg 10 mikroszekundum osztással 2015. január 1-je óta. Szigorúan monoton növekvő üzenetenként egy kapcsolatra értelmezve. Ennek következménye, hogy az időbélyeg minimum 100 000 csomag/másodperc csomagküldési intenzitás esetén a valós idő elé (a jövőbe) torlódik. (A gyakorlatban általában jóval alacsonyabb az intenzitás.)
Aláírás (48 bit)	48 bitnyi SHA256 aláírás, a teljes csomagra, az időbélyegre és egy titkos kulcsra számítva.

Az aláírás tartalmaz egy szimmetrikus titkos kulcsot, amelynek a kulcscseréje titkosítatlanul történik. Ennek megfelelően a kulcscserét mindenképp megbízható kapcsolaton át kell végrehajtani a földi alrendszer és a robotpilóta-egység között. Ezt a hivatalos ajánlás szerint USB-kapcsolaton a legegyszerűbb kivitelezni [12].

Az aláírásmező bevezetésével a J. Marty által leírt támadások egy része kivédhető lett. Az irányítás eltérítése például a szekvenciaszám, időbélyeg és titkos kulcs megfelelő fabrikálása nélkül nem kivitelezhető.

Egy másik, amerikai kutatókból álló csapat ugyancsak vizsgálta a MAVLink 1.0 és 2.0 sebezhetőségeit [13]. A 2.0 verzió esetén a biztonságosabb kulcscserét, illetve az újrajátszásos támadások elleni védelmet emeli ki mint fejlesztendő pontot, illetve egy titkosított kapcsolat kialakítását.

A MAVLink-könyvtár

A protokoll használatát segítendő, C/C++ és Python programnyelven is elérhető nyílt forrású könyvtár a githubon.

A pymavlink Python könyvtár esetén limitáció, hogy csak egy szerver és kliens képes egy kapcsolaton kommunikálni, egyidőben (a két végpont a földi állomás és a légi jármű). A pymavlink forráskódjából kiindulva sikeresen újraírtuk a hálózati socket kezelés kódrészletet, hiszen jelenlegi formájában nem „felhőbarát" a működése. Létfontosságú, hogy egy hálózati porton több légi jármű is tudjon kommunikálni.

A 254 rendszerazonosító limitációja azonban még mindig fennáll. Ez részben mérsékelhető lenne azzal, ha az első bejelentkezéskor egy szabad rendszerazonosítót választanánk az UAV-nak, vagyis a jelenlegi statikus kiosztás helyett mindig újat kaphatna a légi jármű, amikor bejelentkezik, így párhuzamosan 254 légi jármű lehetne bejelentkezve.

MAV Downlink

A MAV Downlink egy Android okoseszköz-alkalmazás. Egy TCP-kapcsolat és az okostelefon USB soros kapcsolata között működik átjátszó szerepben. Az alkalmazás nyílt forrású, így szabadon továbbfejleszthető.

Az alkalmazás a TCP-kapcsolat megszakadása esetén automatikusan újracsatlakozik. Az alkalmazás megfelelően működik Ardupilot APM 2.6-tal, viszont Pixhawk PX4-el történő teszt során az alkalmazás nem ismerte fel a robotpilótát klienseszközként. Egy másik nyílt forrású alkalmazás, a QGroundControl viszont sikeresen felismerte a PX4-et, így valószínűleg elég a MAV Downlink kódjában frissíteni az USB soros kapcsolatért felelős könyvtárat.

APM 2.6

Az ArduPilot Mega 2.6 robotpilóta egy korábbi, mára már kifutott eszköz. Jelen UAS fejlesztésének kezdete 2015 környékére tehető, így történelmi okokból maradt meg a prototípus rendszerben. A 2.5-ös verzióval szemben ez a modul nem rendelkezik beépített iránytűvel, így külső iránytűmodul csatlakoztatása szükséges. Az élesítés előtti ellenőrzések kikapcsolásával lehetséges azonban iránytű nélkül is repülni, bár az inerciális tájékozódás sokkal pontatlanabb – az indításkor előre mutató irányt tekinti északnak a rendszer. Természetesen

távirányítóval történő, legfeljebb magasságtartást segítő repülési módokban nincs szükség az iránytűre.

Mivel az APM 2.6-os verzió már elavult, csak MAVLink 1.0 verziót támogat. A MAVLink 2.0-át 2017 körül vezették be. Emiatt csak a régebbi ArduCopter firmware-ekkel kompatibilis.

Mivel a MAVLink protokoll nyílt forrású, van lehetőség akár meteorológiai adatfolyamok natív leküldésére egy újabb verzióban, ehhez azonban a későbbiekben természetesen firmware-támogatást is meg kell valósítani a robotpilóta-eszközökben.

Összegzés

A cikkben bemutattuk egy felhőalapú, pilóta nélküli légijármű-rendszer prototípusát. A központi felhő-infrastruktúrából és a térképes megjelenítésből álló földi alrendszer, illetve a légi járműből, robotpilótából és okostelefon-alkalmazásból álló légi alrendszer. Részleteztük az egyes elemeket, amelyek az ehhez hasonló rendszerek különlegességeit adják: többek között az önskálázást, a valós idejű, akár 3 dimenziós térképes megjelenítést és küldetéstervezést, illetve -végrehajtást. Áttekintettük a MAVLink-protokoll fejlődését, korlátait és lehetőségeit. Mindeközben megemlítettünk néhány, a jelen rendszer fejlesztése során gyűjtött tapasztalatot.

A rendszer jelen állapotában készen áll az első repülési tesztek végrehajtására, a kezdeti, felszállás nélküli próbák (pozícióadatok elérése, motorok élesítése, majd biztosítása) biztató eredményeket hoztak.

Hivatkozások

[1] Magyarország Kormánya: A Kormány .../2016. (... ...) Korm. Rendelete az egyes légiközlekedéssel összefüggő kormányrendeletek módosításáról, Tervezet, 2016. [Online]. Elérhető: www.kormany.hu/download/8/db/e0000/RPAS__honlapra.pdf (Letöltve: 2019. 03. 31.)

[2] HungaroControl Netbriefing, „MyDroneSpace," *HungaroControl Netbriefing,* [Online]. Elérhető: www.netbriefing.hu/HU/mydronespace.html (Letöltve: 2019. 03. 31.)

[3] D. F. Vránics, „A felhő alapú rendszerekből vezérelt pilóta nélküli repülőgépek biztonsági kockázatai," Diplomamunka, Nemzeti Közszolgálati Egyetem, Hadtudományi és Honvédtisztképző Kar, Katonai Üzemeltető Intézet, 2015.

[4] D. F. Vránics, „Egy felhőalapú, pilóta nélküli légijármű-tesztrendszer bemutatása," *Bolyai Szemle,* 26. évf. 2. sz. pp. 37–44.

[5] Gy. Blazsovszky, „NetBriefing és MyDroneSpace," VFR Repülésbiztonsági Fórum, 2019, [Online]. Elérhető: www.youtube.com/watch?v=RD0gq0Ahngc (Letöltve: 2019. 03. 31.)

[6] G. Szabó, „Hálózati szolgáltatások OpenStack környezetben," Networkshop XXIII, 2014. április 23–25. [Online]. Elérhető: https://docplayer.hu/1587381-Halozati-szolgaltatasok-openstack-kornyezetben.html (Letöltve: 2019. 03. 31.)

[7] HungaroControl, „Légtér.hu," HungaroControl, [Online]. Elérhető: https://terkep.legter.hu (Letöltve: 2019. 03. 31.)

[8] Á. Debreczeni, „My bike ride in AR, (Unity + ARKit + Mapbox + Strava)," Twitter bejegyzés (@heyadam), 2017. június 7, 4:27 [Online]. Elérhető: https://twitter.com/heyadam/status/872278723700994048 (Letöltve: 2019. 03. 31.)
[9] MAVLink, „Serialization – MAVLink Developer Guide," *MAVlink,* [Online]. Elérhető: https://MAVLink.io/en/guide/serialization.html (Letöltve: 2019. 03. 31.)
[10] J. A. Marty, Vulnerability analysis of the MAVLINK protocol for command and control of unmanned aircraft, Diplomamunka, Wright-Patterson Air Force Base, Department of the Air Force Air University, Ohio, 2014. [Online]. Elérhető: https://apps.dtic.mil/dtic/tr/fulltext/u2/a598977.pdf (Letöltve: 2019. 03. 31.)
[11] K. Domin, E. Marin and I. Symeonidis, *Security Analysis of the Drone Communication Protocol: Fuzzing the MAVLink protocol. ESAT-COSIC and iMinds*. KU Leuven, Leuven-Heverlee, Belgium, 2016. [Online]. Elérhető: www.esat.kuleuven.be/cosic/publications/article-2667.pdf (Letöltve: 2019. 03. 31.)
[12] MAVLink, "Message Signing – MAVLink Developer Guide," [Online]. Elérhető: https://MAVLink.io/en/guide/message_signing.html (Letöltve: 2019. 03. 31.)
[13] N. Butcher, A. Stewart and S. Biaz, "Securing the MAVLink Communication Protocol for Unmanned Aircraft Systems," Technical Report #CSSE14-02, [Online]. Elérhető: https://pdfs.semanticscholar.org/4ce0/68b40089549f3d445d30e45fe8b53a141c88.pdf (Letöltve: 2019. 03. 31.)

MISSION AS A SERVICE – IMPLEMENTATION OF A CLOUD-BASED UAS

In our days, unmanned aerial vehicles – drones – and cloud computing systems are widely known branches of technology. The Mission as a Service is not just an utopian vision, the current technology enables us to control swarms of UAVs through the internet. Our research contains the development of a cloud-controlled prototype unmanned aircraft system, together with the necessary functional and security testing, and the publication of the observed results. Applied technologies include amongst others: OpenStack cloud architecture, MapBox map display framework, MAVLink communication protocol, MAV Downlink mobile app and the APM autopilot. The present publication describes the results achieved so far, the system in its current form is ready for the initial flight tests.

Keywords: *UAS, drone, cloud, MAVLink, OpenStack*

Vránics Dávid Ferenc MSc
Doktoranduszhallgató
Nemzeti Közszolgálati Egyetem
Hadtudományi és Honvédtisztképző Kar
Katonai Műszaki Doktori Iskola
vranicsd@gmail.com
https://orcid.org/0000-0003-0637-476X

Dávid Ferenc Vránics MSc
Doctoral Student
National University of Public Service
Faculty of Military Science and Officer Training
Doctoral School of Military Engineering
vranicsd@gmail.com
https://orcid.org/0000-0003-0637-476X

Palik Mátyás PhD
Intézetigazgató, egyetemi docens
Nemzeti Közszolgálati Egyetem
Hadtudományi és Honvédtisztképző Kar
Katonai Repülő Intézet
palik.matyas@uni-nke.hu
https://orcid.org/0000-0002-2304-372X

Mátyás Palik PhD
Director of Institute, Associate Professor
National University of Public Service
Faculty of Military Science and Officer Training
Institute of Military Aviation
palik.matyas@uni-nke.hu
https://orcid.org/0000-0002-2304-372X

VÁKÁT OLDAL