# Machine Learning Method for C&C Server Detection Based on Bot Behavior in an ISP Network

Masataka Nakahara
*KDDI Research, Inc.*
*KDDI CORPORATION*
Saitama, Japan
ms-nakahara@kddi.com

Kosuke Murakami
*KDDI Research, Inc.*
*KDDI CORPORATION*
Saitama, Japan
ko-murakami@kddi.com

Ayumu Kubota
*KDDI Research, Inc.*
*KDDI CORPORATION*
Saitama, Japan
ay-kubota@kddi.com

***Abstract*—Given the increasing threat of cyberattacks facilitated by malware-infected devices, developing effective methods for the early detection of command and control (C&C) servers, which can issue malicious commands to infected devices, is essential. In this paper, we propose a novel detection approach that uses flow data collected from internet service provider (ISP) networks. Typically, normal traffic constitutes the majority of network communications, making it challenging to identify communications between C&C servers and infected devices. This challenge is particularly severe in networks operated by an ISP, where the ratio of malicious communication is very low.**

**To address this, we developed a detection method that combines heuristic analysis based on known C&C server communications with machine learning. The proposed method enables the precise detection of C&C servers from ISP flow data by narrowing down potential targets through heuristic analysis based on flow data characteristics and then applying machine learning. We evaluate the effectiveness of the proposed method through experiments using real flow data from actual ISP networks and known C&C server information. As a result, the performance of the detection has been enhanced. For instance, the F1-score has increased from 0.583 to 0.802.**

***Index Terms*—C&C server, flow data, machine learning, random forest, lightgbm**

## I. Introduction

Cyberattack activity has been increasing rapidly, posing significant threats to service providers through various malicious techniques such as malware exploitation and unauthorized access. For internet service providers (ISPs), malware-infected devices, especially those involved in distributed denial of service (DDoS) attacks, threaten service continuity and infrastructure stability. Prior studies (e.g., Rahman et al. [1]) have explored detection solutions from the perspective of the ISP. Various approaches have been employed to co counteract the vulnerabilities exploited by cyberattacks, including the deployment of web application firewalls (WAFs), intrusion prevention systems (IPSs), and access control measures. Additionally, efforts are being made to identify infected devices and alert their users to prevent attacks from malware-infected terminals [2].

A key approach to mitigating these threats involves the identification of command and control (C&C or C2) servers, which issue commands to infected devices (bots). Detecting these servers allows the mitigation of botnet activities, helping to reduce the incidence of malicious activities. Traditional detection methods include honeypot-based malware analysis and sandbox execution, which observe C2 communication behaviors directly from malware samples or infected hosts. However, analyzing individual malware samples or bots limits the detection scope and thus may fail to capture broader network-level patterns, while scalability to large networks may be difficult. Consequently, flow data from telecommunications networks have been investigated in the development of methods to detect C2 servers. Flow data consist of timestamped communication packets with source/destination IP addresses, ports and protocols and provide a scalable means to analyze network behavior across a large amount of traffic. These data enable the identification of anomalous communication patterns that may be indicative of C2 activity without the need for detailed inspection of each malware sample or infected host.

This paper proposes a method for detecting C2 servers using flow data collected from ISP networks. Since the majority of communications through ISP networks are benign, identifying the small fraction of C2 traffic within these data is a significant challenge. While machine learning techniques are frequently employed in similar tasks, supervised methods face problems such as a class imbalance that favors normal traffic, whereas unsupervised methods face difficulties in modeling the diverse behaviors of legitimate communications. To address these challenges, we introduce a hybrid approach that combines heuristic analysis based on the behaviors of known C2 servers with machine learning. Initially, flow data are filtered using open-source intelligence (OSINT) sources to identify devices communicating with known C2 servers, which are likely to be compromised bots. The communication destinations of these devices are then classified using machine learning. Furthermore, since bots often perform host scans to locate new infected devices, communication destinations associated with host scanning activities are excluded to increase detection accuracy. Our evaluation, conducted on real flow data collected from operational ISP networks, demonstrates

that the combined heuristic and machine learning approach, combined with the exclusion of scan-related communications, results in improved detection precision and recall. The main contributions of this paper are as follows:

- A novel method for detecting C2 servers based on ISP network flow data.
- A hybrid approach that combines heuristic analysis and machine learning to address data imbalance.
- An exclusion strategy for host scan targets to improve detection accuracy.

## II. Related Works

The detection of C2 servers and botnets has been extensively studies through a range of analytical methods, including flow data-based methods, packet analysis and malware binaries. This section reviews relevant works investigating C2 server detection.

Numerous approaches have been proposed for detecting C2 servers. Al Leleh et al. [3], [4] developed methods for identifying C2 servers on cloud platforms by extracting features from portable executable (PE) files, utilizing machine learning techniques such as decision tree (DT), random forest (RF), and naive Bayes (NB). They used malware samples from VirusTotal and benign samples from software repositories to generate features such as API call frequencies, file operations, and communication logs via sandbox analysis.

Another prominent approach involves domain-based detection. Quezada et al. [5] generated domain name server (DNS) log fingerprints to detect bot infections and employed an isolation forest method for anomaly detection and domain generation algorithm (DGA)-based domain query analysis. Biros et al. [6] used convolutional neural network (CNN) and long short-term memory (LSTM) models to detect DGA domains. Koga et al. [7] extracted DNS queries and applied Word2Vec to create feature vectors for detecting malicious domains.

Graph-based methods have also been proposed. Huang et al. [8] constructed directed graphs from payload length sequences to model host communication behaviors.

Vidhun et al. [9] used flow duration and packet size features from DNS queries with CNNs to detect C2 servers via a flow-based approach. Ramos et al. [10] focused on detecting C2 channels concealed within Cobalt Strike traffic, extracting features such as session counts and durations and applying classifiers based on algorithms such as RF, NB, neural networks, support vector machines (SVMs), and k-means. Baruah et al. [11] targeted P2P botnet flows; extracted features such as flow duration, IP addresses, ports, type of service (TOS), packet counts, and sizes; and employed ensemble methods for C2 server detection. Focusing on banking malware, Kazi [12] employed flow characteristics from PCAP data with classifiers based on algorithms such as DT, RF, and K-nearest neighbor (KNN). Merkli et al. [13] proposed tree-based classifiers evaluated on the Locked Shields dataset.

Most prior works, however, evaluated their proposed methods on limited datasets or small-scale networks. In contrast, ISP networks involve large numbers of devices and exhibit a significant class imbalance between normal and C2 traffic, limiting the efficacy of existing methods. In this paper, a hybrid approach combining heuristic analysis based on C2 servers and bot behaviors with machine learning is proposed to improve detection performance.

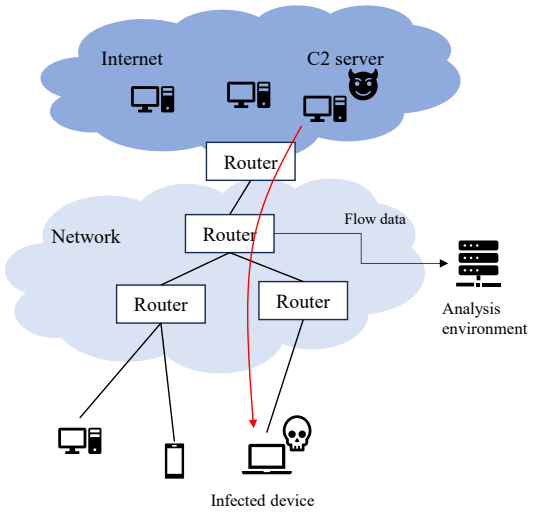


Fig. 1. Flow Data Acquisition System

## III. Flow Data Acquisition System

This section describes the system configuration for the flow data acquisition performed in this study and the format of the acquired flow data.

### A. System Configuration

The overall architecture of the flow data acquisition system is shown in Figure 1. Flow data were collected by mirroring network traffic passing through routers installed in the operational network of the ISP. These routers captured and duplicated packets, which were then forwarded to the analysis environment for processing. The captured traffic includes communications originating from and destined for terminals within the ISP's autonomous system (AS), as well as inter-AS communications passing through the same network infrastructure. These data were used strictly for detecting C2 servers and not for any other analyses.

The flow data were aggregated over five-minute intervals into a single file, where each record corresponds to one flow, containing relevant information such as source/destination IP addresses, ports, protocols, TCP flags, packet counts, and byte counts. Owing to the high volume of network traffic, storing and analyzing all flows continuously was impractical. Therefore, during data collection, the data were resampled at a rate of 1/1000 or lower. This means that only a subset of packets were recorded, reducing the data size and facilitating large-scale analysis. However, this sampling introduces limitations; for example, some communication flows, particularly those

TABLE I
FLOW DATA FORMAT

| No. | Field Name | Description |
|---|---|---|
| 1 | ts | Start Time |
| 2 | te | End Time |
| 3 | td | Duration |
| 4 | sa | Source IP Address |
| 5 | da | Destination IP Address |
| 6 | sp | Source Port Number |
| 7 | dp | Destination Port Number |
| 8 | pr | Protocol |
| 9 | flg | TCP Flag |
| 10 | ipkt | Packet Count |
| 11 | ibyt | Byte Count |

involving short-lived or low-volume exchanges, may not have been captured.

### B. Flow Data Format

The flow data were processed to extract only essential fields, as listed in Table I. The data fields include timestamps, IP addresses, ports, protocol information, TCP flags, and packet and byte counts. Packet count is the number of packets included in the flow. Byte count is the total number of bytes in the packets included in the flow. Since payload contents are not included, compared with PCAP data, the available information is limited, but this approach reduces storage requirements and allows the analysis of large amounts of network traffic.

## IV. C&C SERVER DETECTION METHOD

This section describes the method for detecting C2 servers from flow data in ISP networks, which is the main focus of this paper.

### A. Detection Procedure

The procedure for detecting C2 servers is illustrated in Figure 2. Briefly, the method involves extracting bot IP addresses from flow data using known C2 server information and classifying the communication destinations of the bots with a machine learning model. First, flow data were acquired as described in Section III. The analysis server specifies the target exporter and date from which to read the flow data. Additionally, OSINT information is used to obtain known C2 IP addresses and port numbers. By acquiring more known C2 IP addresses, more bot terminals can be extracted, thereby increasing the number of detectable C2 servers as communication destinations. In this experiment, three threat intelligence sources, namely, Triage [14], FeodoTracker [15], and ThreatFox [16], were used to obtain known C2 information. Triage is an online sandbox that allows users to submit malware samples for dynamic analysis, providing information such as sample files, risk scores, and malware configurations. The malware configuration includes the IP addresses, domains, and port numbers of C2 servers, which are utilized as known C2 information in this method. FeodoTracker specializes in specific malware families such as Emotet and QakBot, summarizing

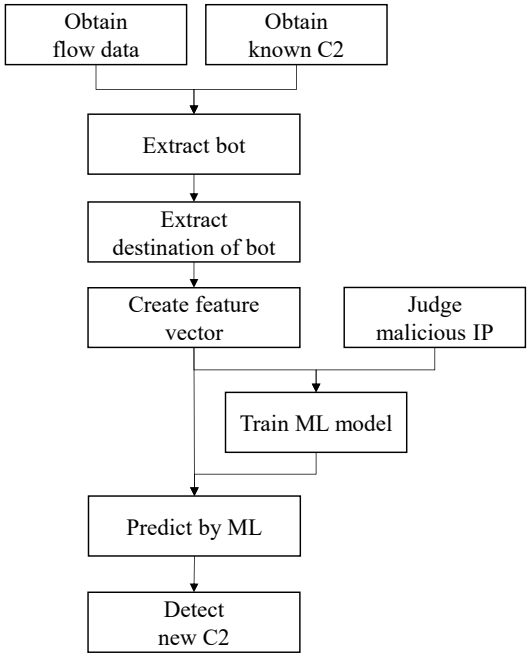


Fig. 2. C2 Server Detection Procedure

the IP addresses, port numbers, and operational statuses of C2 servers, thus providing usable data as known C2 information. ThreatFox compiles indicators of compromise (IoC) provided by the information security community and antivirus vendors into a database, allowing the retrieval of C2 server URLs, IP addresses, port numbers, and malware families. This system collects data daily from these intelligence sources and accumulates IP addresses and port numbers.

C2 servers often have limited lifespans. Therefore, the OSINT information used as known C2 information was limited to data from up to 30 days prior to the date of the flow data being analyzed.

### B. Feature Creation

Next, flows with destination IP address-port pairs included in the known C2 information are extracted from the flow data obtained from the communication network. The sources of the extracted flows are likely to be malware-infected terminals that are receiving commands from C2 servers. Furthermore, some bots communicate with multiple C2 servers, and extracting all communications from these bots may include communications with unknown C2 servers.

Therefore, the communication destinations of these bots are treated as candidate IP addresses for C2 servers, and the information is classified to determine whether these IP addresses are legitimate or C2 servers.

From the extracted bot flows, IP address-port pairs are created, and features are generated for each pair. The features to be generated are shown in Table II. In addition to features related to flow and byte counts, features limited to specific protocols and port numbers, TCP flag features, and information such as AS numbers and country codes associated with

TABLE II
LIST OF FEATURES

| Field Name | Description |
|---|---|
| irecords | Number of incoming flows |
| orecords | Number of outgoing flows |
| ibyt_sum/std | Total/standard deviation of bytes in incoming flows |
| obyt_sum/std | Total/standard deviation of bytes in outgoing flows |
| ipkt_sum/std | Total/standard deviation of packets in incoming flows |
| opkt_sum/std | Total/standard deviation of packets in outgoing flows |
| ibyt_qt_25/50/75 | Quartiles of bytes in incoming flows |
| obyt_qt_25/50/75 | Quartiles of bytes in outgoing flows |
| ipkt_qt_25/50/75 | Quartiles of packets in incoming flows |
| opkt_qt_25/50/75 | Quartiles of packets in outgoing flows |
| in_unique_ip | Unique source IP addresses in incoming flows |
| out_unique_ip | Unique destination IP addresses in outgoing flows |
| in_unique_port | Unique source IP port numbers in incoming flows |
| out_unique_port | Unique destination IP port numbers in outgoing flows |
| in_unique_pr | Unique source IP protocols in incoming flows |
| out_unique_pr | Unique destination IP protocols in outgoing flows |
| ibyt_sum_on_pr_protocol | Total bytes in incoming flows for that protocol |
| obyt_sum_on_pr_protocol | Total bytes in outgoing flows for that protocol |
| ipkt_sum_on_pr_protocol | Total packets in incoming flows for that protocol |
| opkt_sum_on_pr_protocol | Total packets in outgoing flows for that protocol |
| iflg_flag | Number of incoming flows with that TCP flag |
| oflg_flag | Number of outgoing flows with that TCP flag |
| port_80/443/53 | Whether the port number is 80/443/53 |
| in_unique_detected_ip_before1day/2day | Number of source IP addresses in the bot extraction results from 1/2 days prior |
| out_unique_detected_ip_before1day/2day | Number of destination IP addresses in the bot extraction results from 1/2 days prior |
| in_unique_darknet_ip_before1day/2day | Number of source IP addresses in the darknet observation results from 1/2 days prior |
| out_unique_darknet_ip_before1day/2day | Number of destination IP addresses in the darknet observation results from 1/2 days prior |
| port_count | Number of times that the port number appears in the same feature file |
| ASN_count | Number of times the AS number of that IP address appears in the same feature file |
| CC_is_JP/CN | Whether the country code of that IP address is JP/CN |
| rank_is_over_5000 | Whether that IP address is within the top 5000 observed access counts from the flow |

IP addresses are used as features. AS numbers and country codes are assigned using the GeoLite2 ASN Database [17].

Moreover, terminals that communicate frequently with IP addresses identified as bots in the above procedure are likely to be C2 servers. Therefore, the number of IP addresses included in the bot extraction results from the most recent day is added as a feature. Additionally, the number of detected IP addresses is cross-referenced with those observed in the NICTER darknet from the most recent day. The darknet refers to the reachable but unused IP address space [18]. It is rare for normal usage to communicate with unused IP addresses, but packets from malware-infected terminals performing indiscriminate port scans can be observed in the darknet. Therefore, hosts that send packets to the darknet are likely to be infected with malware. Furthermore, servers that communicate frequently with these hosts are also likely to be C2 servers. Features related to incoming communications are generated by a similar method to that for generating features related to outgoing communications. Since the ranges of values for each feature differ, they are normalized to a range of 0 to 1 by min-max scaler.

Notably, labels indicating whether the IP address-port-protocol combinations are malicious are assigned to the generated features. VirusTotal [19] is used for labeling because it can determine the malignancy of IP addresses. In this paper, IP addresses determined to have a score of 1 or higher on VirusTotal are labeled malicious. VirusTotal's detection is performed on an IP basis, and the features are evaluated in terms of IP-port-protocol triplets. However, if an IP address is deemed malicious, the corresponding IP-port-protocol triplet can also be judged to be malicious. Therefore, we employ VirusTotal's IP-based detection to assign labels.

Using the generated labeled features, machine learning models are trained. This paper employs random forest (RF) [20] and LightGBM [21]. Models based on RF and LightGBM are both known to perform well in classifying tabular data. During detection, features are generated from flow data in the same manner as during training, and the trained model is used to determine the malignancy of the IP address-port-protocol combinations.

### C. Exclusion of Scanning Target Terminals

In the detection procedure described in the previous section, when the bot communication destinations as candidate C2 servers are extracted, scanning target terminals are also included among the main communication destinations of the

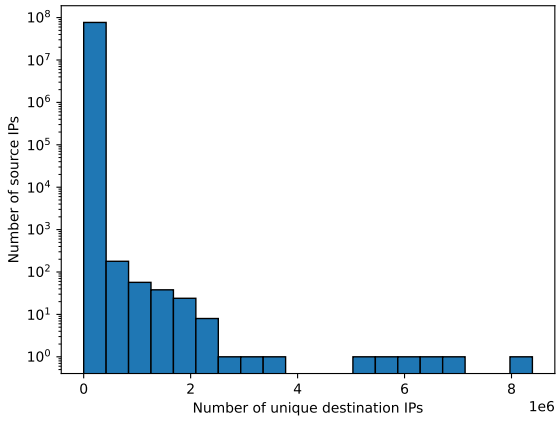


Fig. 3. Unique Number of Destinations

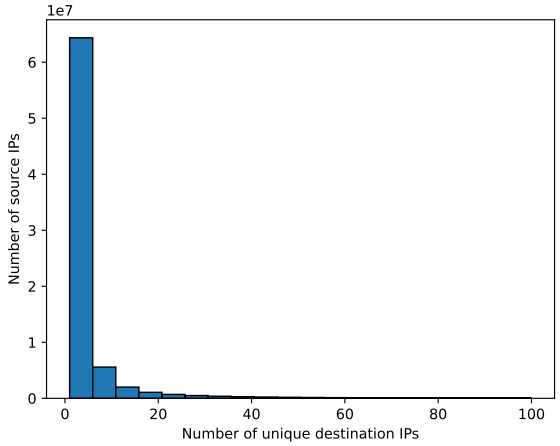


Fig. 4. Unique Number of Destinations (under 100)

bots. By excluding scanning target terminals from the bot communication destinations, the accuracy of the candidate C2 server extraction can be improved, thereby improving the detection precision. Bots engaged in scanning activities communicate with many more destinations than legitimate terminals do. Therefore, bots with a unique destination IP address count exceeding a certain threshold are defined as scanning activity bots, and their communication destinations are excluded from the candidate C2 servers.

To determine the threshold count, we calculated the unique number of destination IP addresses observed in the ISP network on May 2, 2024, which is the same date as the data used for the subsequent evaluation. The results are shown in Fig. 3 and 4. These figures indicate that the number of unique destination IP addresses for most clients is fewer than 10. The total number of source IP addresses is 76,868,329. Among these, 69,274,185 have fewer than 10 unique destination IP addresses, accounting for 90.12%. In this paper, 100 communications each were extracted from source IP addresses with fewer than 10 unique destination IP addresses and from those with 10 or more. The maliciousness of the destination

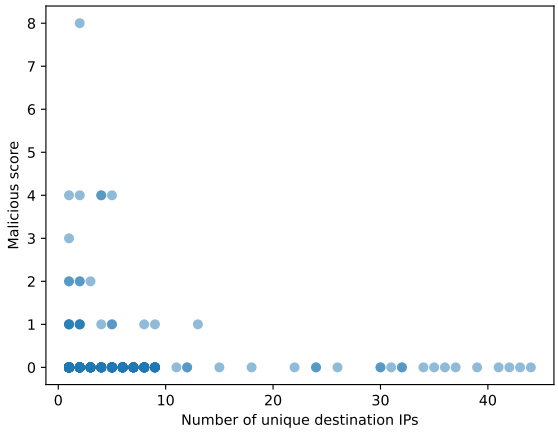


Fig. 5. Unique Number of Destinations

IP addresses was evaluated using VirusTotal. The results are shown in Figure 5.

The intensity of the color represents the number of source IP addresses corresponding to each data point. The results indicate that communications originating from source IPs with fewer than 10 unique destination IP addresses tend to have greater maliciousness. Therefore, in this paper, we set the threshold for a terminal as having many destinations to 10. Therefore, bots that communicate with 10 or more destinations were judged to be engaged in scanning activities, and their communication destinations were excluded from the candidate C2 servers.

## V. Performance Evaluation

This section describes the evaluation of the performance of the method described in Section IV.

### A. Dataset

In this experiment, flow data collected on May 2, 2024, as described in Section III, were used for training and validation. A dataset was created by generating features based on the extraction of flows with destination IP address-port pairs included in the known C2 information, with and without the exclusion of scanning targets.

Since the feature files are generated in 5-minute intervals, a dataset was created by combining 288 files for one day. The dataset was then split into training and validation sets at a 4:1 ratio.

The data specifications are shown in Table III. A comparison of the data before and after scanning target exclusion revealed a clear reduction in the number of records by approximately 80% and an approximate doubling in the percentage of C2 records.

### B. C2 Detection Performance

Next, we evaluated the detection performance of C2 records. After training an RF model with the training data, the validation data were input into the model to infer whether each record's IP address-port-protocol combination was a C2 server.

TABLE III
DATA SPECIFICATIONS

| Item | Before Exclusion | After Exclusion |
|---|---|---|
| Total Records | 841,162 | 171,854 |
| Training Records | 672,929 | 137,483 |
| Validation Records | 168,233 | 34,371 |
| C2 Records | 15,125 | 6,618 |
| Training C2 Records | 12,132 | 5,309 |
| Validation C2 Records | 2,993 | 1,309 |
| C2 Record Ratio | 0.018 | 0.038 |

TABLE IV
EVALUATION RESULTS FOR VALIDATION DATA WITH RANDOM FOREST

| Item | Before Exclusion | After Exclusion |
|---|---|---|
| Accuracy | 0.987 | 0.982 |
| Precision | 0.923 | 0.989 |
| Recall | 0.300 | 0.537 |
| F1-score | 0.453 | 0.696 |

The RF model was implemented using scikit-learn, and all the parameters were kept at their default settings.

The results of the inference were evaluated in terms of accuracy, precision, recall, and F1 score, as shown in Table IV. Prior to scanning target exclusion, the precision was high (0.923), and it was further improved after exclusion. This is due to the reduction in false positives when normal terminals were excluded from the group of detection targets. The recall improved significantly from 0.300 before exclusion to 0.537 after exclusion. This is also thought to be due to the improvement in the ratio of normal communications to C2 communications achieved by excluding normal communications from the training data, making it easier for the model to learn. Since both precision and recall improved, the F1 score, which is their harmonic mean, also increased significantly.

Next, we trained the LightGBM model and evaluated its performance using the LightGBM library, with the parameters kept at their default settings. The results are shown in Table V. These results show that excluding scanning targets increases the detection accuracy in both models. In terms of recall and F1 score, LightGBM outperforms the RF.

### C. Feature Importance

To clarify the details underlying the detection performance, we evaluated the feature importance for each ML model. That is, using the feature importance evaluation functions included in the scikit-learn and LightGBM libraries, we assessed the importance of each feature.

TABLE V
EVALUATION RESULTS FOR VALIDATION DATA WITH LIGHTGBM

| Item | Before Exclusion | After Exclusion |
|---|---|---|
| Accuracy | 0.988 | 0.987 |
| Precision | 0.736 | 0.953 |
| Recall | 0.483 | 0.693 |
| F1-score | 0.583 | 0.802 |

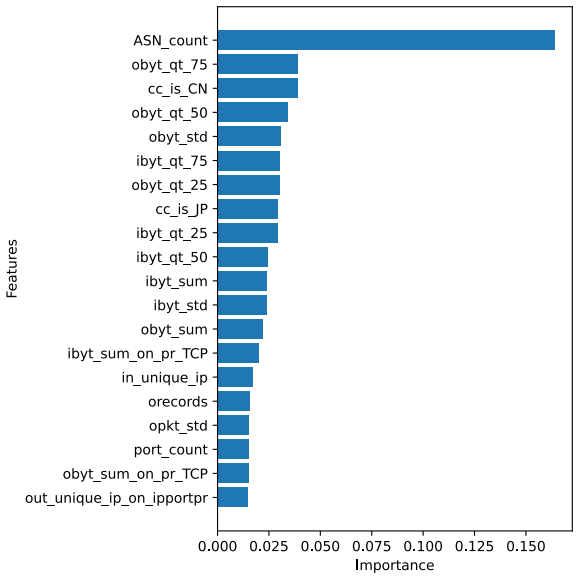


Fig. 6. Feature Importance of Random Forest

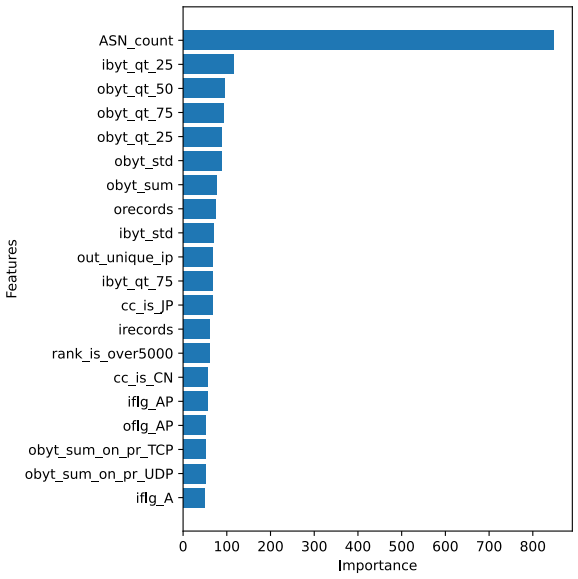


Fig. 7. Feature Importance of LightGBM

The importance scores of the top 20 features for each model after excluding scanning targets are shown in Figures 6 and 7. In both models, the feature importance of the ASN_count, which represents the number of records within the dataset that have the same ASN, was significantly greater than that of the other features.

The distribution of ASN_count, separated into C2 and legitimate traffic, is shown in Figure 8.

While the number of legitimate devices belonging to large ASs is typically very high, C2 servers often belong to smaller ASs, which are less easily detectable. Thus, ASN_count reveals the difference between a C2 server and benign terminals.

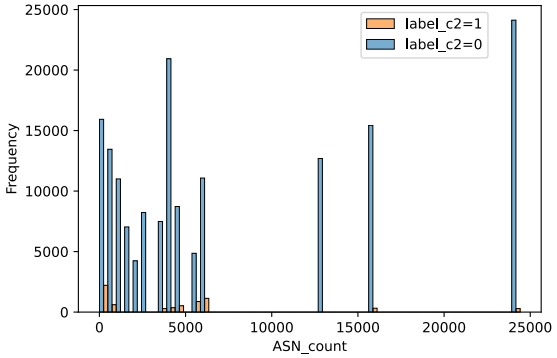


Fig. 8. Histogram of "ASN_count"

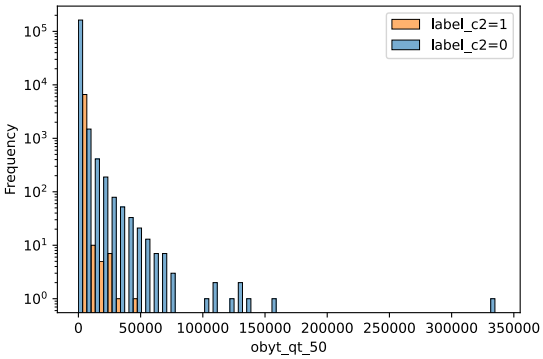


Fig. 9. Histogram of "obyt_qt_50"

Furthermore, statistical measures related to byte count also showed high importance in both models. This is because the number of outbound bytes from C2 servers is lower than that from benign servers and clients. For example, the distribution of obyt_qt_50, separated into C2 and legitimate traffic, is shown in Figure 9. These results indicate that the byte size of C2 communications is lower than that of legitimate communications.

## VI. Conclusion

This paper proposes a method for detecting C2 servers by combining heuristic analysis and machine learning using flow data obtained from the ISP network. We demonstrated that using heuristic analysis to narrow the targets for machine learning can increase detection accuracy. Our evaluation of multiple models demonstrated that features such as AS count and transmitted byte volume are effective for detection.

However, as the behavior of C2 communications is expected to change over time, it will be necessary to consider using longer-term data and establishing a mechanism for continuously updating the detection model. In the current evaluation, known C2 server IP addresses were used to label the data. However, evaluating the detection capability for unknown C2 servers in terms of accuracy and speed will present a challenge in the future.

## Acknowledgments

These research results were obtained from the commissioned research(No. JPJ012368C08101) by National Institute of Information and Communications Technology (NICT) , Japan.

## References

[1] M. M. Rahman, F. Bouhafs, and F. den Hartog, "A survey on the effectiveness of existing smart home cyber attacks detection solution: A broadband service providers' perspective," *IEEE Open Journal of the Communications Society*, 2025.

[2] D. Inoue, M. Eto, K. Yoshioka, S. Baba, K. Suzuki, J. Nakazato, K. Ohtaka, and K. Nakao, "nicter: An incident analysis system toward binding network monitoring with malware analysis," in *2008 WOMBAT Workshop on Information Security Threats Data Collection and Sharing*. IEEE, 2008, pp. 58–66.

[3] T. Al lelah, G. Theodorakopoulos, A. Javed, and E. Anthi, "Machine learning detection of cloud services abuse as c&c infrastructure," *Journal of Cybersecurity and Privacy*, vol. 3, no. 4, pp. 858–881, 2023.

[4] T. Al Lelah, G. Theodorakopoulos, A. Javed, and E. Anthi, "Detecting the abuse of cloud services for c&c infrastructure through dynamic analysis and machine learning," in *2024 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE, 2024, pp. 1–7.

[5] V. Quezada, F. Astudillo-Salinas, L. Tello-Oquendo, and P. Bernal, "Real-time bot infection detection system using dns fingerprinting and machine-learning," *Computer Networks*, vol. 228, p. 109725, 2023.

[6] H. Biros and M. Kantor, "Enhancing dga detection with machine learning algorithms," *Journal of Telecommunications and Information Technology*, 2025.

[7] T. Koga, D. Nobayashi, and T. Ikenaga, "Accuracy improvement method for malicious domain detection using machine learning," in *2024 IEEE 21st Consumer Communications & Networking Conference (CCNC)*. IEEE, 2024, pp. 1108–1109.

[8] Y. Huang, J. Qin, Z. Cui, N. Li, B. Jiang, and Z. Lu, "Hbgraph: a host behavior graph model for c&c traffic detection," in *2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. IEEE, 2024, pp. 2788–2793.

[9] K. Vidhun and J. M. Kannimoola, "Lightweight real-time c&c detection using deep learning for zombie dns queries," in *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. IEEE, 2024, pp. 1–5.

[10] F. M. Ramos and X. Wang, "Detecting stealthy cobalt strike c&c activities via multi-flow based machine learning," in *2023 International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2023, pp. 2200–2206.

[11] S. Baruah, D. J. Borah, and V. Deka, "Detection of peer-to-peer botnet using machine learning techniques and ensemble learning algorithm," *International Journal of Information Security and Privacy (IJISP)*, vol. 17, no. 1, pp. 1–16, 2023.

[12] M. A. Kazi, "Detecting malware c&c communication traffic using artificial intelligence techniques," *Journal of Cybersecurity and Privacy*, vol. 5, no. 1, p. 4, 2025.

[13] Y. Merkli, R. Meier, M. Strohmeier, and V. Lenders, "Defeating and improving network flow classifiers through adversarial machine learning," in *2024 16th International Conference on Cyber Conflict: Over the Horizon (CyCon)*. IEEE, 2024, pp. 103–121.

[14] Triage. Recorded Future. [Online]. Available: https://tria.ge/

[15] Feodotracker. Abuse. [Online]. Available: https://feodotracker.abuse.ch/

[16] Threatfox. Abuse. [Online]. Available: https://threatfox.abuse.ch/

[17] Geolite2 asn database. MaxMind. [Online]. Available: https://dev.maxmind.com/geoip/docs/databases/asn

[18] Nicterweb. National Institute of Information and Communications Technology. [Online]. Available: https://www.nicter.jp/en/

[19] Virustotal. Google. [Online]. Available: https://www.virustotal.com/

[20] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[21] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, 2017.