DCNTTM

DEF CON'S NEXT TOP THREAT MODEL

https://www.threatmodel.us
https://www.twitter.com/ThreatModelUs

Design Artifact for DEF CON 30

Drone Online Food Delivery Design

Revision History

Version	Date	Notes	Status
1	2021-06-01	Initial Draft based on Forderly Proposal	Initial Draft
2	2022-07-01	Updated based on feedback from MDD and MSD teams	Approved by MDD & MSD
3	2022-07-22	Updated based on prototype	Conditionally Approved by Forderly
4	2022-08-06	Final Revision	Approved by MDD, MSD, Forderly

Table of Contents

Marketing Summary

Through our partnership with Forderly, Drone Online delivers your goods to your customers. It's **FAST** and **EASY**. Our fleet of high end drones deliver hot and fresh food in 2 to 8 minutes to any location in range.

Drone Online utilizes state of the art software and hardware technologies, enabling fast, reliable, and predictable delivery services to our retailers and their end customers.

High Level Service Description

Drone Online started out as a military warehouse supply chain service drone solution. Later Drone Online expanded the company into the commercial market with consumer oriented products. The commercial investment started with a warehouse management drone solution - loading and unloading trucks and warehouse shelves, and moving parcels within warehouses to optimize storage space. Mid-2021, Forderly approached Drone Online's management for a strategic partnership to expand into last mile food delivery.

We are centering our last mile food delivery solution around the warehouses that we already automate. These warehouses already act as bulk drone storage - especially when drones are not in use. The first restaurants to be onboarded must be within drone flight range of our warehouse customers. Additional restaurants must be within drone flight range of our staging station network - drones can recharge at any staging station, and drones can be moved between "home" charging stations as needed, to change their individual service areas. Once established with restaurants around the city, we plan to further leverage this last mile solution for parcel delivery. Parcels will be transferable between drones, allowing multiple drones to participate in delivery of individual packages.

In order to minimize hardware costs, we are repurposing our surplus military solution drones. Drones used for commercial business are stripped of military equipment, such as weapons. Reusable components that are useful for the new market, such as surveillance equipment and tracking hardware, are retained.

This leaves Drone Online with three types of drones: Military Defensive Drone (MDD), Military Surveillance Drone (MSD), and the Consumer Supply Chain Drone (CSCD; a rebranded MSD). Drone Online uses the same processing logic for military and consumer supply chain services. This simplifies consumer drone design, and lets us focus on providing exceptional services to both customer bases. Drones are provisioned based on customer needs.

Configuration Options

The CSCD is a standard 6 rotor drone, offering optional configurations to meet our customer needs. This is implemented by repurposing the hardpoints from the MDD, used to provide ordinance options for firing 9mm or .50BMG bullets, or 40mm grenades to military customers. Civilian supply chain options are mounted to these hardpoints, as needed for the mission.

- Light Multi-Cargo Carrier: 3kg, can carry up to 4 packages, up to 5kg each
- Heavy Cargo Carrier: 6kg, can carry 1 package, up to 18kg
- Restaurant Advertising/Branding: Drones can provide a mount for a 1/3kg cloth sign (1m x 1.5m). This can be simple branding, or can advertise deals/specials.

Additional Consumer Supply Chain Drone (CSCD) Specifications

- Standard 6 Rotor Drone
- Flight: Max speed: 10 m/s; Hover time: 18 mins @ 30kg; 12 mins @ 50kg
- Flight range: 3km (with 2.5m altitude); Max Altitude: 30m
- Weight: 30kg; Max Take-off Weight: 50kg
- Camera Resolution: 1280 X 960 @ 30 FPS
- ROM: 32GB + microSD up to 128GB, R/W speed up to UHS-I Speed Grade 3
- Manual control capable

Safety and Autonomy

- If a drone can not safely deliver the order/payload at a delivery location the drone will attempt to “drop” the payload at a height of < 1m (~3 feet)
- If a drone encounters a sensory malfunction, bad weather, or component issues it will attempt a slow descent until it can recover or it lands. After 30 seconds the drone will attempt to continue its planned flight and delivery objectives .

Use Cases

End User Onboarding

Not all delivery locations are viable. Some may be near airports, and others may be near power lines. When new users (or new delivery locations) are defined, Drone Online performs a site evaluation.

This site evaluation is currently manual, but we expect to automate much of it in the future.

1. If user has an account, they log in

2. If user doesn't have an account, they create an account. Specify
    - Username (email address)
    - NIST SP 800-63b compliant password
3. User designates one or more delivery locations
4. If the delivery drop off point was previously approved, no further action is taken
5. If the drop off point is new, a Drop Off Site Survey is scheduled
6. At some future time, a Drone Online employee performs the Drop Off Site Survey
    - Looks up the location on a map to check if the location is too close to a hazard (e.g. too close to an airport)
    - The employee can take control of a drone or may drive out to the drop off site to inspect the location to ensure drop-off is viable
    - Reviews surrounding area (e.g. locations on the same street) to verify that those locations are viable as alternative landing or drop off sites
    - Marks the location (and any other surveyed locations) as "approved"

Restaurant Onboarding

Restaurants are on-boarded to the Drone Online service to:

a. Set up accounts & payment information
b. Determine if the Restaurant has facilities to support drone storage and charging

Restaurants should have a suitable location at or near their operating location for one or more drones to be stationed for order pickup (and returns if supported) and for times when the drone is not in use. The location should have suitable power and space for the drone, which must take into account take-off and landing operations, and be physically secure. Drone Online may recommend additional site security measures when installing the Drone station(s).

1. Restaurant uses an online form to initiate registration, providing the following information:
    - Restaurant name and address
    - Name of restaurant manager or owner
    - Describe landing zone for temporary drone storage and charging between orders. This will also serve as the pickup location where packages will be loaded onto the drone.
    - Date and time for site visit based on a calendar showing available appointments
2. Drone Online employee visits restaurant to evaluate landing zone, pick up location, and train vendor on pick up process
    - Drone Online gives Restaurant a Drone Online operation manual
    - Drone Online Operations configures Drone Cloud for new staging stations, configures internal tunnel port for created on Drone Proxy Host, creates configuration package for Drone Proxy Client..
    - Installs charging station
    - Charging station automatically powers on after being plugged in

    - Connects Charging Station to the Restaurant's Wifi and verifies connectivity to Drone Service
    - Drone Online employee takes manual control of a nearby drone, enables training mode, and flies drone from nearest available initial staging station to newly installed target staging station
3. Restaurant employee submits order to Drone Online in TEST_MODE, loads parcel into drone cargo carrier and observes successful delivery.
4. Drone Online employee advises Restaurant employee to submit regular restaurant customers' addresses for End User Onboarding.

Restaurant Owner Order Management

Restaurant Owners have several tasks they are expected to regularly perform, as described in the following subsections.

Check drone staging stations

Physically check the drone charging station and ensure it's free of debris, the power cable remains connected, and the station is generally in good condition. A checklist with photos is included in the operation manual. Remediate any issues that can be safely and immediately remediated (e.g. reconnect the plug to outlet should it be removed, remove leaves and other small debris from the charging station, unplug and wash the charging station on a weekly basis). Report other issues to Drone Online.

Check Drone Locations

Restaurant owners can use the Drone Online WWW to check locations of drones currently delivering their orders, and status of drones coming to pick up orders.

See Delivery History

Restaurant owners can use the Drone Online WWW to review the delivery history for all orders within the last 30 days. Beyond 30 days, only cumulative monthly data is stored.

Reporting

Restaurant owners can use the Drone Online WWW to generate reports on delivery efficiency: pickups, complaints from customers, etc.

Order/delivery process

1. User buys something through Forderly and requests drone delivery

2. Forderly redirects authentication of user to Drone Online to validate user has a Drone Online account
3. Forderly requests a list of drop off points registered for that user and presents these to the user
4. User selects a drop off point
   - User may enter a new drop off point, but it will be rejected for this order if it hasn't been already approved as part of an End User Onboarding - Drop Off Site Survey. Drone Online will automatically add the new drop off point for a Drop Off Site Survey Request
5. Drone Online verifies that drones are available and can support deliveries from restaurant to user drop off locations.
   - Drones are not assigned to specific customers but a guaranteed capacity is maintained based on customer's subscription level.
   - Drones exist as a mesh network. Any available drone in the area may be selected by the Drone Online AI for the delivery.
   - For markets with existing DO Warehouse locations, DO Warehouse drones may be used to deliver payloads if capacity is exceeded by DO Food Delivery Drones.
   - Some drone models are capable of carrying multiple deliveries. Where logistically appropriate, Drone Online AI uses this capability to service multiple deliveries from multiple restaurants.
6. Drone online reserves the delivery capacity for the order. Capacity is reserved for five (5) minutes, giving the customer time to complete their order with Forderly.
7. Forderly submits delivery requests to Drone Online
8. Drone Online dispatches drone
   - If there's already an available drone charging at the restaurant, that drone will typically be selected
   - If there are no drones available at the restaurant, or if the drone at the restaurant has insufficient battery charge to perform the delivery, a nearby drone with capacity to handle the delivery will be selected.
9. Drone lands at restaurant
10. Restaurant Staff loads items in drone
    - Staging Station UI provides instructions for how to securely load the package
    - If the drone is capable of carrying multiple packages, the Staging Station UI provides instructions for which carrier to use.
11. Restaurant Staff initiates flight by pressing the "Initiate Transfer" button on the staging station UI
12. Drone delivers packages to customers, then returns to Drone Online or vendor staging station

Drone Online staff responsibilities

Drone Online staff have several responsibilities that they fulfill as part of delivery operations, as described in the following subsections.

Manual Drone Control

Drone Online staff can use the Drone Online WWW to take manual control over drones. This is intended to be used when

- Manual “training” mode when a Staging Station is first installed
- Drones get “lost” and cannot find their way to a delivery location,
- Drone location antennas become damaged or otherwise cannot find its way back to a charging station

Real Time Monitoring of Drone Fleet

Drone Online staff can use the Drone Online WWW to monitor the drone fleet. Real time locations of all drones, access data from all drone sensors, previous and planned flight paths, and any other data or controls that are useful in detecting and correcting problems with the fleet.

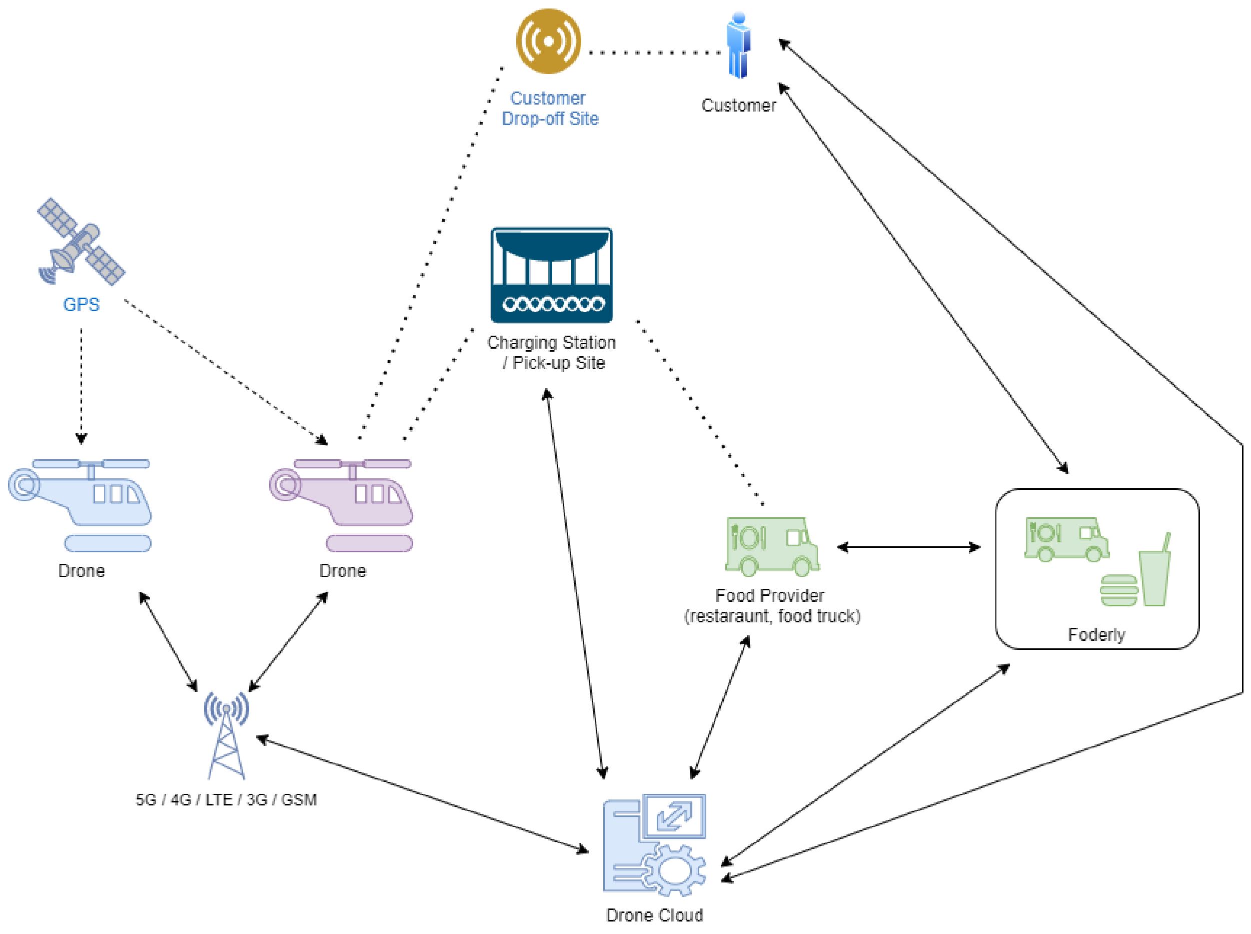
Drone repair/maintenance

If drones become damaged, Drones will automatically attempt to return home to Drone Online facilities. Drone Online employees can enable “return home” mode on misbehaving drones that do not return home on their own.

If drones cannot return home, Drone Online staff will use the drone’s last known location to either repair the drone or bring it back to Drone Online facilities for repair.

Design Diagrams

Context Diagram

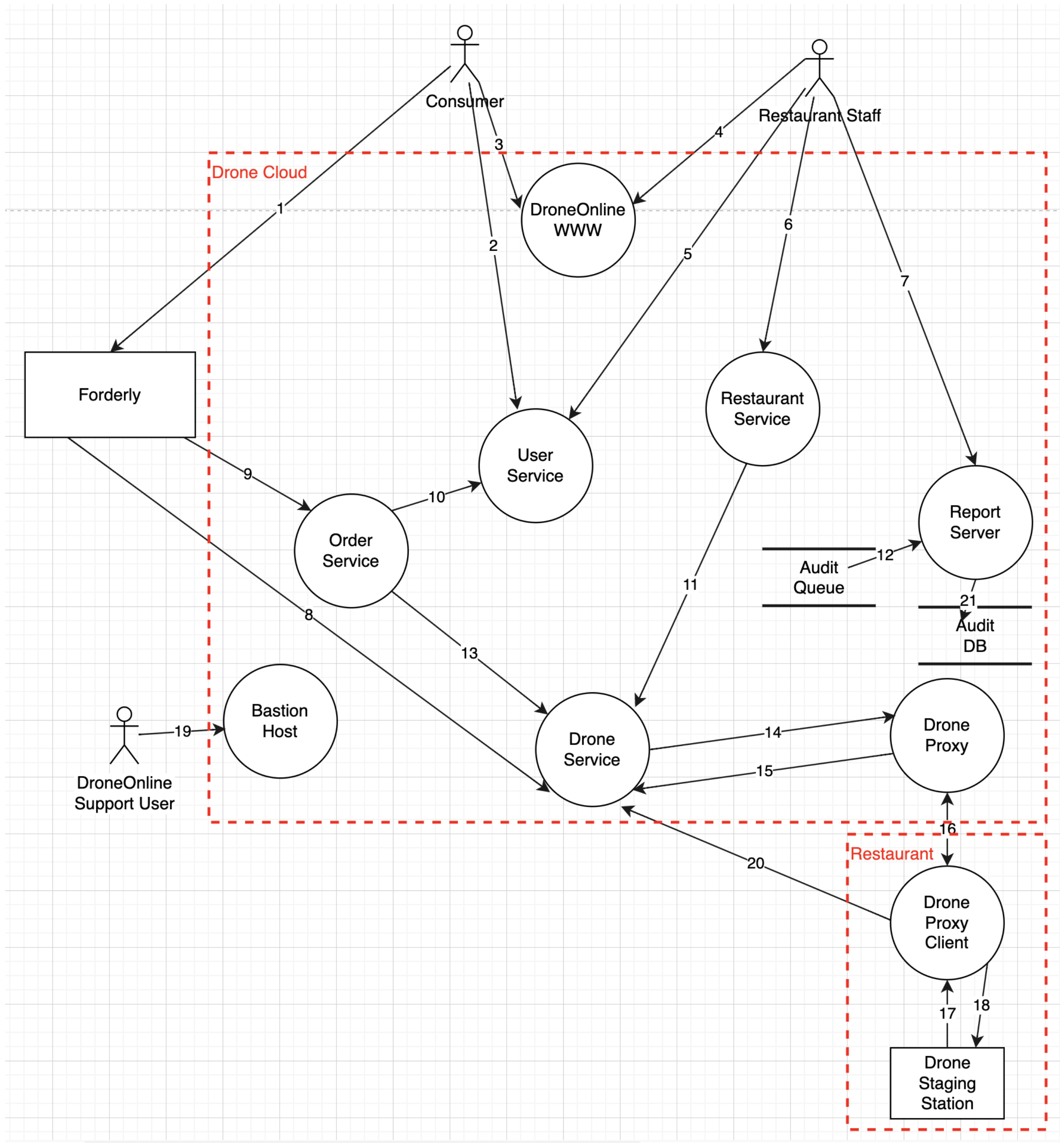


As shown in this diagram, customers ordering food interact with Forderly directly or through a Restaurant, specify pre-approved addresses for food drop off, and Drone Online performs delivery.

Drone Cloud services act as the central orchestrator and entry point to the Drone ecosystem for Forderly users, food ordering customers, and Drone Online personnel. Forderly (an independent online service) communicates with Drone Cloud to perform certain functions, such as user- and provider registrations. Food Provider users interact with Drone Cloud to provide pick-up and delivery instructions; Forderly communicates with Food Providers to transmit payment or order information or other functions (defined separately by Forderly). Customers ordering food interact with Drone Cloud to provide drop-off and delivery instructions. Food Providers must have access to a charging station for order pick-up by a Drone; The Drone Charging/Staging stations provide a convenient place for Drones to land to be recharged, send/receive data transmissions, and to pick-up or drop-off payloads.

Drone Cloud (Drone Online) communicates to the collection of Drones in service via radio communications (short to long range). Drones communicate to Drone Cloud to receive instructions, mapping updates, pick-up/drop-off coordinates, and to send telemetry data. Drones can extend communications range by communicating directly to each other via the radio network. Once every 60 seconds, drones report their location back to Drone Online based on GPS or other location services.

Network Diagram



API

Drone Online's food delivery API empowers customers to utilize drones for commercial delivery. The API facilitates

- user profile and account management
- delivery and order management
- administrative and reporting functions.

The Drone Online API runs on a powerful and robust NodeJS backend.

The API is stateless, based on REST principles using HTTP verbs: GET, POST, PUT, DELETE

- All API communication must be done using **TLS 1.3**
- All API response data will be JSON format
- All POST requests must have a JSON formatted body

API Authentication

Authentication to the Drone Online API is done either using an API Key or using OAuth2 to obtain an access token. API Keys and OAuth2 access tokens are standard JSON Web Tokens, utilizing "DOSIG" a proprietary military grade encryption algorithm for signature generation.

API Keys

API keys are used for system to system requests, such as Forderly communicating to the Order Service. They are communicated with HTTP Basic Authentication. API Keys are managed by the Drone Online IT Admin staff and are generated manually and shared with new API consumers IT or engineering staff via email. Usernames are typically the customer id. Passwords securely generated from a SHA hash of the customer id as a seed to a secure random number generator with 10,000 iterations.

Sample header

```
Authorization: Basic Y3VzdG9tZXJfaWQ6cmFuZG9tX3Bhc3M=
```

OAuth2

Used for User based API interaction from a web browser. Requests to the API will be authorized by including an `Authorization` header with the value `Bearer <accessToken>`.

Sample header

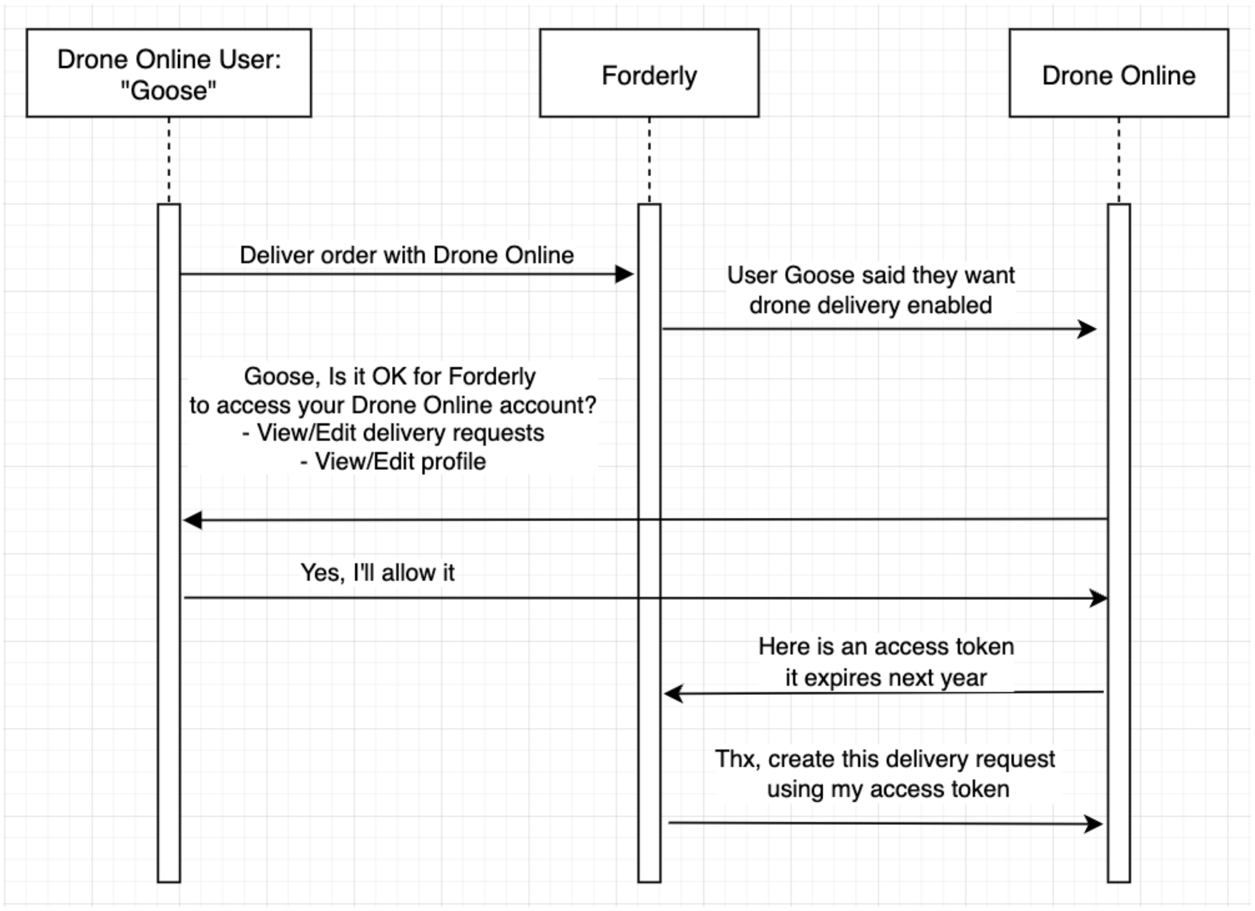
```
Authorization: Bearer
ewogICJhbGciOiAiRE9TSUciLAogICJ0eXAiOiAiSldUIgp9.eyJpc3MiOiJEcm
9uZSBPbmxpbmUgQVBJIiwiaWF0IjoxNjYwMjU2MjM5LCJleHAiOjE2OTIxMzc4M
```

```
zksImF1ZCI6Ind3dy5kcm9uZW9ubGluZS5jb20iLCJzdWIiOiJHb29zZSIsIlJv
bGUiOlsiVXNlciIsIk9yZGVyIiwiV2ViIl19.jxaMced8xgJZ_EbkFyoaN0-m5y
guSieVwywkNMesXOk
```

Decoded

```
{
    "alg": "DOSIG",
    "typ": "JWT"
}
{
    "iss": "Drone Online API",
    "iat": 1660256239,
    "exp": 1692137839,
    "aud": "www.droneonline.com",
    "sub": "Goose",
    "Role": [
        "User",
        "Order",
        "Web"
    ]
}
```

Access tokens are issued via an OAuth2 authorization flow described below:


Drone Online User: "Goose"
Forderly
Drone Online
Deliver order with Drone Online
User Goose said they want drone delivery enabled
Goose, Is it OK for Forderly to access your Drone Online account? - View/Edit delivery requests - View/Edit profile
Yes, I'll allow it
Here is an access token it expires next year
Thx, create this delivery request using my access token

API List

- Calls to `/order/` are implemented by the Order Service
- Calls to `/user/` are implemented by the User Service
- Calls to `/drone/` are implemented by the Drone Service
- Calls to `/restaurant/` are implemented by the Restaurant Service

API	Description
`/order/deliveryOptions` Auth type: API Key	Lists information about currently available delivery options for a customer. GET lists delivery options defined for this customer ● Parameters ● user: email address of the customer who is ordering from the restaurant ● Time: the datetime for which the restaurant wants

	the drone to pick up the delivery. Pickup is a 5 minute time window (time to attach the package to the drone). ● Type. The type of drone that you want to select ● Features. The drone features that you want added. ● Returns ● 200: delivery options are provided as a JSON list. ● 401: The type of drone is invalid ● 404: if email address is not known. End user (restaurant’s customer) needs to be registered ● 425: No drones currently available. Change delivery time or find alternate delivery mechanism ● 500: other error Sample JSON Result `{ user: emailaddr, deliveryoption: { uuid: "uuid”, address: "1 main st, city, state, ip", instructions: "place inside front door" }, ... }`
`/order/order` Auth type: API Key	Manages orders for drone delivery. GET is used to retrieve a list of all current orders POST is used to initially create an order ● Parameters ● customer: the customer to whom to deliver ● location: the UUID of the delivery location to which to deliver the drone ● Returns ● 200 to indicate success. Comes with JSON specifying orderid. Drone capacity is reserved for this order for 5 minutes; PUT must be called within 5 minutes to confirm the order. ● 416: Delivery address outside drone delivery range. Find an alternate delivery mechanism or select another location. ● 401: invalid API key ● 404: invalid customer ● 498: invalid UUID ● 500: other internal errors PUT is used to confirm an order ● Parameters

	◦ orderid • Returns ◦ 200: success ◦ 401: oderid is for a different customer ◦ 404: orderid is not valid. ◦ 498: the 5 minute reservation window for unconfirmed deliveries has expired. ◦ 500: other internal errors
`/user/login` Auth type: none	GET redirects to the Drone Web UI, which presents a login page GET requests can initiate an OAuth2 authorization code flow to authorize other systems to call the API on a user's behalf. • Parameters • response_type • scope • client_id • state • redirect_uri • Returns • 302 indicates successful initiation of authz flow. • 401: The client_id is invalid • 404: user not registered for Drone Online • 500: other errors. POST Performs first factor authentication. • Parameters • Username: user's email address • Password: the user's password • Returns • 200: first factor authentication was successful • 401: invalid password • 404: user not found • 423: account locked; too many failed login attempts • 500: other internal errors NOTE: Client MUST call login2 next.
`/user/login2` Auth type: OAuth2 token	POST Provides a list of authentication mechanisms that the user is signed up for. For example, SMS authentication. • Parameters • API Key: JWT; must represent a partially logged in session • Username: user's email address • Returns • 200: Success. Body contains a JSON list of all authentication mechanisms that the user is signed up for

	<ul><li>401: OAuth token signature is invalid or expired</li><li>404: user not found</li><li>410: API Key does not represent a partially logged in session</li><li>423: account locked; too many failed login attempts</li><li>500: other internal errors</li></ul> Sample JSON `{ options: "SMSCODE", "emailCODE", "password" }` Note: Client MUST call login3 next.
`/user/login3` Auth type: OAuth2 token	POST Accepts 2nd factor authentication information (e.g. Q&A authentication, or SMS tokencode). Verifies the API key represents a partially logged in session.<ul><li>Parameters<ul><li>API Key: JWT. must represent a partially logged in session</li><li>Username: user's email address</li><li>Credential: 2nd factor authentication credential</li></ul></li><li>Returns<ul><li>200: login successful. Returns a new JWT that represents a fully logged in user.</li><li>401: 2nd factor credential is invalid</li><li>404: user not found</li><li>410: API key does not represent a partially logged in session</li><li>423: account locked; too many failed login attempts</li><li>500: internal error</li></ul></li></ul>
`/user/logout` Auth type: OAuth2 token	Sets the expiration datetime of the JWT to the current datetime.
`/user/token` Auth type: OAuth2 token	Used to manage API tokens GET without a parameter lists the available API tokens. With a parameter generates a new API token<ul><li>Parameters<ul><li>alg: signature algorithm. MD5 is used for testing. DOSIG is used for production.</li><li>typ: token type. Specify JWT</li><li>sub: subscriber</li><li>Role: requested roles; specify the parameter multiple times to specify multiple roles</li></ul></li><li>Returns<ul><li>200: Successful list request. Response body lists available tokens.</li><li>201: Successful JWT creation. JWT is included as</li></ul></li></ul>

	JSON response. • 204 for list requests, if there are no tokens for the logged in user • 401: invalid role(s) for the subscriber • 404: subscriber not found • 500: other internal errors • Note: Only 10 tokens are listed at a time. Call GET again to see the rest of the list. DELETE deletes the token. Responds with an HTTP response setting the token cookie's value to empty and expiration to 30 seconds in the future PUT and UPDATE are not implemented
`/user/deliverySite` Auth type: API Key	GET lists delivery sites. • Parameters: email address, API Key • Returns: list of site details (id, address) PUT adds a delivery site to the list of sites that are allowed. Initiates the process described in use case "User Onboarding" • Parameters: email address, API Key, delivery site details • Returns: • 201: indicates request created • 202: indicates site is already approved • 401: invalid API key • 500: server errors DELETE deletes a delivery site • Parameters: site ID • Returns: • 200: site deleted • 401: invalid API key • 404: site id does not exist • 400: could not delete site • 500: server error POST is not implemented. Call DELETE then PUT.
`/user` Auth type: API Key	CRUD api to update user record stored in the service database. Name, Profile image/avatar, email, phone, primary address, password, favorite restaurants. • Parameters: JSON user object with fields.
`/drone/status` Auth type: API Key	Used to check or report drone status issues POST gets status for a particular drone • Parameters • droneid • Returns

	● 401: API key invalid (likely expired) ● 404: droneid invalid PUT reports status for a particular drone ● Parameters ● id: the drone identifier ● status: one of flying, hovering, landing, landed, loading, unloaded ● battery: battery percentage. Number from 0-100 ● location: location based on GPS or other coordinates ● type: Military Defensive Drone (MDD), Military Surveillance Drone (MSD), and the Consumer Supply Chain Drone (CSCD; rebranded MSD) ● Returns ● 200: success ● 400: battery percentage not 0-100 ● 401: invalid API key ● 403: invalid status ● 404: invalid drone id ● 409: drone type does not match database record ● 500: internal errors GET and DELETE are not implemented
`/drone/station/proxy`	APIs used when setting up a proxy station and when updating proxy client configuration. GET obtains configuration data or executable files/scripts for the Proxy Client within the Staging Station. The executable files first update all software running on the Staging Station, then executes shell scripts which apply the configuration settings. ● Parameters ○ serialnum: serial number of the proxy station ○ returnBinary: boolean to return json data or to return zip of binary files and shell scripts with configuration data pre-populated within various args. ● Returns ○ 200: configuration or zipped files returned in body ○ 404: serialnum invalid ○ 500: internal error POST called when the Charging Station is first manufactured, to register a station’s new serial number ● Parameters ○ serialnum: staging station’s serial number ● Returns ○ 200: staging station registered ○ 409: serial number already registered

	○ 500: internal error
	
	PUT is used to confirm that the configuration data was received and the station was successfully configured.
	● Parameters
	○ serialnum: staging station's serial number
	○ Request Body
	■ Configuration data, same as returned in GET.
	● Returns
	○ 200: staging station updated, set registered = true
	○ 409: serial number already registered
	○ 500: internal error
	
	DELETE not implemented; deleting charging stations is a manual process performed only by Tier 5 Customer Support
	
	Example configuration data included in the GET response. Each call returns a new otp
	`{ locationname: "Foobar Restaurant",`
	`  locationaddr: "1 main st, city, state, zip",`
	`  sshport: 4536,`
	`  sshuser: proxyuser`
	`  sshotp: 7c3fef8a-1a20-11ed-b813-5b800f8cd92e`
	`}`

`/restaurant` Auth type: [API Key]	GET lists all restaurants that you have permission to view. Each restaurant is listed with its database unique key (a unique number) ● Returns ● 200: success. JSON list of all restaurants in body ● 401: invalid API key ● 500: server error POST submits a restaurant onboarding request, associating the current user with the restaurant ● Parameters ● Name: restaurant name ● Addr: address of the restaurant ● Returns ● 201: success. new restaurant onboarding request created ● 202: restaurant already approved ● 401: invalid API key ● 409: onboarding request already exists ● 500: server errors PUT and DELETE are not implemented
`/restaurant/user`	Used to associate users with Restaurants.

Auth type: API Key	GET lists all users associated with a particular restaurant • Parameters • restaurant_id: the restaurant's unique identifier, whose users to list • Returns • 200: a list of usernames and their assigned roles • 401: invalid API key • 404: no such restaurant ID • 500: internal error POST • Parameters • username: user to associate with the restaurant • Returns • 201: user was added • 401: invalid API key • 404: no such user • 409: user already associated with restaurant • 500: internal error • Note: use PUT to assign at least one rone after you’ve created the user PUT assigns a role to the user • Parameters • username • role • restaurant_id • Returns • 201: user was added • 401: invalid API key • 404: no such user • 409: user already has role • 500: internal error DELETE deletes a role from a user, or a user from a restaurant • Parameters: • username • role (optional) • restaurant_id • Returns • 200: operation successful • 401: invalid API key • 404: no such user • 406: the user still has roles assigned • 400: the role couldn’t be removed • 500: server error

Sample API Request - Basic authentication

```
GET /user/order/29924153 HTTP/1.1
Host: api.droneonline.com
Authorization: Basic Y3VzdG9tZXJfaWQ6cmFuZG9tX3Bhc3M=
Accept: */*
Accept-Encoding: gzip, deflate
```

Sample API Request - API key

```
GET /user/order/29924153 HTTP/1.1
Host: api.droneonline.com
Authorization: Bearer
ewogICJhbGciOiAiRE9TSUciLAogICJ0eXAiOiAiSldUIgp9.eyJpc3MiOiJEcm9uZSBP
bmxpbmUgQVBJIiwiaWF0IjoxNjYwMjU2MjM5LCJleHAiOjE2OTIxMzc4MzksImF1ZCI6I
nd3dy5kcm9uZW9ubGluZS5jb20iLCJzdWIiOiJHb29zZSIsIlJvbGUiOlsiVXNlciIsIk
9yZGVyIiwiV2ViIl19.jxaMced8xgJZ_EbkFyoaN0-m5yguSieVwywkNMesXOk
Accept: */*
Accept-Encoding: gzip, deflate
```

Rate Limiting

Rate limiting to Drone Online APIs is a simple calculation based on the number of requests made by the source IP address. Depending on your API service level, the API Gateway will return **429** responses when the request allowance is exceeded. See the table below.

API Service Tier	**Max requests per second**
Consumer	50
Business	100
Enterprise	1000

Errors

Error conditions will typically be handled with an HTTP response and appropriate HTTP response code. Error descriptions are in JSON format and contain a reference ID that can be used for support cases. If you see an error response that includes system information or stack trace data please contact our security team.

Sample error response body

```
{
  “errorId”:”f74d65ac-312a-577d-8999-c39eded5df34”,
  “message”:”An error has occurred. Please try again.”
}
```

Webhooks

API consumers can publish a webhook to receive callback notifications for specific Drone Online events. All webhook URLs must have a valid TLS certificate and be in the format: **`https://[www.]yourdomain.com/your-webhook`** and be base64 + URL encoded in the url parameter and include the order number in the order parameter.

Sample URL parameter encoding

```
url=aHR0cHM6Ly93d3cuZm9yZGVybHkuY29tL3dlYmhvb2svZG8vcmVjaWV2ZQ%
3d%3d&ordernumber=29924153
```

Webhook Actions

Drone Online Action	Registration URL	Description
Order Received	`/webhook/order/received`	Called when the drone receives its pickup location and payload type
Order Delivered	`/webhook/order/delivered`	Called when the drone releases its payload at the specified GPS waypoint
Order Status	`/webhook/order/status`	Called every 60 seconds with the delivery status, drone position, and ETA.

Note: To view responses and response code logs, please use the webhook log API endpoint. Webhook response logs are only stored for 24 hours.

```
GET /webhook/log HTTP/1.1
Host: api.droneonline.com
Authorization: Basic Zm9yZGVybHkxOjIwTDM4I0ZDM0pTZA==
Accept: */*
Accept-Encoding: gzip, deflate
```

Sample Registration Request

```
GET
/webhook/order/received?url=aHR0cHM6Ly93d3cuZm9yZGVybHkuY29tL3dlYmhvb
2svZG8vcmVjaWV2ZQ%3d%3d&ordernumber=29924153 HTTP/1.1
Host: api.droneonline.com
Authorization: Basic Zm9yZGVybHkxOjIwTDM4I0ZDM0pTZA==
Accept: */*
Accept-Encoding: gzip, deflate
```

Sample Webhook Response

```
Method: POST
Headers: "Content-Type: application/json"
Body: {
 "id" : "0eb102f8-0e93-11ed-861d-0242ac120002",
 "action" : "Order Received",
  "order" : "29924153",
  "droneType" : "CSCD",
  "loc" : [36.116628, -115.176757],
  "dest" : [36.114704, -115.201462],
  "data" : "image/png;base64,iVBOR...."
}
```

Data Storage

All services use a Redis cluster for shared data storage of everything except account information, audit and report data. Accounts - both user and system accounts - are stored in a PostgreSQL database running on the User service, as are audit and reporting data.

Systems

Forderly (or Any other Restaurant Chain)	
Description	"Forderly" represents any restaurant that might use the Drone Online restaurant delivery services. **Forderly marketing materials** Forderly information technology systems provides a complete technology solution for the most modern restaurants in the world: ● On-prem point of sale systems for your cashiers ● On-prem ordering kiosks located on the restaurant's tables

	● On-prem order printing in the kitchen ● Online ordering, integrated with your website, with user profiles that persist across Forderly customers, so they never have to duplicate their delivery address, allergies, or other preferences, again ● Rich, back-end interface for restaurant owners and managers to customize your menu and monitor all aspects of your store - hot ticket items, sales, stock, and more ● And now fully automated delivery!
In Scope?	No. All internal systems of the restaurant are out of scope of this design. All food payment happens inside the restaurant, before delivery is scheduled through Drone Online.
Other	Audit events from Forderly are captured in the Audit Queue. A report provided by the Report Server is available to Drone Online Support staff to detect orders submitted by Forderly that are not matched by event receipt by Drone Online.

User Service	
Description	This is a back end system with which both the Drone Online Web UI and the API clients interact. Web users are able to use the API provided by the Order Service to save drone delivery locations in their profile. The User Service is responsible for facilitating OAuth2 based authorization requests for consumers like Forderly who need access to the Drone Online API's. Key APIs implemented by this service are: ● /user/login ● /user/login2 ● /user/login3 ● /user/deliverySite
In Scope?	Yes
Other	● The User Service is written in Python. ● The available OAuth scopes are: ● `email`: The users email address which is also the value used for the username ● `order`: The ability to create and list orders for the user ● `site`: The ability to read delivery sites from the user's profile ● OAuth Redirect URLs must use HTTPS and be pre-configured by the Drone Online administrators. Currently only the following URL's are

	configured and permitted for redirection: ● https://localhost/test ● https://*.forderly.com* ● Wild card matching is permitted in the URL to save time when multiple subdomains or endpoints are required. Wild card matching is performed by using a regular expression "`[^ ]*`" in place of the * in the URL. ● User passwords are stored as SHA2-256 hashes combined with a 8 byte salt value derived from the last 8 characters of the user's username. ● Second factor authentication codes are delivered using a third party vendor API which can deliver OTP codes via SMS and Email. The API is also used to validate OTP codes sent back by the user during login. The third party API credentials are securely stored in a configuration file that is deployed with the service. ● Delivery locations have one of three status values: ○ Approved: the location is suitable for drone delivery. ○ Pending: the location is awaiting confirmation that it's suitable for drone delivery. ○ Denied: the location was determined not suitable for drone delivery.

Order Service

Description	The Order Service is the main interface for Forderly or other restaurant automation systems to reserve drone delivery services. The Order Service receives orders for drones from the restaurant, generates order IDs, verifies drone availability from the Drone Service, and reserves drones via the Drone Service. All information is stored in the service database.
In Scope?	Yes
Other	● Order Service is written in Python. ● Order IDs are generated using the TLS certificate as a secure random seed for the random library. https://docs.python.org/3/library/random.html . ● Implements all APIs under the `/order/` path

DroneOnline WWW	
Description	Single page web application written in Angular.js, used by Drone Online customers, end users, and Drone Online staff (customer support, hardware maintenance, and site evaluators). Functionality includes: Customers use this UI to register, submit/update/delete delivery sites for onboarding, view their delivery status, view their delivery videos (stored for 10 days), and contact customer support. Restaurants use the UI to check order status, fix delivery locations (if customers need to change location of an in-flight order or cancel delivery), view delivery videos, and contact customer support. Customer Support uses this UI to check delivery status, view delivery videos to confirm delivery, fix delivery sites (if customers need to change their delivery location for an in-flight order), check drone status, take manual control of drones that need assistance completing a delivery, and reproduce customer issues with the application. Customer support has a special API that allows them to switch their account to the customer account, so that they can use the UI and reproduce the same issues that customers experience. Hardware maintenance staff use this UI to identify (last known) locations of drones needing repairs, mark broken drones as “unavailable,” check drone repair history, log repair work, mark fixed drones as “available,” download updates (which will be installed via the drone’s MicroSD slot). Minor drone repairs may occur on-location, such as replacing a broken rotor blade or a dead battery, then released back into circulation. More complex repairs are performed on Drone Online premises, and are rotated back into circulation using the same process as drones are originally put into circulation. Site Evaluators use the UI for all evaluations (both restaurants and delivery sites). The UI lists locations that require evaluation, offers manual control of drones to perform site evaluations, and marks sites as approved or rejected.
In Scope?	Yes
Other	Where functionality is implemented by other services, the UI makes direct HTTPS calls to those services to perform those actions. For example, login is implemented in the User Service.

Restaurant Service

Description	This service provides API endpoints for Drone Online WWW for Restaurant users, in the same way that the User Service provides such functionality for the User Service. Right now, this endpoint only supports restaurant onboarding requests, once onboarded the service is used to retrieve drone staging station status and query drone status and location. A future version will provide additional information, such as historical deliveries, uploading customer lists for bulk account creation.
In Scope?	Yes
Other	Users with any Drone Online internal roles are able to list all restaurants known to Drone Online. External users are able to list only the restaurants for which they have submitted onboarding requests.

Bastion Host	
Description	The bastion host provides access to the production network so that maintenance tasks can be performed. This includes any necessary steps during software upgrades, as well as fixing outages. Access is restricted to Tier 4 and Tier 5 Customer Support staff.
In Scope?	Yes
Other	Authentication is via SSH using SSH keys. Keys are generated by the individual users on their assigned laptops, stored by the SSH software encrypted using the user's password and protected with strong file permissions (`user:user -rw-----`). A custom process replaces the shells of all Tier 4 staff, restricting what commands they are allowed to execute to a predefined allow-list. Common operations include: ● `view-host-log`. Executes `more do_logfile`, which contains all log data stored on that server ● `install-upgrade`. Executes a custom script that executes the latest version of the installer stored on the host. The installer is deployed as part of the software deploy process, which is outside the scope of this design. ● `update-os`. Updates all packages on the operating system. This is normally automatically run weekly, but can be executed manually in case of a critical outage.

	Tier 5 staff get a shell when they log in. They can sudo to root, which requires them to enter their password and provide an SMS OTP. Executing the sudo command alerts all other Tier 5 staff via SMS. The host is a fully patched, stripped down ubuntu installation that contains only the software packages necessary to run the Drone Online software and maintain the system. All existing packages are configured as per the relevant DISA STIGs.

Drone Service

Description	The drone service is responsible for drone management. Drones are assigned orders based on ● order time and expected drone availability, ● Drone charge, order weight, flight time from its current location, and charge level ● drone carrying capacity The Drone Service is responsible for calling all web hooks to ensure status is up to date.
In Scope?	Yes
Other	Internal details of this system are classified.

Drone Charging (Staging) Station

Description	In our Warehouse and Military business lines these Drone Charging (Staging) Stations exist on-site and hardwired to DO networks. In the new consumer food delivery service, these stations will exist on-site at restaurants. The Drone Proxy Host/Client were added to support bi-directional communication between the Drone Service and the Drone Stations. Food Providers must have access to a charging station for order pick-up by a Drone; The Drone Charging (Staging) Stations provide a convenient place

	for Drones to land to be recharged, send/receive data transmissions, and to pick-up or drop-off payloads The Drone Charging (Staging) Stations can send data to- and receive data from Drone Online's cloud environment, such as software updates, commands to relay to the Drone, and Drone telemetry data; this is done through the Drone Proxy Client co-resident with each Charging Station.
In Scope?	No. The station itself is out of scope as is Drone and Drone Staging Station specific functionality. How it interacts with other Drone Online systems via the Proxy components is in scope.

Drone Proxy Host	
Description	The Drone Proxy Host is an SSH server used for tunnels to allow secure communication between Drone Cloud and Drone Staging Stations. During connection a port is opened on the Proxy Host which the Drone Service uses to connect to the Client and access components within the Drone Staging Station. This port is controlled and only configured by Drone Online Operations during restaurant onboarding.
In Scope?	Yes
Other	

Drone Proxy Client	
Description	Drone Proxy Client is software running in a docker container, running inside the drone staging station. Software consists of shell scripts which retrieves information from the Drone Station Station components and uses an SSH client to connect to the Drone Proxy Host to initiate a tunnel. Software package is downloaded from the drone service. Proxy Client creates a tunnel which allows bidirectional communication with components within the Drone Staging Station and the Drone Cloud. Each Drone Charging (Staging) Station has a unique serial number generated during manufacturing.

In Scope?	Yes
Other	

Audit Database	
Description	The Audit Database used by the Report Server and is a series of tables in a PostgreSQL database. Tables are named by the date that they cover - a table for 20220814 exists for all events that are written to the queue on August 14, 2022. Every day, at 1am local time, a new table is created for the following day. The 20220814 table was created on 2022-08-13 at 1am. This ensures that the table exists well in advance of when it is needed. Tables have the columns: ● Datetime - when the event occurred ● Event ID - unique ID of the event that occurred ● Service ID - identity of the service that created this event ● User's Name - identity of the user that created this event ● Event Data - any additional data related to the event Primary key is a combination of the datetime, eventid, serviceid, and userid. There is also an Event Translation Table, which is used to translate Event ID to strings for easier human consumption. For example: ● Event 503 is "Drone Online service going offline for maintenance" ● Event 504 is "Drone Online service restarted" Events older than 30 days are automatically deleted from the PostgreSQL database. This purge occurs once per day, at 2am local time.
In Scope?	Yes
Other	Audit events are written to the Audit Database. The User's Name for the event is retrieved using the User ID from an API token, guaranteeing that the user was logged in at the time of the event.

Audit Message Queue	
Description	All Services communicate with the Audit Message Queue to post events in realtime. This data is saved to the Audit Database and is used by the Report Server to generate various reports. Messages are short lived in the queue as they are retrieved by the Report Server. ● Datetime - when the event occurred ● Event ID - unique ID of the event that occurred ● Service ID - identity of the service that created this event ● User’s Name - identity of the user that created this event ● Event Data - any additional data related to the event
In Scope?	Yes
Other	

Report Server	
Description	The Report Server pulls data from the Audit Queue, processes it and commits it to its database. When requested, data is retrieved from its database and used to query the service database for additional relevant data.
In Scope?	Yes
Other	The report server runs on dedicated hardware, to ensure that it does not impact other services.

Drone Cloud Trust Boundary

Description	Drone Online's systems are designed to be able to run anywhere - our own datacenter, hosted services, etc. The edge of the network is protected with a firewall, WAF, and IP filters on the router. Internal systems are given RFC 1918 addresses. Connections from outside the network are explicitly blocked both by the firewall and the router. The Drone Online IT department manages these devices and is responsible for keeping them fully patched, and the firewall & routing rules fully up to date. The WAF has security rulesets enabled to protect against all of the OWASP Top 10 and SANS Top 25 vulnerabilities.
In Scope?	Yes

Data Flow Annotations

1. Consumer to Forderly

Description	This interaction is outside the scope of this design. The customer interacts with Forderly to place an order and select their preferred delivery site.
Encryption	Out of scope
Authentication	Out of scope
Other	The Forderly systems are an Angular application that calls the Drone Online APIs to enable Drone-based order delivery for their customers. During onboarding, Forderly will generate an API token, automatically submit their restaurants and end users for drone delivery as part of the initial service roll out. Once this onboarding is completed, and Forderly will add drone delivery as a delivery option in its web order form.

2. Consumer to User Service

Description	The single page web app (Drone Online WWW) interacts with the User Service on behalf of the end user. Both are served from the same URI. The diagram shows the arrow going through the Drone Online trust boundary, but this is inaccurate. This connection does not pass through Drone Online.
Encryption	TLS 1.3
Authentication	See API description
Other	● All HTTP requests are redirected to HTTPS ● All cookies have the HTTPOnly and Secure flags set ● The following HTTP headers are set: ○ Strict-Transport-Securiity: max-age=2419200; includeSubDomains; preload ○ X-Frame-Options: sameorigin ○ Cache-Control: no-store

3. Consumer to Drone Online WWW

Description	Consumers download the Drone Online WWW single page web application from a CDN. The site itself is static content, and interacts with the APIs exposed by the User Service to the Internet.
Encryption	TLS 1.3
Authentication	Server: Certificate User: Password & SMS OTP

4. Restaurant Staff to DroneOnline WWW

Description	See “Consumer to Drone Online WWW” annotation. This functions the exact same way.

5. Restaurant Staff to User Service	
Description	See “Consumer to Drone Online WWW”. Details are the same.

6. Restaurant Staff to Restaurant Service	
Description	Restaurant staff will use the Restaurant Service to submit onboarding requests and once onboarding to see status on the on-site Drone Staging Station and status of drones. A future version of the restaurant service will provide additional information, such as historical deliveries, uploading customer lists for bulk account creation.
Encryption	TLS 1.3
Authentication	See Restaurant Service section

7. Restaurant Staff to Report Server	
Description	Restaurant staff can run canned reports with parameters (see the Report Server for additional details) to pull audit data and usage metrics. Reports are requested and downloaded through the Drone Online WWW.
Encryption	TLS 1.3
Authentication	See API section
Other	The Report Server prevents duplicate reports, to prevent wasted resources when a user clicks the “generate report” button on the Web UI multiple times. When a new request comes in for a report, it is checked against existing in-progress reports to ensure that it is not a duplicate. Both which canned report and the unique parameters for the report (e.g. the user to audit) are checked.

8. Forderly to Drone Service	
Description	Restaurants can check the status of drone deliveries using the Drone Service using the `/drone/` API endpoints..
Encryption	TLS 1.3
Authentication	See API description

9. Forderly to Order Service	
Description	While completing an order from a customer, Restaurant calls `/order/deliveryOptions` to get the list of delivery options available for that customer and reserve a delivery time and a drone for the delivery. Restaurant should then let the customer select the delivery option. Restaurant must then call `/order/order` within 5 minutes to select the destination and commit the delivery drone.
Encryption	TLS 1.3
Authentication	See API section

10. Order Service to User Service	
Description	When Restaurant calls `GET /order/deliveryOptions` on Order Service, Order Service calls `GET /user/deliverySite` to get the list of delivery options defined for that user, retrieved from the service database.
Encryption	TLS 1.3
Authentication	See API description

11. Restaurant Service to Drone Service	
Description	Restaurants can check the status of drone deliveries using the Drone Service via the `/drone/` API endpoints.
Encryption	TLS 1.3
Authentication	See API section

(Not shown) All Services to Audit Queue	
Description	Every service publishes events to the Audit Queue using an internal shared library that provides an `audit(serviceid, eventid, API Token, additionaldata).` The library pulls the current time from the local system and delivers the data to the audit queue.
Encryption	TLS 1.3
Other	

12. Audit Queue to Report Server	
Description	The Report Server polls the Audit message queue looking for events.
Encryption	TLS 1.3
Authentication	User/Pass created as part of environment creation, stored in configuration on the Report Server file system. User has a specialized role to read data from the queue.

13. Order Service to Drone Service	
Description	Order service interacts with the drone service to ● Reserve drones deliveries ● Cancel drone deliveries ● Proxy manual control of drones from the User Service to the Drone Service ● Retrieve drone status ● Remove damaged drones from the available delivery fleet ● Add repaired drones back to the available delivery fleet Order service uses a combination of externally- and internally- exposed APIs. The externally-exposed APIs are documented in the APIs section of this document. This interaction is entirely inside the Drone Online network..
Encryption	HTTP
Authentication	IP Filtering
Other	See API section for externally-exposed APIs.

16. Drone Proxy Client & Drone Proxy	
Description	Two SSH tunnel are initiated; one opens a port on the Drone Proxy Host which the Drone Service connects to that tunnels back to the Drone Staging Station. Another opens a local port on the Drone Proxy Client that the Staging Station uses to connect to the Drone Service. Using this architecture no changes are required on the Drone Staging Stations or Drone Service to support relocating the Drone Staging Stations outside the Drone Online Network.
Encryption	SSHv2, aes-ctr
Authentication	sshpass otp from proxy configuration
Other	.

14->18 Drone Service to Drone Staging Station (via ProxyHost/Client Tunnel)	
Description	Drone Service communicates with Drone Proxy to interact with charging stations via the secure tunnel.
Encryption	Tunnel is encrypted via SSH
Authentication	SSH Auth
Other	Drone Staging Station specific functionality is out of scope.

17->15 Drone Staging Station to Drone Service (via ProxyHost/Client Tunnel)	
Description	Drone Staging Station communicates with the Drone Proxy Client to interact with the Drone Service.
Encryption	Tunnel is encrypted via SSH
Authentication	SSH Auth
Other	Drone Staging Station specific functionality is out of scope.

19. DroneOnline Support User to Bastion Host	
Description	See “Bastion Host” section for details about how Drone Online support staff access the host.
Encryption	SSHv2 configured to accept only `aes256-ctr`

Authentication	SSH key

20. Proxy Client to Drone Service	
Description	The Proxy Client accesses the Drone Service to retrieve its configuration data or binary/script data used with tunnels.
Encryption	TLS 1.3
Authentication	serialno

21. Report Server to Audit Database	
Description	Report Server reads and writes data to the Audit Database. The RS will query the Audit Database and the Service Database when generating reports.
Encryption	Data encrypted at rest
Authentication	User/Pass created at env creation, stored in report server deploy configuration.