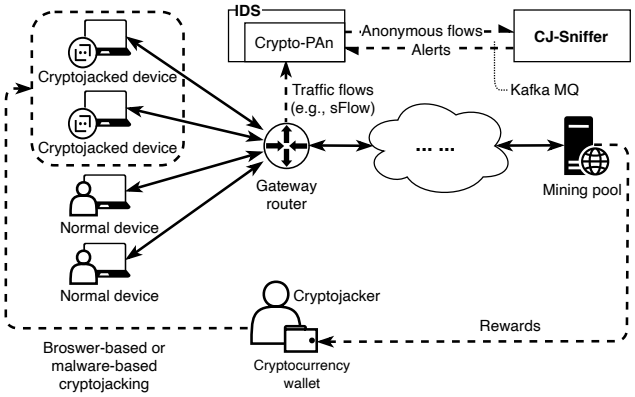
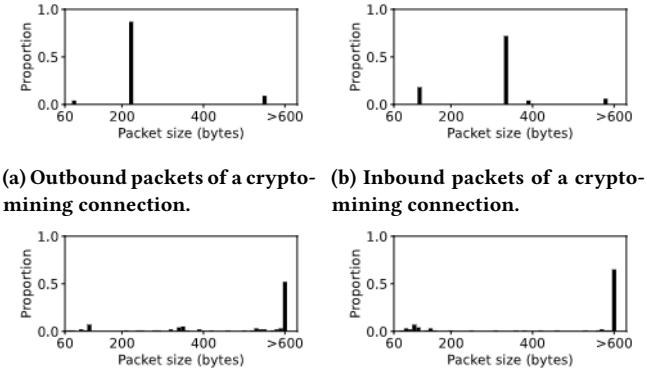


Figure 8: One operational model of CJ-Sniffer, where it is deployed apart from the IDS.



(a) Outbound packets of a cryptomining connection.

(b) Inbound packets of a cryptomining connection.

(c) Inbound packets of a Steam downloading connection. Most packets are larger than 600 bytes.

(d) Inbound packets of a video streaming connection. Most packets are larger than 600 bytes.

Figure 9: Packet size distribution of different types of traffic.

service provider or any third parties cannot fetch necessary information to trace back to individuals in the network. Figure 8 illustrates the operational model of CJ-Sniffer in this type of deployment,

As stated in Section 1, CJ-Sniffer detects cryptojacking activities in three phases. The first detection phase can quickly filter out irrelevant traffic flows, leaving only suspicious ones for future analysis. The second phase inputs suspicious traffic and outputs confirmed cryptomining traffic. At last, the third detection phase utilizes an LSTM model to distinguish cryptojacking traffic from user-initiated cryptomining traffic.

4.1 Preprocessing

CJ-Sniffer requires the timestamps and six fields in the IP packet headers for detection, which are the source and destination IP addresses, the source and destination port numbers, the protocol type, and the packet size. CJ-Sniffer is therefore content-agnostic, as it does not require any payload information.

To obtain this content-agnostic data, we recommend installing sFlow [39] in the router or switch to stream traffic flows to CJ-Sniffer in real time. Other network traffic capturing engines like Netmap [41] and PF-RING [7] are also compatible with the proposed approaches. Once CJ-Sniffer fetches the traffic flows, it extracts the aforementioned data fields and stores them in a table for future analysis. In the table, each entry represents a received packet. Meanwhile, CJ-Sniffer discards all other information from the traffic flows.

4.2 Phase one: rapid filtration

In phase one, CJ-Sniffer rapidly filters out irrelevant network traffic and picks out only suspicious traffic for future analysis. This step can significantly reduce the size of the traffic data for inference, increasing the throughput of CJ-Sniffer.

To conduct rapid filtration, CJ-Sniffer first eliminates packets without payload (e.g., TCP SYN, ACK, and FIN packets), packets of irrelevant protocols (e.g., ICMP), and internal packets. CJ-Sniffer then groups remaining packets by connections, which are defined as consecutive packets that are sent between the same IP addresses and port numbers. Then, CJ-Sniffer inspects the packet sizes in each connection to judge whether it could be a suspicious cryptomining connection.

We define a sliding time window to monitor each connection and make a judgment. During each time window t, CJ-Sniffer collects a list of packets P ($P = \{p_1, p_2, p_3, ..., p_n\}$) from the ongoing connection. It then represents inbound and outbound packets with their size values and stores them into two lists (l_{in} for inbound packets and l_{out} outbound packets) respectively. Once this time window is about to end, CJ-sniffer will check the value distributions of l_{in} and l_{out} to determine whether the connection is suspicious. According to the traffic measurement study in Section 3.1, CJ-Sniffer utilizes three sets of rules to determine suspicious connections:

- The majority of the packets' sizes should lie within the cryptomining packet size range;
- The majority of the outbound packets' sizes should be uniform (illustrated in Figure 9a);
- The majority of the inbound packets' sizes should have values drawn from the same narrow interval (illustrated in Figure 9b).

Once packets within a connection follow these three rules, CJ-Sniffer will label this connection as suspicious and pass it onto the next phase for deeper analysis. Note, labeling a connection as suspicious does not mean this connection is confirmed to be related with cryptomining. For example, some connections generated by NTP or DNS can have similar distribution of packet sizes. Hence, CJ-Sniffer only filters out obviously irrelevant traffic in this phase to accelerate the detection process.

4.3 Phase two: detection of cryptomining

In phase two, CJ-Sniffer uses filtered network traffic from phase one as input and outputs detected cryptomining traffic. According to the cryptomining traffic study in Section 3.1, to accurately identify cryptomining traffic, CJ-Sniffer inspects the packet intervals of suspicious connections to determine whether they are generated

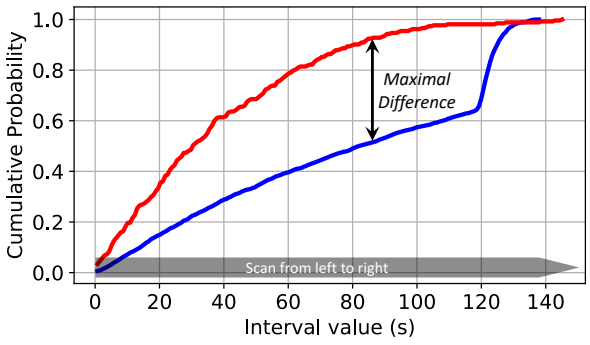


Figure 10: Illustration of the two-sample Kolmogorov–Smirnov statistic. The blue line corresponds to the empirical distribution function of labeled cryptomining traffic. The red line corresponds to the empirical distribution function of downloading traffic. The black arrow is the two-sample KS statistic.

by the cryptominer or the mining pool. Specifically, CJ-Sniffer compares both the inbound and outbound packet interval distributions to distributions from collected cryptomining traffic. As long as one of the inbound or outbound packet intervals follow the same distribution with cryptomining traffic data, CJ-Sniffer will label the connection as a cryptomining connection.

The distribution compliance test is conducted with the Two-Sample Kolmogorov–Smirnov (KS) test [34], which is a nonparametric testing approach to determine whether two data samples come from the same distribution. Compared with other approaches to testing the distribution compliance, the KS test has no restrictions on the size of data sample, which means little cryptomining traffic data can help achieve decent accuracy. Moreover, the KS test is distribution-free. Users can easily update the contrast sample to cover the latest cryptomining traffic regardless of the sample's distribution.

The Two-Sample KS test works as follows. Suppose that the inbound or outbound packet interval sample from P has size m with an observed cumulative distribution function of $F(x)$. Furthermore, suppose that the labeled cryptomining sample Q has size n with an observed cumulative distribution function of $G(x)$. CJ-Sniffer defines the null hypothesis (H_0) as: both samples come from a population with the same distribution. It also defines the Two-Sample KS statistic $D_{m,n}$ with Equation 1 (illustrated in Figure 10).

$$D_{m,n} = \max_x |F(x) - G(x)|. \tag{1}$$

After calculating $D_{m,n}$, CJ-Sniffer rejects the null hypothesis at significance level α if $D_{m,n} > D_{m,n,\alpha}$, where $D_{m,n,\alpha}$ is the critical value and can be calculated with Equation 2.

$$\begin{aligned} D_{m,n,\alpha} &= c(\alpha)\sqrt{\frac{n+m}{n \cdot m}} \\ &= \sqrt{-\ln(\frac{\alpha}{2}) \cdot \frac{1+\frac{m}{n}}{2m}}. \end{aligned} \tag{2}$$

Conversely, if $D_{m,n} \leq D_{m,n,\alpha}$, CJ-Sniffer will accept the null hypothesis H_0 and label the incoming traffic as cryptomining.

The significance level α in the Two-Sample KS test is the probability of rejecting the null hypothesis when it is true. Users of CJ-Sniffer can adjust the value of α to reach different detection sensitivities. A large significance level α can lead to a small critical value $D_{m,n,\alpha}$, which will raise the standard of the distribution compliance test. In our implementation, we set α as 0.10, which is a relatively large value but can increase the usability of CJ-Sniffer by reducing the false positive rate.

Algorithm 1 Cryptomining traffic detection using KS test.

Input: P, m, Q, $G(x)$, n, k, α ▷ P is the packet list with m packets, Q is the labeled cryptomining packet list with n packets, $G(x)$ is the cumulative distribution function of Q, k is the granularity for calculating the KS statistic, α is the significance level
Output: 1 for cryptomining traffic, 0 for other traffic
$l_P = inboundInterval(P)$ ▷ extract the inbound packet intervals and store them in a list
$l_Q = inboundInterval(Q)$
$r = max(l_Q) - min(l_Q)$ ▷ calculate the range of $G(x)$
initialize list l_d ▷ to store the differences of two cumulative distribution functions (CDFs)
for i in $range(k)$ **do**
    $x \longleftarrow \frac{i \cdot r}{k} + min(l_Q)$
    $l \longleftarrow \{j | j \in l_P \text{ and } j \leq x\}$
    $f \longleftarrow \frac{l.size()}{l_P.size()}$ ▷ calculate the CDF value at x for P
    append $|f - G(x)|$ to l_d
    if $f == 1$ **then**
        break
    end if
end for
$D_{m,n} \longleftarrow max(l_d)$
if $D_{m,n} \leq \sqrt{-\ln(\frac{\alpha}{2}) \cdot \frac{1+\frac{m}{n}}{2m}}$ **then**
    return 1 ▷ accept the hypothesis H_0
else
    return 0
end if

Algorithm 1 demonstrates the detailed procedure that CJ-Sniffer utilizes to determine cryptomining traffic. CJ-Sniffer only tests the distribution compliance of inbound packet intervals, as inbound packet intervals are more robust compared with outbound packet intervals (discussed in Section 3). Moreover, to reduce the process time, the cumulative distribution function of labeled cryptomining traffic is calculated beforehand. Therefore, when receiving new suspicious traffic, CJ-Sniffer only needs to build one cumulative distribution function.

Once CJ-Sniffer completes the analysis in phase two, network operators can choose the next step according to their needs. For instance, if some companies and institutions prohibit any cryptomining inside their networks, then they can stop at this phase and block any connection that is labeled as cryptomining. Meanwhile, some network operators allow user-initiated cryptomining activities. Nonetheless, they still want to distinguish cryptojacking traffic

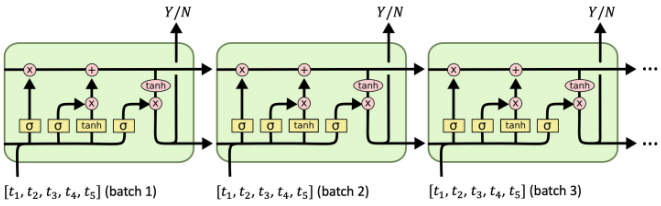


Figure 11: Structure of the LSTM model that CJ-Sniffer utilizes.

to safeguard users' computing resources. In this case, CJ-Sniffer can enter phase three to dig further into the labeled cryptomining traffic.

4.4 Phase three: detection of cryptojacking

In the third phase, CJ-Sniffer uses detected cryptomining traffic as input and outputs identified cryptojacking traffic. According to the detection results of CJ-Sniffer, network operators can conduct access control only on cryptojacking connections while still leaving user-initiated cryptomining connections alive.

Enlightened from the measurement results in Section 3.2, CJ-Sniffer distinguishes cryptojacking traffic from user-initiated cryptomining traffic by inspecting the long-term robustness of the result submission messages. For various reasons, the hash rate of cryptojacking activities is relatively unstable compared with user-initiated cryptomining. This hash rate instability further affects the frequency of result submission messages (msg_r). The key to identifying cryptomining connections is to recognize such frequency instability.

To achieve the goal, CJ-Sniffer utilizes a LSTM machine learning model to learn the cryptojacking traffic patterns, with cryptojacking traffic as positive samples and user-initiated cryptomining traffic as negative samples. Then, CJ-Sniffer applies the trained LSTM model to identify cryptojacking traffic. LSTM is a type of artificial recurrent neural network (RNN) architecture used in the field of deep learning [23]. Compared with other candidate approaches, LSTM is particularly suitable to detect cryptomining traffic for the following reasons: (1) as an RNN variant, LSTM is good at processing time series or sequential data, such as cryptomining traffic; (2) LSTM introduces both a short-term and a long-term memory component, allowing it to uncover hidden frequency changes out of traffic data from a long-term perspective; (3) LSTM is a learning-based approach, which means it can automatically and dynamically learn the detection thresholds from our collect dataset without any human interventions. Nonetheless, the detection component here is modular. Users may use other machine learning algorithms or statistical approaches to fit their environments once the inputs are the same.

As for the input data, CJ-Sniffer extracts the variation vector v from outbound traffic to profile the changes in result submission message frequency. CJ-Sniffer first picks out only the outbound packets from the traffic and divides them into batches. Every batch consists of six consecutive packets with five consecutive packet intervals. Then, CJ-Sniffer generates a variation vector v to represent each batch of data, where $v = [t_1, t_2, t_3, t_4, t_5]$ and t_n denotes the interval of two consecutive packets. CJ-Sniffer will input the variation vectors into the LSTM model and conduct cryptojacking detection batch by batch.

Table 2: Information of the collected cryptomining dataset.

Cryptocurrencies	Size	Length	Mining Chips
XMR, ETH	250 MB	~ 750 hours	Intel Core i5-5257U, Intel Core i7-6820HQ, Apple M1, AMD Ryzen 5 1400 quad-core, NVIDIA GeForce GTX 1080, Intel Core i7-8700K, Intel Xeon CPU E5-2430, AMD Radeon RX 570, AMD Ryzen 5 1400 quad-core + AMD Radeon RX 570.

Figure 11 illustrates the structure of the LSTM model that CJ-Sniffer utilizes. As the input data is not complicated, CJ-Sniffer employees a Vanilla LSTM model, which has a single hidden layer of LSTM units, and an output layer used to make the decision. We set the number of neurons in the hidden layer to 20 according to a rule of thumb that was introduced in [17]. Equation 3 demonstrates the rule, where N_i denotes the number of input neurons, N_o denotes the number of output neurons, N_s denotes the number of samples in the training set, α is an arbitrary scaling factor (usually ranges from 2 to 10), and N_h denotes the maximum number of neurons in the hidden layer.

$$N_h = \frac{N_s}{\alpha \cdot (N_i + N_o)}. \tag{3}$$

Moreover, the LSTM uses binary cross entropy as the loss function and Sigmoid as the activation function for the output neuron, as this combination is the most commonly used for binary classification problems. CJ-Sniffer keeps inputting variation vectors as time-series data to the LSTM, until the incoming traffic is identified as cryptojacking or all the batches have been processed.

The LSTM model requires training before it can be used. Since there are no existing cryptojacking traffic datasets available in public repositories, we captured both user-initiated cryptomining traffic and cryptojacking traffic to train the LSTM model. We also release part of the packet-level dataset with this paper.

5 EVALUATION

We evaluated CJ-Sniffer in campus network environments with real-world network traffic. In this section, we first describe how we collect labeled cryptomining traffic and real-world contrast traffic (Section 5.1). Then, we show our evaluation results of CJ-Sniffer regarding the efficacy of cryptomining traffic detection (Section 5.2), the ability of detecting other cryptocurrencies' mining traffic (Section 5.3), the efficacy of cryptojacking traffic detection (Section 5.4), and comparisons with other network-based approaches (Section 5.5). In the end, we evaluate the operation efficiency and the real-world deployability of CJ-Sniffer (Section 5.6.2).

5.1 Data collection

To support the measurement study of cryptomining traffic, train the statistical model and the machine learning model, and promote related research, we collected a labeled cryptomining traffic dataset from multiple calculation platforms, including both servers and personal computers, CPUs and GPUs. The cryptomining traffic dataset contains both user-initiated cryptomining traffic and cryptojacking traffic.