9-4-2022

# Distribution of DDS-cerberus Authenticated Facial Recognition Streams

Andrew T. Park
*Air Force Institute of Technology*

Nathaniel Peck
*Air Force Institute of Technology*

Richard Dill
*Air Force Institute of Technology*

Douglas D. Hodson
*Air Force Institute of Technology*

Michael R. Grimaila
*Air Force Institute of Technology*

*See next page for additional authors*



## Recommended Citation

Park, A.T., Peck, N., Dill, R. et al. Distribution of DDS-cerberus authenticated facial recognition streams. J Supercomput 79, 3471–3488 (2023). https://doi.org/10.1007/s11227-022-04771-2

The Journal of Supercomputing
https://doi.org/10.1007/s11227-022-04771-2

# Distribution of DDS-cerberus authenticated facial recognition streams

**Andrew T. Park**[1] · **Nathaniel Peck**[1] · **Richard Dill**[1] · **Douglas D. Hodson**[1] · **Michael R. Grimaila**[1] · **Wayne C. Henry**[1]

Accepted: 10 August 2022


**Abstract**
Successful missions in the field often rely upon communication technologies for tactics and coordination. One middleware used in securing these communication channels is Data Distribution Service (DDS) which employs a publish-subscribe model. However, researchers have found several security vulnerabilities in DDS implementations. DDS-Cerberus (DDS-C) is a security layer implemented into DDS to mitigate impersonation attacks using Kerberos authentication and ticketing. Even with the addition of DDS-C, the real-time message sending of DDS also needs to be upheld. This paper extends our previous work to analyze DDS-C's impact on performance in a use case implementation. The use case covers an artificial intelligence (AI) scenario that connects edge sensors across a commercial network. Specifically, it characterizes how DDS-C performs between unmanned aerial vehicles (UAV), the cloud, and video streams for facial recognition. The experiments send a set number of video frames over the network using DDS to be processed by AI and displayed on a screen. An evaluation of network traffic using DDS-C revealed that it was not statistically significant compared to DDS for the majority of the configuration runs. The results demonstrate that DDS-C provides security benefits without significantly hindering the overall performance.

**Keywords** Kerberos · DDS · Cyclone DDS · UAV · AI · QoS

## 1 Introduction

Networked sensor devices typically follow the paradigm where one node tasks and receives input from multiple Internet of Things (IoT) devices. For example, a command node sends operational messages and receives sensor data from

✉ Andrew T. Park
andrew.park075@gmail.com

Extended author information available on the last page of the article

Published online: 04 September 2022

multiple unmanned aerial vehicles to conduct a search and rescue operation. These messages, ranging from simple commands to video frames, could have Quality of Service (QoS) attributes such as retransmitting unreceived messages to ensure nodes that joined late or have re-started receive all messages. For example, a unmanned aerial vehicle (UAV) requires specific messages to navigate search and rescue missions correctly in lossy environments. Remotely operated bases use forward-deployed UAVs to support battlespace surveillance in contested environments [1]. The inter-communication links between UAVs and external links to cloud support services need to be robust enough to send and process video and images given terrain diversity and secure enough to thwart adversary attacks [2, 3]. Other messages could have QoS as best effort when a system can handle not receiving every message. For example, artificial intelligence (AI) facial recognition software on an IoT device may not require all frames from a live video feed to detect entities correctly. These use cases are essential in understanding DDS-Cerberus's (DDS-C) impact on real-world operations.

Data Distribution Service (DDS) is an open-source middleware that has been used in many sectors like finance, healthcare, and the military [4]. For real-time communication, DDS messages do not need to include the intended recipient but have a *topic*, represented as a unique string, from publisher to subscriber. The subscribers receive messages based on the associated *topic*. The messages have QoS properties to determine the sender and messages' behavior. Despite its efficient and real-time message sending capabilities, DDS is prone to impersonation attacks which allow an attacker to gain unauthorized access to messages [5–7].

DDS-C, a security layer for DDS, handles the authentication of DDS participants using Kerberos tickets [8–10]. This authentication mitigates impersonation attacks by verifying the identity of authenticated participants. This research's experiment captures network traffic from DDS and DDS-C to assess if DDS-C significantly impacts regular DDS performance. It uses the Bright Apps cloud architecture and network layout to evaluate DDS-C [11].

The experiment testbed relies on Cyclone DDS (an implementation of the DDS Standard) and the a commercial network infrastructure. The goal is to demonstrate that DDS-C is mature enough to support commercial artificial intelligence (AI) applications, specifically evaluating the impact on transmitting video frames. This impact is quantified by capturing total network traffic. The experiment emulates a network of UAVs with Raspberry Pi devices that send video frames. In conjunction with Bright Apps, this experiment aims to support UAV deployment in the field with DDS-C, such as in search and rescue. There are three use-cases detailing real-world scenarios for Bright Apps network infrastructure applications. To address the three use-cases, the experiment has one network configuration whose goal is to send video frames processed by AI over a Virtual Private Network (VPN). The network setup consists of a Raspberry Pi device, Elastic Compute Cloud (EC2), and laptop personal computer (PC). First, the Raspberry Pi device sends video frames to the EC2 for facial recognition AI processing, and then the PC displays the processed frames. The same message types of interest are selected. The QoS of interest is best effort to mimic use case scenarios. The

data collected are categorized by equipment to determine the traffic impact on each device.

This research builds on the previous paper, Park et al.'s *Quantifying DDS-Cerberus Network Control Overhead*, by collecting network packet quantities for facial recognition streams [10]. The collected traffic is split into three categories: *data message*, *security*, and *discovery+*. The research determines if DDS-C *security* traffic has a significant impact on the other DDS traffic by comparing the three through statistical analysis. This paper aims to contribute to other DDS research in use case applications.

This paper is organized as follows. Section 2 outlines DDS, DDS-C, and related works. Section 3 contains the experiment setup, assumptions and limitations, and results. Section 4 provides future research recommendations.

## 2 Background

This section provides background information on the design and implementation of DDS-Cerberus (DDS-C) by explaining Data Distribution Service (DDS) and Kerberos. Additionally, it presents other similar application works that support the development of this research's experiment.

### 2.1 DDS-cerberus (DDS-C)

DDS-C is a security layer that mitigates impersonation attacks [8–10]. It is integrated into DDS to provide participant authentication through Kerberos.

DDS is managed by Object Management Group (OMG) and is open-source, allowing for several implementations from different vendors. Its primary function is to handle message delivery between communicating entities. The communication is done through *topics*, or unique strings, that are sent by publishers and received by subscribers. Subscribers receive a message by specifying a unique string a message has. Quality of Service (QoS) policies dictate publisher and subscriber behavior on how to send messages. The policies are adapted to different network setups, such as having subscribers only read the most recent message.

The research focuses on the Data-Centric Publish-Subscribe (DCPS) layer containing the following components: publishers, subscribers, and *domain participants*. The components are seen in Fig. 1 where *domain participants* can contain any number of publishers and subscribers. The messages are sent with *topics* to the DDS domain to be read by subscribers. Previous DDS-C research focused on authenticating publisher and subscriber nodes, but this experiment focuses on authenticating *domain participants*. Two reasons to use *domain participants* are adding and authenticating nodes becomes less of a hassle, and they allow for easy integration into the Bright Apps architecture. *Domain participants* assist with executing publishers and subscribers in parallel, which helps send multiple video frames.

Kerberos is an authentication protocol used on distributed networks to authenticate users or nodes who request to talk on the network [12]. Kerberos servers have

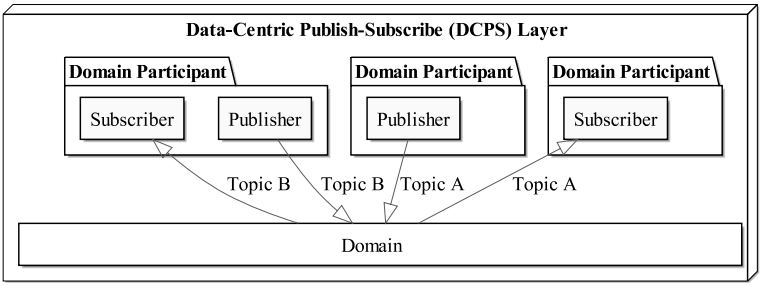


**Fig. 1** DCPS layout [8]

a realm name that is used to specify where the authentication is taking place. The |kinit| command grants tickets with the correct principal, or username, realm, and password. When the command runs, the requesting device or node communicates with Kerberos's Key Distribution Center (KDC), which has two main components: the Authentication Server (AS) and Ticket Granting Server (TGS). The requester authenticates with the AS if their credentials are in the server. If credentials match, the authenticated can gain a ticket from the TGS by sending a Ticket Granting Ticket (TGT) given by the AS. The lifetime of a ticket is default of 24 hours, but it is possible to change the lifetime to add ticket security.

One important function of Kerberos that DDS-C leverages are *keytabs*, long-term keys to aid in creating tickets. Each *domain participant* in DDS is paired with a unique *keytab* for seamless authentication. These *keytabs* are encrypted using AES-256. When running the |kinit| command, the password has to be manually entered; however, manually typing the password is not required if run with passing in the long-term key. This authentication is important in mitigating impersonation attacks because if the attacker does not have access to the *keytab*, DDS-C does not allow them to impersonate a node or *domain participant* [5–7]. For example, an attacker is able to get on the same network and deploys a rogue node to communicate with other nodes. However, DDS-C authenticates this rogue node by enforcing Kerberos authentication, and since the attacker does not have the correct credentials, it is not able to communicate on the network. The attacker would have to either replicate or steal the long-term key, which would be difficult due to the key's encryption and additional network security. There are multiple methods to enforce authentication to mitigate impersonation attacks. In this experiment, the *domain participants* individually handle their own authentication. Another variation would be to implement a DDS-C node to act as a firewall or switch facilitating authenticated nodes. The attacker would have to either replicate or steal the long-term key, which would be difficult due to the key's encryption and additional network security.

Figure 2 shows these *keytabs* in action with a sequence diagram of two *domain participants*, DP1 and DP2, authenticating with a KDC. The leftmost gray area, "Domain Participants utilizing KDC," represents the *keytabs* that were created and

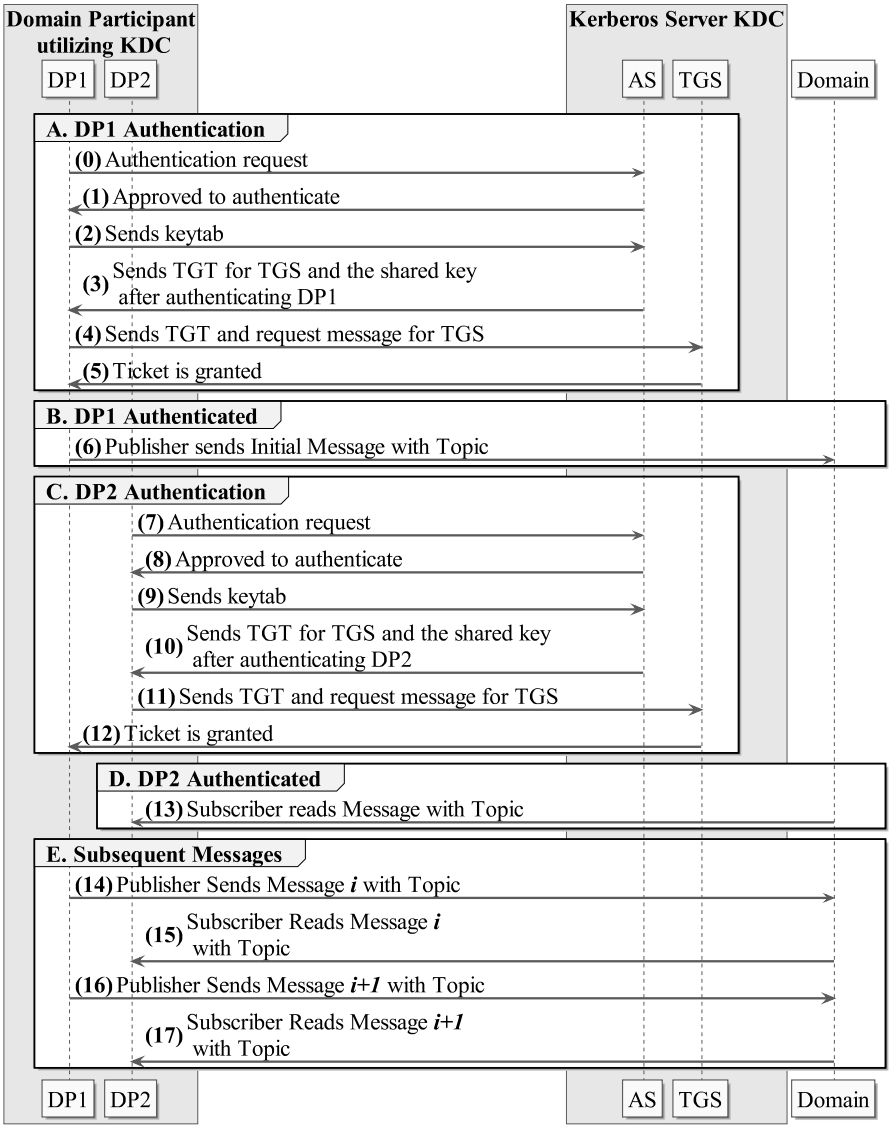


**Fig. 2** DDS-C authentication process with *domain participants* [10]

stored for DP1 and DP2. DP1 contains one publisher, and DP2 contains one subscriber. Messages flow as follows:

A. DP1 Authentication:

(0) DP1 requests to authenticate and starts the process to receive a ticket using a *keytab*. The AS receives DP1's request.

(1) The AS sends a message back that DP1 is able to authenticate.
(2) DP1 sends its *keytab* to the AS.
(3) The AS sends a TGT and its shared key for TGS after authenticating DP1. The shared key is used by the *domain participant* to encrypt messages for the TGS.
(4) DP1 sends the TGT and message request to the TGS to get a ticket.
(5) The TGS grants a ticket to DP1.

B. DP1 Authenticated:
(6) Afterward, DP1 is successfully authenticated, and the publisher can send its messages to the DDS domain.

C. DP2 Authentication:

(7) DP2 requests to authenticate and starts the process to receive a ticket using a *keytab*. The AS receives DP2's request.
(8) The AS sends a message back that DP2 is able to authenticate.
(9) DP2 sends its *keytab* to the AS.
(10) The AS sends a TGT and its shared key for TGS after authenticating DP2. The shared key is used by the *domain participant* to encrypt messages for the TGS.
(11) DP2 sends the TGT and message request to the TGS to get a ticket.
(12) The TGS grants a ticket to DP2.

D. DP2 Authenticated:

(13) Afterward, DP2 is successfully authenticated, and the subscriber can read messages. In this case, it would be reading data sent from DP1's publisher.

E. Subsequent Messages:

(14) Since DP1 and DP2 authenticated, no further authentication is needed. DP1's publisher sends Message $i$ with *Topic*.
(15) DP2's subscriber receives the Message $i$.
(16) DP1's publisher sends Message $i + 1$ with *Topic*.
(17) DP2's subscriber receives the Message $i + 1$.

DDS-C authentication executes at the beginning of a *domain participant's* lifecycle; however, this authentication can run more than once based on an administrator's needs. Additionally, this can be performed in conjunction with shorter ticket lifespans. This research does not integrate these scenarios with the experiment and is possibly integrated into future work.

### 2.2 Related use case applications

DDS-C's application and use cases are inspired by search and rescue and battlefield operations. Many papers provide solutions to these complex problems, but this research focuses on those that offer solutions using unmanned aerial vehicles (UAV). Understanding the other researchers' proposed designs and experiments helps craft the experiment use cases and real-world application.

The first paper to inspire the experiment design was Munir et al.'s research on proposing FogSurv, a fog-assisted architecture to be used in urban areas for real-time surveillance using artificial intelligence (AI) [13]. They constructed a centralized cloud server with fog nodes to offload communication and computation power burdens. Their use cases mention battlefield applications for security and control, which this paper's research does on a more specific scale with DDS-C. The experiment has two scenarios with an Internet of Things (IoT) device where in the first scenario, it offloads tasks to a fog node and the second where it offloads it to the cloud server. They measure latency in different experiment runs with data fusion and AI. The results revealed that offloading to a fog node is 37% more efficient than to the cloud. The use cases and design for broad low-power surveillance helped motivate DDS-C use case research.

In 2017, Ribeiro et al. conducted simulated and physical experiments with UAVs and DDS [14]. They also used a cloud architecture but focused on using a DDS communication infrastructure. They designed a two-layer UAV network for UAVs closer to the ground and those far away from the ground. They selected sensors ranging from those that work near the ground to those far away. The types of sensors on the UAV categorized what layer it would operate in. The simulated experiments tested different network links for low bandwidth and lossy environments in wired and wirelessly configurations. They tested with both QoS best effort and reliable and found more throughput with reliable QoS acknowledgments. The physical experiment only utilized one UAV with ROS (Robot Operating System) for DDS communication with a base station on the cloud. They observed high signal attenuation and loss of connectivity since they used default DDS QoS policies. For future work, they plan to extend the experiment to four UAVs.

ROS is a middleware with two versions: ROS 1 and ROS 2. The difference is that ROS 2 uses DDS for real-time communication. In 2019, Sandoval and Thulasiraman's research goal was to use simulated experiments to test ROS 2's ability to protect against cyber attacks for UAV communication [15]. This work was to help support the integration of ROS 2 into the US Navy's UAV swarms. Since ROS 1 was still in use for Naval UAV control, they simulated an environment where ROS 1 and ROS 2 were connected with a bridge to control three UAVs. The first two UAVs were susceptible to rogue node attacks, unwanted disabling and landing, due to ROS 1, but the third UAV used the bridge with ROS 2 and its security plugins to prevent these attacks. Even though the plugins mitigated the attacks, there was significant latency overhead due to the bridge setup. This work contributes to DDS-C by highlighting the need for node authentication when controlling UAVs.

Table 1 lists the three related works that relate to DDS-C design and experiments regarding AI, network environments, and security. The following experiment combines these three elements to measure DDS-C's performance in a cloud-based network.

**Table 1** Use case applications

| Paper | Description |
|---|---|
| Munir et al. [13] | Utilizing fog-assisted architecture for real-time surveillance. |
| Ribeiro et al. [14] | Evaluated DDS communications with using a two-layer UAV network. |
| Sandoval and Thulasiraman [15] | Simulating cyber attacks against UAVs over ROS 1 and ROS 2. |

## 3 Experiments

The goal of the experiment is to analyze and determine if DDS-Cerberus's (DDS-C) addition to Data Distribution Service (DDS) operations significantly impacts the real-time message sending performance of DDS in addition to the artificial intelligence (AI) processing. The experiment mimics use case scenarios such as search and rescue where DDS-C, DDS, and AI would benefit security, processing, and performance. These experiments are performed in conjunction with Bright Apps, aiming to integrate their Azoth AI with these particular missions. This section goes over the experiment details, apparatus, assumptions and limitations, and results.

### 3.1 Experiment details

Bright Apps developed Azoth AI with unmanned aerial vehicles (UAV) for facial recognition in real-world use cases like search and rescue [11]. Azoth AI provides AI and machine learning (ML) capabilities while disconnected from networks in degraded environments using low computing power resources which is applicable to this experiment's use of Raspberry Pi devices. It uses existing OpenCV capabilities for ubiquitous object recognition. The long-term goal is to create and train a neural network for more specific image recognition. It is paired with Cyclone DDS, a variation of DDS developed by the Eclipse Foundation, to send live video frames in lossy environments to be processed by Azoth AI [16, 17]. Cyclone DDS is related to ROS 2 (Robot Operating System) because it is a tier-1 ROS 2 Middleware Interface (RMW). It uses a python binding which helps integrate DDS-C and the AI [18]. Bright Apps uses UAVs connected to and controlled by Raspberry Pi devices. These devices communicate with Amazon Web Services (AWS) Elastic Compute Cloud (EC2) instances for facial recognition processing by Azoth AI [19]. This configuration is to mimic operations where UAVs send live video feeds. OpenVPN connects this framework by providing additional security and network maintenance [20]. DDS-C authenticates *domain participants* to allow for multiple node executions. Integrating it with Bright Apps technology is still in development, and this research presents initial work in this integration with a real-world commercial network infrastructure. The experiment's results aim to support this paper's previous experiment results and if DDS-C authentication traffic adds negligible latency overhead to affect normal DDS message traffic significantly.

**Table 2** Equipment specifications

| | Raspberry Pi: Cyclone DDS | Laptop PC: Cyclone DDS, KDC | EC2: Cyclone DDS, KDC |
|---|---|---|---|
| Name | *Cyclone1* | *Cyclone2* | *KerAzoth* |
| Machine | Raspberry Pi 4B | XPS 13 9310 | t3.2xlarge [21] |
| OS | Ubuntu 20.04.3 LTS | Ubuntu 20.04.3 LTS | Ubuntu 20.04.3 LTS |
| CPU | ARM Cortex-A72 | 11th Gen i7-1185G7 | Intel Xeon E5-2676 v3 |
| Disk Space | 64 GB | 2 TB | 58 GB |
| RAM | 8 GB | 31 GB | 32 GB |

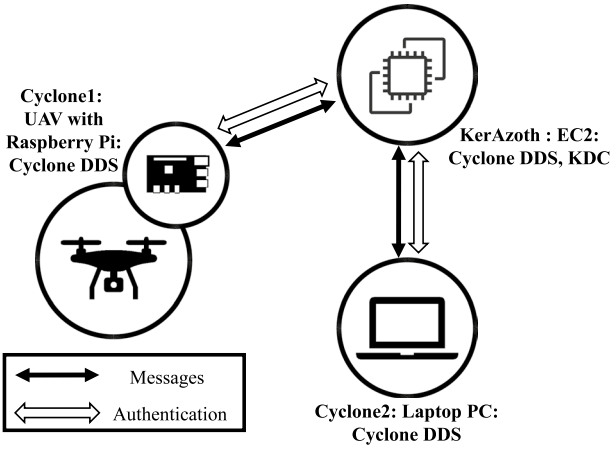


**Fig. 3** Experiment Network Diagram

### 3.2 Experiment apparatus

The experiment testbed for DDS-C uses Cyclone DDS and Kerberos. There are three pieces of apparatus-one Raspberry Pi 4B device, a Dell XPS 13 laptop personal computer (PC), and one EC2 instance. These labels distinguish the three pieces of equipment: Cyclone1, Cyclone2, and KerAzoth. Table 2 lists the main equipment and its specifications. All devices have Kerberos installed. Additionally, to communicate with each other, Cyclone1 and Cyclone2 are OpenVPN clients, and KerAzoth is the OpenVPN server; all traffic routes through KerAzoth from the other two. KerAzoth is located in the AWS region code us-west-2a within Oregon. Figure 3 is this equipment's testbed network diagram. All components are connected wirelessly through OpenVPN and use Cyclone DDS to communicate.

Cyclone1's *domain participants* authenticate with KerAzoth's Key Distribution Center (KDC) before sending messages. Similarly, KerAzoth's *domain*

*participants* authenticate with Cyclone2's KDC. Cyclone1 represents the UAV with Raspberry Pi device, Cyclone2 represents command and control (C2), and KerAzoth represents a network bridge and AI processing.

This experiment covers three main use cases that encompass the apparatus used in the commercial network infrastructure.

- ***Use case 1***: Perform DDS-C authentication on a Raspberry Pi device and EC2, and after authentication, both devices communicate using Cyclone DDS. This tests communication from a Raspberry Pi device to an EC2 on the cloud.
- ***Use case 2***: Authenticate using DDS-C over a Virtual Private Network (VPN). OpenVPN clients utilize unique credentials to communicate with the OpenVPN server. This tests communication using a VPN between devices and the cloud.
- ***Use case 3***: Send a video feed to be processed by AI for face recognition. The video feed is sent over DDS with compressed video frames. These frames are sent with best effort reliability Quality of Service (QoS) to handle lossy environments. This tests sending video frames over DDS for AI processing.

There is one network configuration to encompass these three use cases. It is run with QoS best effort to mimic real-world UAV use when sending video frames. Only one UAV is used to set a baseline for future experimentation with more UAVs.

- ***Cyclone with KerAzoth***: All *domain participants* authenticate either with Cyclone2 and KerAzoth when starting up. Cyclone1 sends compressed video frames to KerAzoth for AI processing, and when finished, KerAzoth sends the processed frames to Cyclone2.

The *domain participants* authenticate using Kerberos through enforcement of DDS-C. Authentication prevents impersonation attacks by adding an extra security layer that the *domain participants* use outside of DDS. They also authenticate the publishers and subscribers they have. After the *domain participants* receive their ticket, they can send messages. The administrator can configure the ticket's time-to-live to determine if the *domain participants* need to be re-authenticated.

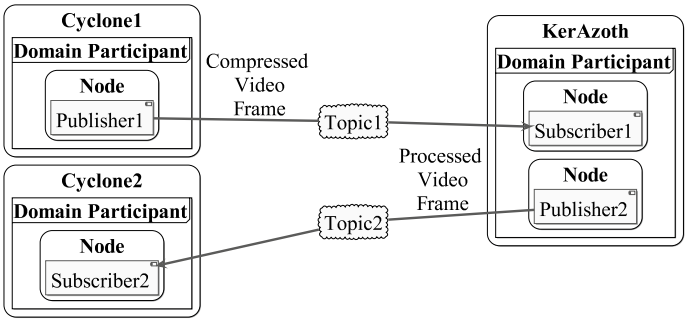


**Fig. 4** Experiment node layout

**Table 3** Experiment participants

| | | Set 1 | Set 2 | Set 3 | Set 4 | Set 5 | Set 6 |
|---|---|---|---|---|---|---|---|
| Cyclone1 | *Domain participants* | 1 | 1 | 1 | 1 | 1 | 1 |
| | Nodes (Pub/Sub) | 1 | 2 | 3 | 4 | 5 | 6 |
| Cyclone2 | *Domain participants* | 1 | 1 | 1 | 1 | 1 | 1 |
| | Nodes (Pub/Sub) | 1 | 1 | 1 | 1 | 1 | 1 |
| KerAzoth | *Domain participants* | 1 | 2 | 3 | 4 | 5 | 6 |
| | Nodes (Pub/Sub) | 2 | 4 | 6 | 8 | 10 | 12 |

The scalability goal of Cyclone with KerAzoth is to increment the number of *domain participants* in KerAzoth to increase DDS-C authentication traffic and highlight the use of Azoth AI. Figure 4 and Table 3 illustrate how the configuration participants are set up and how the frames are passed. The number of participants was chosen based on the network diagram with one UAV as an initial proof of concept and execution of the use cases. Future experiments can extend the participant count to greater numbers to further determine DDS-C's impact. Figure 4 illustrates Set 1 from the table. Cyclone1's *domain participant* has one publisher in the figure but increases, as seen in the table, by one publisher as each set is tested to handle video frame publishing. KerAzoth's publisher and subscriber node count also increases based on the experimented set. KerAzoth has one publisher to one subscriber increasing with subsequent runs: 1:1, 2:2, 3:3, 4:4, 5:5, 6:6. Each of these ratios has a unique *domain participant*. Cyclone2's one subscriber with one *domain participant* does not increase in number, and it receives all AI processed video frames to display on the laptop PC screen.

Cyclone1's publishers send 100 messages with frames as data to Cyclone2. The frames are sent with the Real-Time Publish-Subscribe (RTPS) protocol and are not encrypted to allow for constant User Datagram Protocol (UDP) connections. Although encryption is not enabled to focus on this experiment's concept, performance, and function, future experiments can enable DDS Security to utilize encryption such as AES and Diffie-Hellman key exchange [22]. Figure 4's Publisher1 sends a frame with a *topic*, and as more publishers are added to Cyclone1, they send different frames with unique *topics*. These frames are compressed and then fragmented when sent over the network. The subscribers in KerAzoth receive these *topics*. Subscriber1 receives Topic1 and applies facial recognition AI to the frame by using rectangles to identify any faces. Other subscribers would be waiting for their respective *topics*. Afterward, Publisher2 sends the processed video frame with Topic2 to Subscriber2. In this case, the publishers in KerAzoth send the processed video frame with a *topic* that only Cyclone2's subscriber uses.

To incorporate the AI, the configuration was designed to parallelize AI processing with *domain participants*. With the number of messages set to 100 for every experiment iteration, the messages were divided up so each publisher in Cyclone1 sends a unique frame to the subscriber in Cyclone2. For example, Table 3's Set 2 has two publishers in Cyclone1's *domain participant*. One publisher would publish only

odd frames and the other even frames. For subsequent Sets, the frames are divided by every third or every fourth frame for the newly added publishers. This parallel processing is an important aspect of this network configuration to showcase Azoth AI while also increasing participant count for DDS-C authentication.

On all three pieces of equipment, tcpdump captured the experiment's data and was sent to a separate Windows machine to be processed [23]. The data are first run with Windows Powershell scripts involving tshark, a Wireshark filtering tool [24, 25]. Afterward, the Student's t-test is used to calculate the p-value to analyze the results with a $\alpha$ of 0.05 to conclude if the null hypothesis is rejected. In these experiments, since the population variance is unknown it is appropriate to use the Student's t-test for the p-values. The tools used are Python and SciPy [26, 27]. Since the population variance is unknown, this test is applicable to the population of DDS use cases discussed in this paper. Table 4 lists all the software mentioned for the experiment.

### 3.3 Assumptions and limitations

These are the experiment's assumptions and limitations to execute the specified network setup. The assumptions are as follows:

- *Domain participants* do not fail authentication and that an attacker does not compromise them.
- Cyclone1's publishers send all 100 video frames, and KerAzoth's subscribers receive the specified messages.
- *Keytabs* were not renewed or changed between experiment runs.
- Relevant packet protocols such as RTPS and Kerberos (KRB5) were selected. Protocols such as Simple Service Discovery Protocol (SSDP) were excluded because they did not contribute to any of the three traffic categories in analyzing DDS-C.

**Table 4** Software information

| Name | Version | Location |
|---|---|---|
| Cyclone DDS | 0.8.1 | Cyclone1, Cyclone2, KerAzoth |
| OpenVPN | 2.4.7 | Cyclone1, Cyclone2, KerAzoth |
| Kerberos | V5 | Cyclone1, Cyclone2, KerAzoth |
| tcpdump | 4.9.3 | Cyclone1, Cyclone2, KerAzoth |
| tshark | 3.4.7 | Windows |
| PowerShell | 5.1.19041.1237 | Windows |
| Python | 3.9.7 | Cyclone1, Cyclone2, KerAzoth, Windows |
| SciPy | 1.7.0 | Windows |

- All RTPS packets without the video frame payload were categorized as *discovery+*.
- Azoth AI detected only one human face during experimentation. Other experiments may incorporate more faces to analyze the AI's processing load on the EC2.

The limitations are as follows:

- The experiment is only performed with the us-west-2a zone. If other zones were used, the experiment may differ with a lossier environment.
- RTPS messages containing video frames were fragmented, resulting in more packet traffic.
- The experiment only experimented with reliability QoS of best effort. Best effort fits the use cases; however, future experimentation could include other QoS policies.
- Collecting the total packet traffic is only one factor in determining DDS-C's impact on DDS. Other factors could include latency and location of equipment. The appartus in this experiment were in the same immediate area while the EC2 was not.
- The default discovery protocols were used. The *discovery+* traffic could differ if other discovery protocols were invoked.

### 3.4 Experiment results

The data are organized based on the three used equipment as seen in Fig. 5 for Cyclone1 and Cyclone2 and Fig. 6 for KerAzoth. The figures use the three data categories: *data message*, *security*, and *discovery+* traffic. The figures' independent variable uses the total number of *domain participants* for each participant set in Table 3, and the dependent variable is based on the traffic amount for each data category. The *security* traffic bytes in the experiment are indistinguishable compared to the greater *data message* and *discovery+* traffic.

Table 5 shows the p-values for all three equipment. Overall, the quantity of participants was not statistically significant; however, for seven participants, two cases were statistically significant for Cyclone2 and KerAzoth. The network and

**Table 5** Configuration *p*-values

| Participants | Cyclone1 | Cyclone2 | KerAzoth |
|---|---|---|---|
| 3 | 0.911 | 0.888 | 0.785 |
| 4 | 0.9 | 0.77 | 0.751 |
| 5 | 0.09 | 0.114 | 0.8 |
| 6 | 0.946 | 0.905 | 0.899 |
| 7 | 0.315 | 0.002[1] | 0.003[1] |
| 8 | 0.234 | 0.82 | 0.8 |

[1]Statistically significant *p*-values with $\alpha$ 0.05

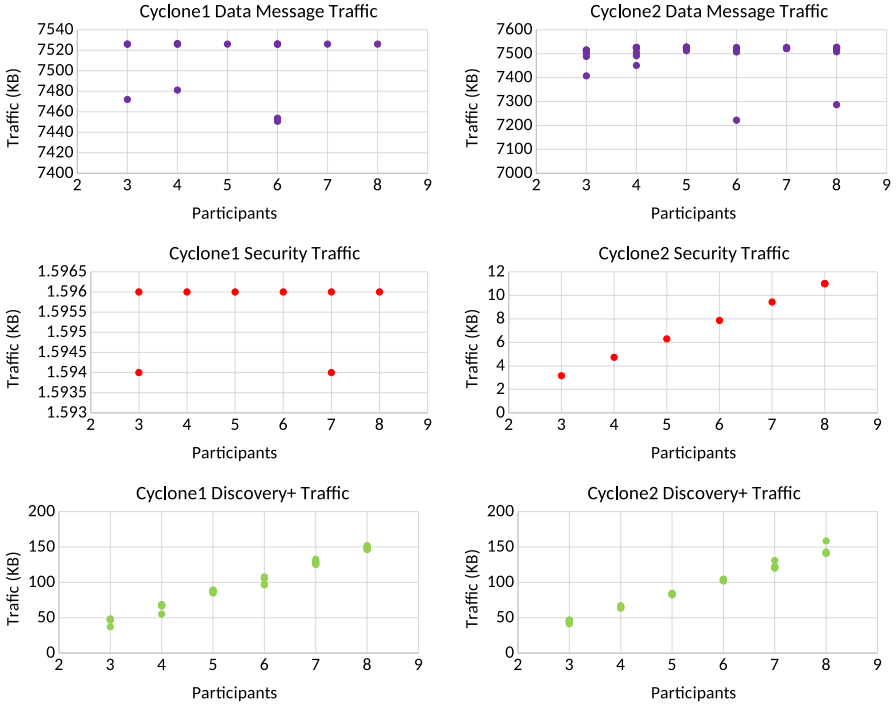


**Fig. 5** Experiment results. Left column: Cyclone1. Right column: Cyclone2

experiment setup could have influenced this situation. The experiment setup and best effort QoS use resulted in a more lossy environment and no packet retransmissions. The use of an EC2 brings possible unreliability with its network. With the addition of using a VPN, the sent video frames could be lost over the network. Cyclone1 did not have a statistically significant *p*-value because it is the starting point for all message traffic by sending captured video frames; only the components receiving the messages were affected. Even though the p-values were statistically significant, the results reveal that choosing the correct network and equipment setup is important in ensuring all components function as intended. Also, the other participant p-values show that this significance is uncommon and that DDS-C's additional *security* traffic did not impose a statistically significant change in the overall traffic. Instead, the Cyclone DDS and network setup contributed to this change.

Figure 5 and 6's *data message* traffic for all three components show no dramatic change overall because Cyclone1 sends out 100 video frames regardless of participant count. Cyclone1 and Cyclone2 send traffic through KerAzoth's OpenVPN server; therefore, the average traffic of both Cyclone1 and Cyclone2 should be roughly equal to KerAzoth's. For Cyclone1 and Cyclone2 the average is around 7,509 KB, which doubled is 15,018 KB. This byte average is roughly equal to KerAzoth's *data message* traffic's byte average of 15,028 KB. The figures' *security* traffic is consistent with participants and their DDS-C authentication. Cyclone1 has only one *domain participant* to authenticate; therefore, overall traffic has no substantial

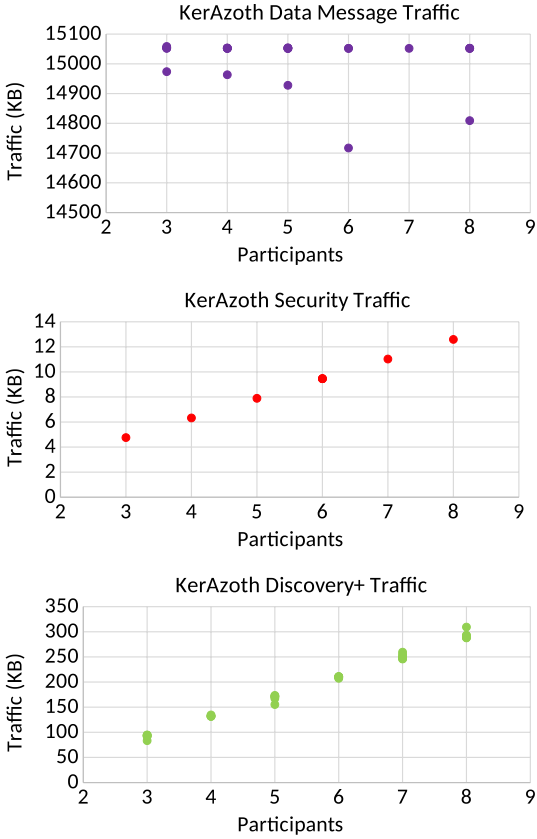


**Fig. 6** Experiment results for KerAzoth

change. Cyclone2 and KerAzoth's *domain participant* count increases for each experiment run, resulting in increased authentication and a strong positive linear correlation. Due to the consistent participant counts, all three components' *discovery+* traffic have positive linear correlations.

The experiment results show that DDS-C's security overhead is not statistically significant enough to hinder normal DDS operations with sending video frames. Using DDS-C in a real-world environment with architecture similar to Bright Apps will benefit DDS security and expand its integration in more use cases.

## 4 Conclusion

This research experimented DDS-Cerberus (DDS-C) with use cases around unmanned aerial vehicles (UAV), the cloud, and video streams. The background on Data Distribution Service (DDS), Kerberos, and related works on UAV-related papers culminated into the three use cases. The experiment tested these use cases by connecting devices through a Virtual Private Network (VPN) and used DDS-C

to authenticate *domain participants*. The experiment's configuration emulated use cases for real-world operations such as using UAVs for search and rescue. It used Cyclone DDS as its testbed where the nodes in the *domain participants* deal with sending and receiving video frames, emulating UAVs sending their video feeds for artificial intelligence (AI) processing. The Quality of Service (QoS) used was best effort to mimic an operational environment where some frames are not needed for the facial recognition AI. To analyze the collected traffic from the configuration, the packets were divided into three message categories: *data message*, *security*, and *discovery+* traffic. The *security* traffic quantity was low enough to not be statistically significant for the majority of the configuration runs. The results show that the mean traffic from DDS-C overhead is insignificant when constant video frames are sent over the network. The experiment shows that DDS-C applied to other DDS implementations or even in conjunction with other software adds security benefits without hindering overall performance.

Future work would incorporate middleware handling multiple Internet of Things (IoT) devices for integration into real-time systems. The node authentication provided by DDS-C would be beneficial for search and rescue and battlefield operations. Future research aims to integrate the experiment setup with multiple UAVs and more diverse AI to build up deep neural networks for object recognition. Additionally, as DDS-C evolves with better functionality and features, future work could also include experimenting with cyber attacks against it. These future work ideas could develop, extend, and add to the use cases in this paper.

Governments, companies, and people are looking to improve existing technologies through future works. DDS-C is still in development and requires more real-world experimentation before operational use to improve DDS security and development.

**Acknowledgments** This research was supported by Bright Apps. The team there provided the resources and support to conduct these experiments. The views expressed in this document are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense or the United States Government. This material is declared a work of the US Government and is not subject to copyright protection in the United States.

## Declarations

**Conflict of interest** The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request. The authors have no relevant financial or non-financial interests to disclose.

## References


1. Sánchez R, Evans J, Minden G (1999) Networking on the battlefield: challenges in highly dynamic multi-hop wireless networks, In: MILCOM 1999, IEEE military communications, conference proceedings (Cat. No. 99CH36341), vol. 2, p 751-755, IEEE
2. Munir A, Kwon J, Lee JH, Kong J, Blasch E, Aved AJ, Muhammad K (2021) FogSurv: a fog-assisted architecture for urban surveillance using artificial intelligence and data fusion, IEEE Access, 9, 111938–111959.
3. Nobre J, Rosario D, Both C, Cerqueira E, Gerla M (2016) Toward software-defined battlefield networking, IEEE Commun Mag 54(10):152–157. https://doi.org/10.1109/MCOM.2016.7588285
4. Object Management Group: OMG Standards for Industries, https://www.omg.org/industries/index.htm. Accessed 11 Oct 2021
5. Abdulghani RM, Alrehili MM, Almuhanna AA, Alhazmi OH (2020) Vulnerabilities and security issues in IoT protocols, In: 2020 First international conference of smart systems and emerging technologies (SMARTTECH), p 7–12, IEEE
6. Michaud MJ, Dean T, Leblanc SP (2018) Attacking OMG data distribution service (DDS) based real-time mission critical distributed systems, In: 2018 13th International conference on malicious and unwanted software (MALWARE), pp. 68–77, IEEE
7. Goerke N, Timmermann D, Baumgart I (2021) Who controls your robot? An evaluation of ROS security mechanisms, In: 2021 7th International conference on automation, robotics and applications (ICARA), p 60–66, IEEE
8. Park AT, Dill R, Hodson DD, Henry WC (2021) DDS-Cerberus: data distribution via ticketing, In: The 2021 world congress in computer science, computer engineering and applied computing (CSCE'21)
9. Park AT, Dill R, Hodson DD, Henry WC (2021) DDS-Cerberus: Ticketing performance experiments and analysis, In: The 2021 international congress on computational science and computational intelligence (CSCI'21)
10. Park AT, Peck N, Dill R, Hodson DD, Grimaila MR, Henry WC Quantifying DDS-cerberus network control overhead, Unpublished
11. Bright apps LLC: bright apps LLC Accessed 11 Oct 2021, https://brightappsllc.com/
12. MIT Kerberos: kerberos V5 system administrator's guide, Accessed 11 Oct 2021, https://web.mit.edu/kerberos/krb5-1.10/krb5-1.10.7/doc/krb5-admin.html
13. Munir A, Kwon J, Lee JH, Kong J, Blasch E, Aved AJ, Muhammad K (2021) Fogsurv: A fog-assisted architecture for urban surveillance using artificial intelligence and data fusion. IEEE Access 9:111938–111959. https://doi.org/10.1109/ACCESS.2021.3102598
14. Ribeiro JP, Fontes H, Lopes M, Silva H, Campos R, Almeida JM, Silva E (2017) Uav cooperative perception based on dds communications network, In: OCEANS 2017 - Anchorage, p 1–8
15. Sandoval S, Thulasiraman P (2019) Cyber security assessment of the robot operating system 2 for aerial networks, In: 2019 IEEE International systems conference (SysCon), p 1–8,https://doi.org/10.1109/SYSCON.2019.8836824
16. Eclipse foundation: eclipse cyclone DDS Accessed 11 Oct 2021, https://github.com/eclipse-cyclonedds/cyclonedds
17. Eclipse foundation: eclipse cyclone DDS Accessed 11 Oct 2021 https://projects.eclipse.org/projects/iot.cyclonedds
18. Eclipse foundation: python binding for eclipse cyclone DDS Accessed 11 Oct 2021, https://github.com/eclipse-cyclonedds/cyclonedds-python
19. Amazon: Amazon EC2, Accessed 11 Oct 2021, https://aws.amazon.com/ec2/
20. OpenVPN: OpenVPN, Accessed 11 Oct 2021, https://openvpn.net/
21. Amazon: Amazon EC2 instance types, Accessed 11 Oct 2021, https://aws.amazon.com/ec2/instance-types/
22. Object management group: DDS security, Accessed 11 Oct 2021, https://www.omg.org/spec/DDS-SECURITY/1.1/PDF
23. The Tcpdump group: TCPDUMP/LIBPCAP public repository, Accessed 11 Oct 2021, https://www.tcpdump.org/
24. Windows: powerShell, Accessed 11 Oct 2021, https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/powershell
25. Wireshark: tshark, Accessed 11 Oct 2021, https://www.wireshark.org/docs/man-pages/tshark.html

26. Python: python 3.0 release, Accessed 11 Oct 2021, https://www.python.org/download/releases/3.0/
27. SciPy: SciPy, Accessed 11 Oct 2021, https://scipy.org/



## Authors and Affiliations

**Andrew T. Park[1] · Nathaniel Peck[1] · Richard Dill[1] · Douglas D. Hodson[1] · Michael R. Grimaila[1] · Wayne C. Henry[1]**

Nathaniel Peck
2012raptor@gmail.com

Richard Dill
richard.dill@afit.edu

Douglas D. Hodson
douglas.hodson@afit.edu

Michael R. Grimaila
michael.grimaila@afit.edu

Wayne C. Henry
wayne.henry@afit.edu

[1] Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Dayton, OH 45433, USA