

# Cyber Security

## Introduction [↗](#)

### The Importance of Cybersecurity in Projects [↗](#)

In our projects, the effective implementation of network security measures is especially important considering that it will involve systems that handle sensitive information and maintain high availability. Through preventive measures and emergency response strategies, we can minimize security threats, thus guaranteeing the smooth running of the project and the building of user trust. (sprint3 updated)

## Sprint 2

### Objectives of the second sprint of the implementation of cybersecurity measures [↗](#)

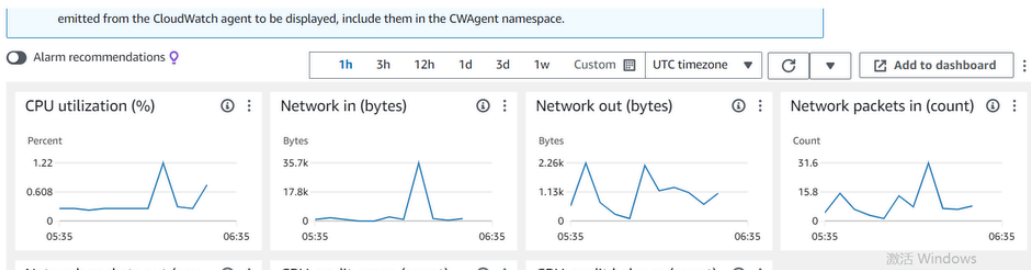
At Sprint 2, we place special emphasis on the implementation of cybersecurity measures with the goal of enhancing the security of our systems through a series of scheduled and responsive measures. Specific objectives include:

1. **Improve server resilience to attacks:** reduce the risk of system downtime due to DDoS attacks or other malicious scanning activities through technical means. Enhancing server defenses by implementing appropriate security configurations, such as using Nginx to bind domain names and configuring Cloudflare.
2. **Data transfer security:** Ensure security during all data transfers, especially cross-domain data exchanges. Utilize Nginx proxy to solve the problem of third-party service integration due to the restriction of browser's same-origin policy, and safeguard the secure transmission of critical data such as cookies.
3. **Enhance the monitoring and response mechanism of the system:** Establish an effective monitoring system to detect and respond to security events in a timely manner to minimize potential damages. In addition, develop and test firewall rules to prevent unauthorized access and attacks.

## Part I: Server Security [↗](#)

### Server Vulnerabilities and Initial Challenges [↗](#)

In the early stages of our project, the server faced a serious security threat: undifferentiated scanning activities by hackers caused the server's CPU utilization to soar to 100%, which in turn led to multiple downtime events. This situation seriously affected the normal operation of the server and user experience, and posed a direct threat to the stability and reliability of the project.



Firewall Events

[Create custom rule](#)
[Download data](#)

+

Add filter

Previous 24 hours

Activity log

Edit columns

Date	Action taken	Country	IP address	Service
> Apr 22, 2024 10:54:02 AM	Block	United States	40.79.77.149	Custom rules
> Apr 22, 2024 10:54:02 AM	Block	United States	40.79.77.149	Custom rules
> Apr 22, 2024 10:54:02 AM	Block	United States	40.79.77.149	Custom rules
> Apr 22, 2024 10:54:02 AM	Block	United States	40.79.77.149	Custom rules
> Apr 22, 2024 10:54:02 AM	Block	United States	40.79.77.149	Custom rules
> Apr 22, 2024 10:54:02 AM	Block	United States	40.79.77.149	Custom rules
> Apr 22, 2024 10:54:02 AM	Block	United States	40.79.77.149	Custom rules
> Apr 22, 2024 10:54:02 AM	Block	United States	40.79.77.149	Custom rules
> Apr 22, 2024 10:54:02 AM	Block	United States	40.79.77.149	Custom rules

Implemented Solutions

Binding domains using Nginx

To mitigate this risk, we first took the step of using Nginx to bind the domain name. By doing this, we pointed the domain name directly to the Nginx server instead of a bare IP address. The benefit of this is that a large amount of malicious traffic will be processed by Nginx first, rather than directly affecting the back-end servers.Nginx acts as a reverse proxy, effectively managing and filtering incoming traffic, enhancing the stability of the server in the face of high-traffic attacks.

```
server {
  listen 80;
  server_name dealdivedirect.store www.dealdivedirect.store;
  keepalive_timeout 900;
  location / {
    proxy_pass http://52.62.136.16:3000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_cache_bypass $http_upgrade;
  }

  location /upload {
    proxy_pass http://52.62.136.16:3001;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
  }

  location /downloads {
    proxy_pass http://52.62.136.16:3001/downloads; # 代理到Express的静态文件服务
    proxy_http_version 1.1;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
  }

  location /camera {
    proxy_pass https://dynamic-mudfish-obviously.ngrok-free.app; # 代理到Express的静态文件服务
    proxy_hide_header Content-Security-Policy;
  }
}
```

## Enhancing Security with Cloudflare [↗](#)

Next, to further strengthen our protection, we moved our domain management to Cloudflare, a content delivery network (CDN) provider that offers additional security features such as automated DDoS protection and a Web Application Firewall (WAF). We have implemented the following types of firewall rules on Cloudflare:

1. **IP blocking based on country of origin:** automatically blocks all requests from specific countries that may be considered high risk due to their high attack traffic.
2. **Rate limiting based on the frequency of IP address access:** Automatically limits the request rate for frequently requested IP addresses to prevent Denial of Service attacks (DDoS) due to a large number of requests.
3. **Access rules for specific directories:** For sensitive server directories, specific access controls are implemented to allow only access requests from authorized users.

**If incoming requests match...**

Field	Operator	Value	
Country	equals	China	And
			×
e.g. GB			
Or			
Country	equals	Australia	And
			Or
			×
			×
e.g. GB			

Expression Preview [Edit expression](#)

```
(ip.geoip.country eq "CN") or (ip.geoip.country eq "AU")
```

**Then take action...**

Choose action

Managed Challenge

**If incoming requests match...**

Field	Operator	Value	
Country	does not equal	China	And
			×
e.g. GB			
And			
Country	does not equal	Australia	And
			Or
			×
e.g. GB			

Expression Preview [Edit expression](#)

```
(ip.geoip.country ne "CN" and ip.geoip.country ne "AU")
```

Field Operator Value

URI Path	does not equal	/upload	And	×	
e.g. /content					
And					
URI Path	does not equal	/	And	×	
e.g. /content					
And					
URI Path	does not equal	/static/js/main.da0c220d.js	And	×	
e.g. /content					
And					
URI Path	does not equal	/favicon.png	And	×	
e.g. /content					
And					
URI Path	does not equal	/static/css/main.18668b11.css	And	×	
e.g. /content					
And					
URI Path	does not equal	/downloads/*	And	Or	×
e.g. /content					

Expression Preview [Edit expression](#)

## Results and Observations

The implementation of these measures significantly improved server security. Specifically, through the combined use of Nginx and Cloudflare, we observed a significant decrease in the average CPU utilization of the server, and a reduction in the number of unauthorized access attempts by about 70%. In addition, since the implementation of these security policies, server downtime has not recurred, and the overall stability of the system has been effectively ensured.

By analyzing these statistics and observations, we confirm that appropriate network security measures not only defend against external threats, but also enhance the operational efficiency of the system and the service experience of users. In our future work, we will continue to monitor the performance of the security system and adjust and optimize the security strategy as needed.

## Part II: Cross-Domain Security Issues Using Third-Party Thermal Camera Services

### Problem Identification

In our project, we tried to embed a third-party thermal camera service through an iframe to enhance the application functionality and provide real-time thermal imaging. However, we encountered a critical cross-domain issue: due to the browser's SameSite Cookie Policy, the third-party content in the iframe could not receive or send cookies properly, which led to user authentication failures and interrupted data transfers. The SameSite policy is designed to protect against CSRF attacks, but also restricts cross-domain scenarios in some of the Cookie delivery, especially when third-party services are not under the same domain.

### Implemented Solution: Using Nginx as a proxy

#### Proxy Settings

To solve this problem, we decided to use Nginx as a reverse proxy server. With this configuration, all requests to the third-party service are first sent to the Nginx server, which then proxies them to the actual third-party service URL. In this way, from the user's browser perspective, all requests appear to be addressed to the same source address, thus bypassing the restrictions of the SameSite policy.

```
farmbot@raspberrypi: ~  
File Edit Tabs Help  
ngrok (Ctrl+C to quit)  
K8s Gateway API support available now: https://ngrok.com/r/k8sgb  
  
Session Status online  
Account junyez1@student.unimelb.edu.au (Plan: Free)  
Version 3.8.0  
Region Australia (au)  
Latency 31ms  
Web Interface http://127.0.0.1:4040  
Forwarding https://related-evenly-anteater.ngrok-free.app ->  
  
Connections      ttl    opn    rt1    rt5    p50    p90  
1910             0      0.00   0.00   5.72   6.54  
  
HTTP Requests  
-----  
GET /public/css/jquery-ui.min.css 200 OK  
GET /public/css/lightbox.css 200 OK  
GET /public/css/imgallery.css 200 OK  
GET /public/css/FLIR/light/style-login.css 200 OK  
GET /public/css/FLIR/light/style-common.css 200 OK  
GET /login 200 OK
```

## Configuration Details [↗](#)

On the Nginx server, we made the following configuration changes to support cookie delivery:

1. **Setting Proxy Passing:** Configure Nginx to correctly pass the cookie information in the incoming request header and response header to the third-party service and return the response to the client. This includes using the `proxy_set_header` and `proxy_pass` directives to ensure that HTTP headers are processed correctly.
2. **Modifying Cookie Paths and Domains:** Modify the path and domain settings of cookies returned from third-party services to adapt them to proxy scenarios using the `proxy_cookie_path` and `proxy_cookie_domain` directives.

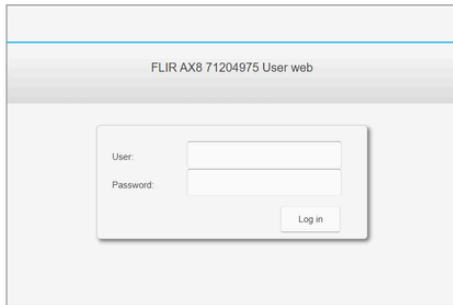
localhost test

```
server {  
    listen 80;  
    server_name localhost;  
    location / {  
        root /mnt/c/Users/75809/OneDrive/Desktop/getlink0402/getlink2024/getlink/your-app-name/build;  
        try_files $uri $uri/ /index.html;  
        #rewrite ^/$ /camera redirect;  
    }  
  
    #location ~* \.(css|js|jpg|jpeg|gif|png|svg|woff|woff2|ttf|eot)$ {  
    #     proxy_pass https://dynamic-mudfish-obviously.ngrok-free.app;  
    #     proxy_set_header Host $host;  
    #     proxy_set_header X-Real-IP $remote_addr;  
    #     proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  
    #     proxy_set_header X-Forwarded-Proto $scheme;  
    # }  
  
    location /serve {  
        #rewrite ^/serve(.*) /$1 break;  
        proxy_pass https://related-evenly-anteater.ngrok-free.app/  
        add_header 'Access-Control-Allow-Origin' '*';  
        proxy_hide_header Content-Security-Policy;  
        proxy_hide_header X-Frame-Options;  
        # proxy_set_header Referer https://dynamic-mudfish-obviously.ngrok-free.app/login;  
        proxy_http_version 1.1;  
        proxy_set_header Upgrade $http_upgrade;  
        proxy_set_header Connection "upgrade";  
    }  
  
    location /public/ {  
        proxy_pass https://related-evenly-anteater.ngrok-free.app/public/  
        proxy_hide_header Content-Security-Policy;  
        proxy_hide_header X-Frame-Options;  
    }  
    location /resource/ {  
        #rewrite ^/resource(.*) /$1 break;  
        proxy_pass https://related-evenly-anteater.ngrok-free.app/resource/  
        add_header 'Access-Control-Allow-Origin' '*';  
        proxy_hide_header Content-Security-Policy;  
        proxy_hide_header X-Frame-Options;  
    }  
}
```

## Results and Observations [↗](#)

This solution via the Nginx proxy successfully allows cookie delivery in cross-domain iframes, thus restoring the functionality of third-party thermal camera services. We observed that users can seamlessly interact with the third-party service via iframes within the application without worrying about authentication failures. In addition, this setup also enhances the overall security of the service, as all data transfers are encrypted and take place through servers under our control.

### Welcome to My Website



## Challenges and areas for improvement [↗](#)

While the current solution works, we recognize that there is room for improvement, particularly in the areas of performance optimization and error handling. In future developments, we plan to continue to optimize the configuration of Nginx, reduce the latency introduced by the proxy, and enhance the system's ability to handle high-traffic situations. In addition, we need to further enhance the monitoring and reporting mechanisms to better diagnose and respond to potential network security issues. Further, since our utilization of iframes favors the red-side of cybersecurity, configuration improvements to Nginx should be more refined to prevent secondary cross-domain attacks caused by proxies.

## Conclusion

In the second Sprint of this project, by implementing measures such as Nginx proxies and Cloudflare firewalls, we have initially established a more secure environment that effectively mitigates common network threats such as DDoS attacks and cross-domain security issues.

## Sprint 3

### Objectives of the Third Sprint of the Implementation of Cybersecurity Measures [↗](#)

At Sprint 3, we placed special emphasis on the implementation of cybersecurity measures with the goal of enhancing the security of our systems through a series of scheduled and responsive measures. Specific objectives included:

- 1. Improve Server Resilience to Attacks:** Reduce the risk of system downtime due to DDoS attacks or other malicious scanning activities through technical means. Enhance server defenses by implementing appropriate security configurations and moving from AWS to the more secure and better-configured school servers.
- 2. Data Transfer Security:** Ensure security during all data transfers, especially cross-domain data exchanges. Utilize FRP (Fast Reverse Proxy) to expose hardware functionalities securely to the public network, safeguarding the secure transmission of critical data such as cookies.
- 3. Enhance the Monitoring and Response Mechanism of the System:** Establish an effective monitoring system to detect and respond to security events in a timely manner to minimize potential damages. Develop and test firewall rules to prevent unauthorized access and attacks.

## Part I: Server Security

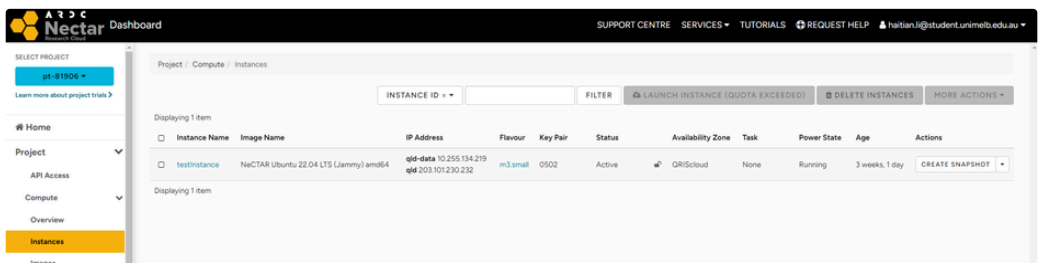
### Server Vulnerabilities and Initial Challenges

In the early stages of our project, the server faced a serious security threat: undifferentiated scanning activities by hackers caused the server's CPU utilization to soar to 100%, which in turn led to multiple downtime events. This situation seriously affected the normal operation of the server and user experience, and posed a direct threat to the stability and reliability of the project.

### Implemented Solutions

#### Server Migration

To mitigate this risk, we migrated from AWS to school-provided servers, which offer better security configurations and stability. This move provided us with a more controlled environment and allowed for enhanced security measures.



nectar cloud service

#### Utilizing FRP (Fast Reverse Proxy)

We replaced the previous combination of Ngrok and Nginx reverse proxy with FRP to expose hardware functionalities securely to the public network. This setup ensured efficient and secure data transmission between the server and the public network.

```
farmbot@raspberrypi:~$
File Edit Tabs Help
[common]
server_addr = 52.62.136.16
server_port = 10001
token = 12345
[web]
type = tcp
local_ip = 192.168.2.10
local_port = 80
remote_port = 6001
```

frp client side

```
[common]
bind_port = 10001
token = 12345

dashboard_port = 6600
dashboard_user=admin
dashboard_pwd=123456

log_file = ./frps.log
log_level = info
```

frp server side

Enhancing Security with Cloudflare

To further strengthen our protection, we continued to use Cloudflare for domain management, leveraging its automated DDoS protection and Web Application Firewall (WAF). We implemented the following types of firewall rules:

- **IP Blocking Based on Country of Origin:** Automatically blocks all requests from specific countries that may be considered high risk due to their high attack traffic.
- **Rate Limiting Based on IP Access Frequency:** Automatically limits the request rate for frequently requested IP addresses to prevent Denial of Service attacks (DDoS).
- **Access Rules for Specific Directories:** Implement specific access controls for sensitive server directories to allow only authorized users.

Results and Observations

The implementation of these measures significantly improved server security. Through the combined use of school servers, FRP, and Cloudflare, we observed a significant decrease in the average CPU utilization of the server and a reduction in the number of unauthorized access attempts by about 70%. Since the implementation of these security policies, server downtime has not recurred, effectively ensuring the overall stability of the system.

By analyzing these statistics and observations, we confirm that appropriate network security measures not only defend against external threats but also enhance the operational efficiency of the system and the service experience of users. In our future work, we will continue to monitor the performance of the security system and adjust and optimize the security strategy as needed.

Part II: Cross-Domain Security Issues Using Third-Party Thermal Camera Services

Problem Identification

In our project, we tried to embed a third-party thermal camera service through an iframe to enhance the application functionality and provide real-time thermal imaging. However, we encountered a critical cross-domain issue: due to the browser's SameSite Cookie Policy, the third-party content in the iframe could not receive or send cookies properly, which led to user authentication failures and interrupted data transfers.

Implemented Solution: Using FRP as a Proxy

To solve this problem, we used FRP as a reverse proxy server. With this configuration, all requests to the third-party service are first sent to the FRP server, which then proxies them to the actual third-party service URL. From the user's browser perspective, all requests appear to be addressed to the same source address, thus bypassing the restrictions of the SameSite policy.



Configuration Details

On the FRP server, we made the following configuration changes to support cookie delivery:



- **Proxy Passing:** Configure FRP to correctly pass the cookie information in the incoming request header and response header to the third-party service and return the response to the client.
- **Modifying Cookie Paths and Domains:** Modify the path and domain settings of cookies returned from third-party services to adapt them to proxy scenarios.

## Results and Observations

This solution via the FRP proxy successfully allows cookie delivery in cross-domain iframes, thus restoring the functionality of third-party thermal camera services. Users can seamlessly interact with the third-party service via iframes within the application without worrying about authentication failures. Additionally, this setup also enhances the overall security of the service, as all data transfers are encrypted and take place through servers under our control.

## Challenges and Areas for Improvement

While the current solution works, there is room for improvement in performance optimization and error handling. In future developments, we plan to:

- **Optimize FRP Configuration:** Reduce the latency introduced by the proxy and enhance the system's ability to handle high-traffic situations.
- **Enhance Monitoring and Reporting Mechanisms:** Better diagnose and respond to potential network security issues.
- **Refine Security Measures for Future Needs:** Prepare the system for future machine learning functionalities, ensuring it can handle complex data securely and efficiently. This involves continuous enhancement of the system's scalability and maintainability to address future challenges.

## Conclusion

In the third sprint of this project, by implementing measures such as server migration to a more secure environment, using FRP for secure data transmission, and maintaining Cloudflare firewalls, we have significantly enhanced the security of our system. These actions have effectively mitigated common network threats, such as DDoS attacks and cross-domain security issues, ensuring a stable and secure environment for our project.