Examining the Robustness of Machine Learning-based Phishing Website Detection: Action-Masked Reinforcement Learning for Automated Red Teaming

Yang Gao
Kelley School of Business
Indiana University Bloomington
Bloomington, IN, United States
gaoyang@iu.edu

Benjamin Ampel
Robinson College of Business
Georgia State University
Atlanta, GA, United States
bampel@gsu.edu

Sagar Samtani
Kelley School of Business
Indiana University Bloomington
Bloomington, IN, United States
ssamtani@iu.edu

***Abstract*—As machine learning (ML)-based detectors become increasingly prevalent in identifying phishing websites, attackers are also exploiting their vulnerabilities through evasion techniques. By subtly manipulating phishing websites, attackers can evade detection. The threats posed by evasion attacks necessitate proactive robustness testing of these detectors prior to deployment. Traditional red teaming efforts, where security experts manually emulate attacker behaviors, are labor-intensive and limited in scalability. To address this challenge, we propose an automated red teaming framework leveraging action-masked reinforcement learning (RL) to realistically emulate evasion attacks and evaluate the robustness of ML-based phishing website detectors. Our RL agent is equipped with HTML manipulation techniques commonly used by human attackers. Additionally, action masking ensures the RL agent selects only evasion actions that are feasible for a given website and prevents compromising website rendering. We evaluate our approach by testing the robustness of three ML-based detectors: Logistic Regression, Random Forest, and Convolutional Neural Networks. Experimental results demonstrate that our approach achieves high evasion capabilities and efficiency in converting detectable phishing websites into well-rendered evasion ones, thus effectively testing the robustness of the detectors.**

***Index Terms*—Red Teaming, Phishing, Reinforcement Learning, Adversarial Machine Learning**

I. Introduction

Phishing is a pervasive form of cybercrime, and phishing website is one of the most common modalities of phishing attacks. In such attacks, attackers design fraudulent websites that mimic legitimate ones to deceive users and illegally acquire sensitive information. According to the Anti-Phishing Working Group (APWG), millions of unique phishing websites were reported in 2024, with numbers ranging from 260,000 to 370,000 per month [1]. To counteract these threats, defenders turn to machine learning (ML) to develop defense mechanisms [2]–[5]. However, ML-based detectors are vulnerable to evasion attacks. Attackers can manipulate phishing websites using evasion techniques to trick less robust detectors into misclassifying phishing sites as legitimate [6], [7]. Therefore, before deploying ML detectors, it is essential to proactively examine their robustness against such evasion attacks.

To conduct such examinations, security experts commonly employ red teaming, where ethical attackers emulate potential attack behaviors as realistically as possible without causing actual harm [8]. Traditional red teaming practices are primarily manual, requiring significant human effort and limiting the scale of potential attack behaviors explored. Therefore, in this study, we propose an automated red teaming approach using reinforcement learning (RL) to reduce human effort and conduct large-scale emulation of phishing evasion. This approach incorporates human attackers' knowledge of manipulating HTML file code into the RL action space. The RL learns to make optimal decisions on selecting evasion techniques to manipulate the code of phishing websites, transforming detectable versions into well-rendered evasive ones. Additionally, the variety of web development methods results in diverse code structures among phishing websites, leading to different feasibility of evasion techniques for each unique site. To address this, we implement action masking to "mask out" invalid evasion techniques based on the observations of HTML, preventing the RL agent from recommending techniques that cannot be executed or that might disrupt website rendering.

With our action-masked RL approach, we can effectively emulate human attackers' behaviors and assess the robustness of ML detectors by how easily they can be evaded. The remainder of this paper is organized as follows: First, we review related work on phishing website detection, evasion attacks, and the application of reinforcement learning in adversary emulation. Next, we present our proposed automated red teaming approach. We then discuss our experimental results, followed by conclusions.

II. Related Work

Existing literature has demonstrated the effectiveness of machine learning in phishing website detection. Many studies

DOI 10.1109/SPW67851.2025.00041

leverage security knowledge and representation learning to extract features from URLs and HTML source code, which are then fed into either traditional machine learning models, such as Logistic Regression (LR), or deep learning classifiers, such as Convolutional Neural Network (CNN), to classify websites as phishing or benign [9].

On the offensive side, researchers have explored methods to evade ML-based detectors. Several studies have successfully generated evasive URLs using Wasserstein GAN with gradient penalty (WGAN-GP) [10]. However, realistically emulating evasion attacks that target detectors utilizing HTML features presents a greater challenge. It requires manipulating HTML source code while preserving the website's rendering. Most existing studies focus on adding theoretical noise to feature vectors of detectors to create evasive feature vectors, without directly manipulating the HTML code to produce fully functional evasive websites [11]. As a result, these approaches are less representative of real-world threats posed by human attackers, who typically lack access to the internal components of detectors and rely primarily on modifying website code.

Apruzzese et al. [6] and Montaruli et al. [7] are pioneers in proposing evasion attacks that directly employ evasion techniques used by real attackers to manipulate HTML source code, resulting in well-rendered evasive phishing websites. However, Apruzzese et al. [6] utilized only two evasion techniques for HTML manipulation and did not model the decision-making process for optimally selecting these techniques. Montaruli et al. [7] introduced 16 evasion techniques, significantly expanding the evasion action space. However, their decision-making process relies on continuous decision scores from detectors. This information is typically inaccessible to most real attackers who can only know whether their websites are detected or not. Therefore, there is a need for a more realistic evasion emulation method that incorporates a broad range of evasion techniques and mimics human decision-making to optimally select and execute evasion techniques. Such a method should efficiently generate well-rendered evasive phishing websites without requiring internal knowledge of ML detectors.

Reinforcement learning (RL) can model an autonomous agent to take sequential actions optimally, with little or no prior knowledge of the environment, making it particularly adaptable and effective in adversarial contexts [12]. Although RL has not yet been adapted in the context of phishing evasion, model-free RL approaches have proven successful in adversary emulation within other cybersecurity domains, such as evading malware detectors [13] and conducting cyber-physical systems (CPS) attacks [14]. Additionally, Huang et al. [15] theoretically demonstrated the efficiency of action masking to "mask out" invalid actions.

III. Methodology

Building on the literature, we propose an action-masked RL approach in this study for the automated emulation of evading HTML-based ML phishing website detectors (Figure 1). Our study framework comprises three primary components: (A) Data Collection, (B) Action-Masked RL Agent for Adversarial Phishing Website Generation, and (C) Evaluation. Each component is detailed in the subsequent subsections.

A. Data Collection

We developed a crawler to harvest phishing and benign websites. For phishing data, the crawler ran daily, collecting newly reported phishing URLs verified by PhishTank, a community-driven phishing verification platform. GNU Wget was used to visit the URLs, retrieving and downloading all necessary files to reproduce the rendering of these websites. For benign websites, we collected URLs and associated rendering files from the top 10,000 websites on Tranco [16], a research-oriented website ranking list hardened against adversarial manipulation for security experts. This ensured a clean, reliable benign dataset. The collection was conducted in a sandboxed environment on Jetstream2 [17].

With the collected data, we established two datasets. Dataset 1 is used to train and test the detector. It includes 10,000 phishing websites from PhishTank collected between May and July 2024, and 10,000 benign websites from Tranco. The dataset is split into 80% for training and 20% for testing. Dataset 2 is used for evasion experiments. It comprises 10,000 phishing websites collected from PhishTank between August and October 2024. Of these, 5,000 are used to train RL agents to accumulate experience in manipulating HTML source code to evade a certain detector. The remaining 5,000 are used to test the evasion performance of the trained RL agents and the robustness of target detectors. Importantly, our RL agents, acting as attackers, have no access to Dataset 1, emulating realistic attack scenarios where adversaries lack knowledge of the detectors' training data.

B. Action-masked RL Approach

Our RL approach consists of four components (Figure 1): (a) the environment, comprising phishing samples, the rendering checker, and the target detector, which provides feedback on the success or failure of evasion attempts; (b) the state space, capturing the current status of the manipulated HTML source code; (c) the action space, offering the RL agent a repertoire of evasion techniques; and (d) the RL model, guiding the decision-making and learning of the RL agent.

a) Environments: To emulate interactions between real attackers and target detectors, we built an environment that allows RL agents to manipulate the HTML of detectable phishing websites. The environment integrates three components: detectable phishing websites, a target detector, and a rendering checker. In each interaction step, the RL agent selects an evasion technique to modify the HTML source code of a phishing website. After each manipulation, the detector classifies the website as phishing or benign. The rendering checker captures and compares screenshots of the website before and after manipulation at the pixel level to ensure visual consistency. If rendering breaks, the manipulation is withdrawn, and the evasion technique causing the break is "masked out" to prevent future selection for that website.

TABLE I
PERFORMANCE OF TARGET DETECTORS ON TEST SET OF DATASET 1

Detector	Accuracy	Precision	Recall	F1	AUC	True Positive	False Positive	True Negative	False Negative
RL	0.7916	0.7699	0.8317	0.7996	0.7916	1650	493	1492	334
RF	0.9365	0.9279	0.9466	0.9371	0.9365	1878	146	1839	106
CNN	0.9098	0.8757	0.9551	0.9137	0.9098	1895	269	1716	89

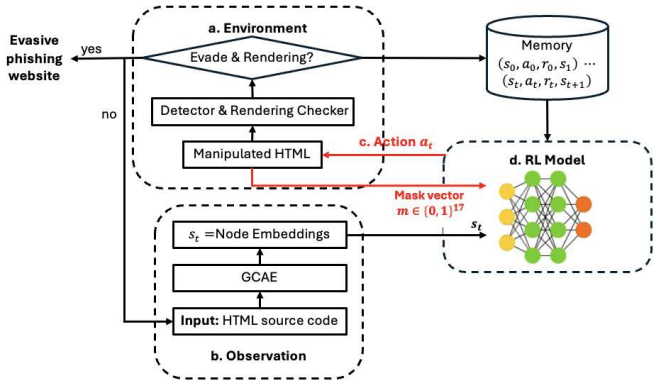

Fig. 1. Action-masked RL Approach

The environment provides a reward of 10 to the RL agent if the detector is misled into classifying the phishing website as benign while maintaining proper rendering. Otherwise, the reward is 0. This reward design emulates realistic scenarios where attackers only receive binary classification results from detectors and must verify whether their websites render properly after deployment. Considering the cost of evasion attacks, attackers are unlikely to manipulate websites indefinitely. Therefore, we set interaction budget as 10, limiting the number of interactions the RL agent can have with the detector to manipulate one phishing website.

b) State Space: At each interaction step, the RL agent relies on an observation s of the HTML structure to determine the evasion technique. Since the HTML source code is organized by elements in a tree structure, we train a Graph Convolutional Autoencoder (GCAE) to represent each element in the HTML as a node embedding. This representation captures both the intrinsic features of the elements (such as element name, attribute names, attribute values, and textual content) and their relational positioning within the Document Object Model (DOM) tree. Consequently, the HTML source code of a website is represented as a matrix, where each row corresponds to the embedding of an individual HTML element. This matrix serves as the state input for the RL agent, providing comprehensive contextual information to guide its decision-making process.

c) Maskable Action Space: The action space equips our RL agent with a set of evasion techniques commonly used by real attackers. It comprises three categories: (1) Injection techniques, which insert invisible benign content into URLs and HTML source code, disrupting the semantics and structure of the original phishing content; (2) Obfuscation techniques, which encode or encrypt phishing content to hinder detector analysis; and (3) Techniques for updating sensitive information, replacing attribute values that may trigger detection with alternative execution methods. These attributes often make malicious code invisible to users, such as "visibility:hidden", "display:none" and "javascript:void(0)". Compared to Montaruli et al. [7], our evasion technique set incorporates more injection techniques, enabling the RL agent to inject both text information and JavaScript code. In total, the action space includes 17 evasion techniques: eight injection techniques, one obfuscation technique, and eight techniques for replacing sensitive attribute values (see Appendix).

At each interaction step, the RL agent recommends an evasion technique a from the action space. However, due to the varied web development practices in phishing websites, not all evasion techniques are applicable to every HTML code file. Some techniques may lack an identifiable execution location, while others may break the website's rendering, leading to action withdrawal in the previous step. Therefore, before each interaction, by scanning the current HTML structure and the rendering check result from the prior step, a mask vector $m \in \{0,1\}^{17}$ is given, which masks out inapplicable actions.

$$m_i = \begin{cases} 1, & \text{if } a_i \text{ is applicable,} \\ 0, & \text{if } a_i \text{ is inapplicable,} \end{cases}$$

where i denotes the i^{th} evasion technique in the action space.

d) Action-masked RL model: To integrate the action masking into RL models, we need to zero out the probabilities of invalid actions when the RL models predict the cumulative rewards to recommend actions. Therefore, for value-based components such as the Q network of Deep Q-Network (DQN) and the Critic network of Soft Actor-Critic (SAC),

$$Q_{\text{masked}}(s, a_i) = \begin{cases} Q(s, a_i), & \text{if } a_i \text{ is applicable,} \\ -\infty, & \text{if } a_i \text{ is inapplicable.} \end{cases}$$

In addition, for the policy-based components such as the Actor network of SAC, we set the logits of invalid actions to a very large negative value as well,

$$Z_{\text{masked}}(a_i|s) = \begin{cases} Z(a_i|s), & \text{if } a_i \text{ is applicable,} \\ -\infty, & \text{if } a_i \text{ is inapplicable.} \end{cases}$$

This ensures the softmax of the Actor network

$$\pi_{\text{masked}}(a|s) = \frac{\exp(Z_{\text{masked}}(a|s))}{\sum_{a'} \exp(Z_{\text{masked}}(a'|s))}$$

assigns zero for the predicted probabilities of invalid actions.

TABLE II
PERFORMANCE OF TARGET DETECTORS AFTER ATTACKS ON DATASET 2

Target Detector	Evasion Agent	Interaction Budget	Evasive phishing websites	After Attack			Evasion Efficiency
				Recall	TP	FN	
LR	Random	10	734	0.7688	3844	1156	0.018
	Maskable DQN	10	**1793**	**0.557**	**2785**	**2215**	**0.057**
	Maskable SAC	10	1671	0.5814	2907	2093	0.055
RF	Random	10	3224	0.3166	1583	3417	0.131
	Maskable DQN	10	238	0.9138	4569	431	0.005
	Maskable SAC	10	**4288**	**0.1038**	**519**	**4481**	**0.524**
CNN	Random	10	3732	0.2326	1163	3837	0.151
	Maskable DQN	10	**4406**	**0.0978**	**489**	**4511**	0.594
	Maskable SAC	10	4404	0.0982	491	4509	**0.686**

C. Evaluation

To evaluate the performance of our proposed action-masked RL approach, using Dataset 1, we replicated three HTML-based ML detectors from Apruzzese et al. [6] as target detectors. These detectors leveraged Logistic Regression (LR), Random Forest (RF), and Convolutional Neural Network (CNN) as classifiers, respectively. Their performance on the test set of Dataset 1 is summarized in Table I.

In our evasion experiments, we compare the evasion performance of Maskable DQN and Maskable SAC agents trained on the training set of Dataset 2, as well as random agents that do not optimally select evasion techniques. Our evaluation metrics assess two dimensions of performance: evasion capability and evasion efficiency. For evasion capability, as evasion attacks primarily affect the detection of phishing websites, we compare the True Positive (TP), False Negative (FN), and recall of the detectors before and after the attacks on the test set of Dataset 2. These metrics directly reflect the number of detectable phishing websites successfully transformed into evasive ones. We measure evasion efficiency as:

$$\text{Evasion efficiency} = \frac{\text{\# of evasive phishing websites}}{\text{Total number of Interactions}} \in [0, 1].$$

Higher values, approaching 1, indicate that the agent can generate evasive phishing websites with fewer interactions, demonstrating greater evasion efficiency.

IV. EXPERIMENT RESULTS

Before conducting evasion attacks, we evaluated the original performance of our target detectors on 5,000 phishing websites from the test set of Dataset 2. Out of these 5,000 websites, LR detected 4,578 phishing websites, achieving a recall of 0.9156, with 422 false negatives. RF identified 4,807 phishing websites, achieving a recall of 0.9614 and 193 false negatives. CNN achieved the highest performance, correctly detecting 4,895 phishing websites with a recall of 0.979 and only 105 false negatives. All three detectors showed high performance in detecting phishing websites before the evasion attacks.

Then we trained Maskable DQN and Maskable SAC agents against each detector using the training set of Dataset 2 until convergence. Subsequently, these trained RL agents, along with a random agent, were used to deploy evasion attacks on the test set of Dataset 2. The results of these evasion experiments are presented in Table 2. We have three key findings from the results. First, all agents, including the random agent, were able to generate well-rendered evasive phishing websites, demonstrating the effectiveness of our evasion techniques in the action space. Second, most RL agents outperformed the random agent in evasion performance. The RL agents transformed more detectable phishing websites into evasive ones, significantly reducing the detectors' recall. Additionally, RL agents exhibited higher evasion efficiency, requiring fewer interactions with detectors to achieve successful evasion. Third, comparing performance degradation across detectors revealed that although the original detection performance of LR was lower than that of RF and CNN, LR demonstrated greater robustness against evasion attacks. The best-performing evasion agent reduced LR's recall from 0.9156 to 0.557, whereas RF's recall dropped from 0.9614 to 0.1038, and CNN's recall fell from 0.979 to 0.0978.

V. CONCLUSIONS

Our study makes several contributions to the field of adversarial machine learning and phishing website detection. First, we propose an automated red-teaming approach using action-masked reinforcement learning (RL) to emulate realistic evasion attacks against ML-based phishing website detectors. Our approach effectively integrates human attackers' knowledge of HTML manipulation into the RL action space, allowing the agent to optimally select and apply evasion techniques without compromising website rendering. Second, our study demonstrates the necessity of proactively evaluating the robustness of ML-based phishing website detectors before deployment, as higher detection performance in non-adversarial settings does not guarantee higher detector reliability. In the future, we aim to expand our testbed to include a broader range of detectors that utilize different training datasets, feature extraction methods, and machine learning classifiers.

APPENDIX
EVASION TECHNIQUES

17 evasion techniques (Table III) are incorporated into the action space of the RL agent. For each technique, we predefined both the manipulation method and the manipulation location. With this set of evasion techniques, we can manipulate elements, attributes, text information, JavaScript code, and the DOM structure of the HTML.

TABLE III
EVASION TECHNIQUES

Category	Evasion Techniques	Manipulation method	Manipulation Location
Injection (Links, Text, JavaScript)	InjectIntAnchorElem (IIAE)	Injecting 10 internal links by <a> elements in body/footer	Appending to the first element in the body or footer as a sibling
	InjectIntLinkElem (IILE)	Injecting 10 internal links by <link> elements head/body/footer	Appending to the first element in the head, body or footer as a sibling
	InjectExtAnchorElem (IEAE)	Injecting 10 external links by <a> elements in head/body/footer	Appending to the first element in the body or footer as a sibling
	InjectFakeFavicon (IFF)	Injecting a fake favicon by <link> element	Appending to <head>
	InjectTextElem (ITE)	Injecting a <p> element	Appending to the first element in the body or footer as a sibling
	InjectSpaceInText (ISIT)	Injecting 10 zero-width no-break space	Except for <script>, <style>, <noscript>, injecting within the first element that has text information
	InjectFakeCopyright (IFC)	Injecting a fake copyright by <p> element.	Appending to <body>
	InjectJSElem (IJSE)	Injecting a <script> element	Appending to the first element in the head, body or footer as a sibling
Obfuscation	ObfuscateJS(OJS)	Encoding functions within a <script> element	Encoding the first <script> element
Update Sensitive Information (AttrsValue, TitleText)	UpdateExtLink (UEL)	Replacing external links in body's elements by internal links, and adding a <script> to put external links back by JavaScript at execution time	The first <img>, <link>, <a>, <sound>, <video>, or <form> element that has external links
	UpdateForm (UF)	Replacing action attribute value of forms	The first <form> element that has sensitive "hidden" values
	UpdateIntAnchors (UIA)	Replacing "href" attribute value of anchors	The first <a> element that has sensitive "hidden" values
	UpdateHiddenDivs (UHD)	Replacing style's attribute value of elements.	The first element that has sensitive "hidden" values
	UpdateHiddenButtons (UHB)	Removing disabled attribute of <button> elements, and adding a <script> to put disable attribute back by JavaScript at execution time	The first <button> element that has disabled attribute
	UpdateHiddenInputs (UHI)	Replacing type attribute value of <input> elements or removing the "disabled" attribute of <input> elements and adding a <script> to put the "disabled" attribute back by JavaScript at execution time.	The first <input> element that has sensitive "hidden" values
	UpdateIframe (UI)	Replacing style's attribute value of <iframe> elements.	The first <iframe> element that has sensitive "hidden" values
	UpdateTitle (UT)	Replacing the original title with domain name and adding a <script> to put the original title back by JavaScript at execution time.	<title> element

REFERENCES

[1] APWG, "Phishing activity trends reports," *[Online]. Available: https://apwg.org/trendsreports/*, 2024.

[2] A. Abbasi, F. ahedi, D. Zeng, Y. Chen, H. Chen, and J. F. Nunamaker Jr, "Enhancing predictive analytics for anti-phishing by exploiting website genre information," *Journal of Management Information Systems*, vol. 31, no. 4, pp. 109–157, 2015.

[3] C. Opara, B. Wei, and Y. Chen, "Htmlphish: Enabling phishing web page detection by applying deep learning techniques on html analysis," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[4] L. Ouyang and Y. Zhang, "Phishing web page detection with html-level graph neural network," in *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2021, pp. 952–958.

[5] A. Aljofey, Q. Jiang, A. Rasool, H. Chen, W. Liu, Q. Qu, and Y. Wang, "An effective detection approach for phishing websites using url and html features," *Scientific Reports*, vol. 12, no. 1, p. 8842, 2022.

[6] G. Apruzzese, M. Conti, and Y. Yuan, "Spacephish: The evasion-space of adversarial attacks against phishing website detectors using machine learning," in *Proceedings of the 38th Annual Computer Security Applications Conference*, 2022, pp. 171–185.

[7] B. Montaruli, L. Demetrio, M. Pintor, L. Compagna, D. Balzarotti, and B. Biggio, "Raze to the ground: Query-efficient adversarial html attacks on machine-learning phishing webpage detectors," in *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, 2023, pp. 233–244.

[8] IBM, "What is red teaming?" *[Online]. Available: https://www.ibm.com/think/topics/red-teaming*, 2024.

[9] A. Safi and S. Singh, "A systematic literature review on phishing website detection techniques," *Journal of King Saud University-Computer and Information Sciences*, vol. 35, no. 2, pp. 590–611, 2023.

[10] F. Charmet, H. C. Tanuwidjaja, T. Morikawa, and T. Takahashi, "Towards polyvalent adversarial attacks on url classification engines," in *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, 2022, pp. 1246–1248.

[11] H. Shirazi, S. R. Muramudalige, I. Ray, A. P. Jayasumana, and H. Wang, "Adversarial autoencoder data synthesis for enhancing machine learning-based phishing detection algorithms," *IEEE Transactions on Services Computing*, vol. 16, no. 4, pp. 2411–2422, 2023.

[12] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 8, pp. 3779–3795, 2021.

[13] R. Ebrahimi, J. Pacheco, J. Hu, and H. Chen, "Learning contextualized action representations in sequential decision making for adversarial

malware optimization," *IEEE Transactions on Dependable and Secure Computing*, 2024.

[14] S. Banik, T. Ramachandran, A. Bhattacharya, and S. D. Bopardikar, "Automated adversary-in-the-loop cyber-physical defense planning," *ACM Transactions on Cyber-Physical Systems*, vol. 7, no. 3, pp. 1–25, 2023.

[15] S. Huang and S. Ontañón, "A closer look at invalid action masking in policy gradient algorithms," *arXiv preprint arXiv:2006.14171*, 2020.

[16] V. L. Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen, "Tranco: A research-oriented top sites ranking hardened against manipulation," *arXiv preprint arXiv:1806.01156*, 2018.

[17] D. Y. Hancock, J. Fischer, J. M. Lowe, W. Snapp-Childs, M. Pierce, S. Marru, J. E. Coulter, M. Vaughn, B. Beck, N. Merchant *et al.*, "Jetstream2: Accelerating cloud computing via jetstream," in *Practice and Experience in Advanced Research Computing*, 2021, pp. 1–8.