# Detecting video-game injectors exchanged in game cheating communities

Panicos Karkallis[1], Jorge Blasco[1], Guillermo Suarez-Tangil[2], and Sergio Pastrana[3]

[1] Royal Holloway, University of London, United Kingdom
[2] IMDEA Networks, Spain
[3] Universidad Carlos III de Madrid, Spain

**Abstract.** Video game cheats destroy the online play experience of users and result in financial losses for game developers. Similar to hacking communities, cheat developers often organize themselves around forums where they share game cheats and know-how. In this paper, we perform a large-scale measurement of two online forums, MPGH and UnknownCheats, devoted to video game cheating that are nowadays very active and altogether have more than 7 million posts. Video game cheats often require an auxiliary tool to access the victim process, i.e., an injector. This is a type of program that manipulates the game program memory, and it is a key piece for evading cheat detection on the client side. We leverage the output of our measurement study to build a machine learning classifier that identifies injectors based on their behavioural traits. Our system will help game developers and the anti-cheat industry to identify attack vectors more quickly and will reduce the barriers to study this topic within the academic community.

**Keywords:** Game Cheating & Hacks · Underground Forums · Injectors.

## 1 Introduction

There are more than 2 billion video game players worldwide, with many of them playing online [21]. Games such as *Counter Strike: Global Offensive* (CS:GO), released in 2012, still attract more than 600K monthly average players.[4] Most modern video games also include ranking systems that prompt players to compete to get more content and features. They also offer access to exclusive in-game events, that in some cases are used as an entry point for professional e-sports.

While most players compete using their ability and experience, others use game hacks and cheats to gain advantages against their competitors. Cheating in online games undermines legitimate player's efforts, reduces their engagement in the game, and results in financial losses [9]. To mitigate this, most modern games include anti-cheat software that continuously monitors the state of the game (or the system where the game runs) in the look for cheats. This has resulted in an arms race between cheaters and game developers.

[4] Data extracted from `https://steamcharts.com/app/730` on 16th April 2021.

Video game cheaters interact in online underground forums to share knowledge, and trade products and services, similar to other online communities focused on illicit and even illegal activities [14, 24, 1, 20, 27]. Communities dedicated to game cheats include a range of topics that go from cheating tutorials and documentation to the trading and free-sharing of cheating programs that can work with the latest game versions.

In this work, we conduct a measurement on two of the largest English-speaking online communities (forums) dedicated to game cheating: Multiplayer Game Hacking (MPGH)[5] and UnknownCheats (UC)[6]. These communities have been continuously operating since 2007. Our study provides a bi-dimensional view of the ecosystem, using both: i) social data science techniques on the forum data, and ii) binary analysis of files released for free. In a nutshell, we first shed light on how these communities are structured, the type of cheats and tools being developed, the actors involved, and the games they target. We observe that a cornerstone component of this ecosystem relies on stealthy techniques design to inject cheats into the games' program memory. Accordingly, we leverage information from our measurement to build a classifier to detect such injectors.

To the best of our knowledge, this is the first study focusing on longitudinal data on game cheats covering multiple games, years, and cheating communities (see §4). In particular, our work makes the following contributions:

- We analyze two of the most popular communities that focus on the trading and sharing of hacks and cheats using a novel methodology (see §2). Specifically, we apply social network analysis to describe the relationships and interests of the forum users (as shown in §3.3).
- We build a machine learning classifier to quickly identify injectors used to execute cheats within games. Our classifier uses features extracted from the static and dynamic analysis of binaries and is able to correctly classify 91% of the 632 test samples (§3.2).
- We discuss how our work could be used to help anti-cheat analysts and discuss the limitations involved when analysing these communities (§5).

## 2 Methodology

We rely on data gathered from two well-known English communities dedicated to video game hacks and cheats to perform an analysis of the cheating ecosystem. In particular, we analyse how users share and distribute cheats, the type of files being shared, and the demographics and interests of such users. We then focus our analysis on files that feature code injecting capabilities. These kinds of files, normally known as *injectors*, are a key component of the cheats ecosystem, as they are needed to inject the actual cheat payload into the game's process memory. As part of our framework, we also develop a method to quickly identify

[5] https://www.mpgh.net/
[6] https://www.unknowncheats.me/

cheat injectors based on their static and dynamic characteristics, using, among others, features commonly shared with malware.

Figure 1 presents an overview of the pipeline used for our analysis. Our framework starts with the data collection and pre-processing of the forum data obtained from the two online communities that are part of our analysis: *MPGH* and *UC* (§2.1). Once the data is collected and pre-processed, we analyze the content of each forum post and its corresponding attachments. These two analyses are done independently. On the one hand, the post analysis focuses on the relationships between users and video game cheats (§2.2). On the other hand, the attachment analysis looks into the kind of files that are shared by these communities for cheating purposes (§2.3), with a specific focus on attachments used to inject the cheats in the games' processes (§2.4).

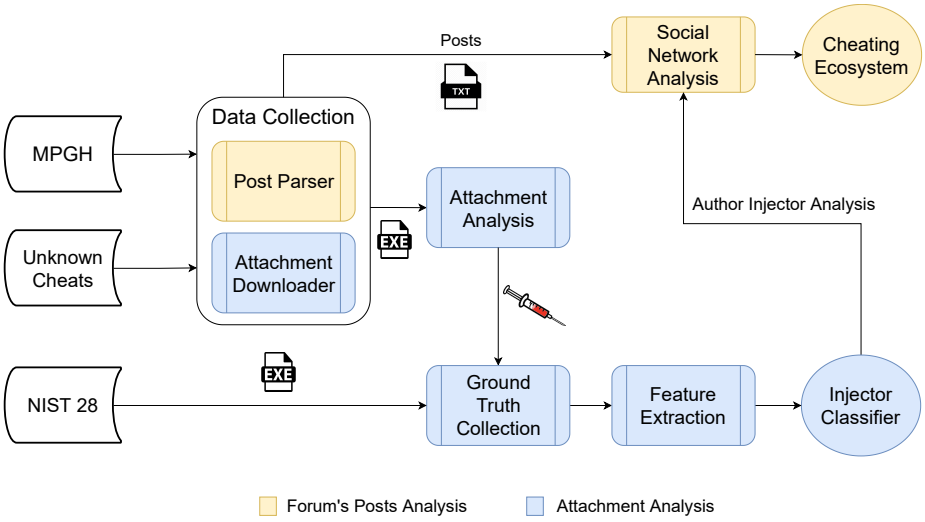


Fig. 1: Steps taken to process the posts and the attachments from the forums.

## 2.1 Data collection

Acquiring data from some online communities is a daunting task. We use data from the CrimeBB dataset [25], available from the Cambridge Cybercrime Centre[7]. While this dataset contains data from various underground forums, we focus on two popular English forums for video game hacks and cheats,i.e., Multi Player Game Hacking (MPGH) and UnknownCheats (UC). Our methodology is specifically designed to cover both MPGH and UC as provided by the CrimeBB dataset. However, it can be adapted to other data sources such as non-preprocessed forums or other online communities. Our dataset includes over a decade of posts

[7] `https://www.cambridgecybercime.uk`.

as of June 2020, totaling more than 9M posts made in 767K threads by 511K actors in MPGH, and 1.8M posts made in 120K threads by 184K actors in UC[8].

Both forums are organized around the main index page that hosts several sub-forums grouped in categories and sub-categories. For MPGH, all game forums are grouped under the "MultiPlayer Game Hacks & Cheats" category. For UC the categories group together game genres, e.g.: shooters, strategy, etc. The sub-categories match games or games from the same franchise (e.g., the Call of Duty franchise has 14 different games at the time of writing this paper). In both cases, the communities have other non-game-related forums covering a great variety of topics that go from programming to other topics outside the gaming domain. Additionally, MPGH provides specific sub-forums and marketplaces, where users can trade their hacks and cheats, and also other gaming-related goods such as virtual items, accounts, or tutorials.

Besides the text obtained from the posts, our system also analyzes the resources (cheats and auxiliary tools) shared in the forum. Both communities allow users to share files in the form of attachments (internal links to hosted resources). Links used to share attachments follow a specific format, and are thus labeled with the tag ****ATTACHMENT**** in the CrimeBB dataset. For this work, we have developed a custom crawler that automatically scrapes the text from the posts, extracts the URL of the links, and downloads all the attachments[9].

### 2.2 Post Analysis

Analyzing forum data is of particular interest for this research, since it provides both the context and metadata about the attachments (e.g. timestamp, game, accompanying explanations, etc.), and also the impact or popularity of the cheat providers in the community (e.g. number of downloads, reputation of the actor, number of replies, etc.) Since the dataset spans more than 15 years, processing this data allows conducting a longitudinal analysis to analyze the evolution of the interests and discussions in the community. Accordingly, we have applied specific techniques used to analyze social media data. Concretely, we use text mining to understand the topics covered in the posts, and Social Network Analysis (SNA) to understand the roles, connections, and social interactions of the actors releasing cheats.

**Game annotation** Both communities have special sub-forums dedicated to releases and requests of hacks and cheats for each game. These typically have the structure "[GAME_TITLE] hacks and cheats". Accordingly, we annotate these sub-forums and map them to the referred game, by manually looking at forum titles and extracting the particular game where possible. For this, we used our

[8] In the remainder of the paper, we use the terms 'user' and 'actor' indistinguishably to refer a forum account uniquely identified by a user ID.

[9] As a result of our work, these attachments have been included in the CrimeBB catalog, and are thus available for other researchers under a legal agreement with the Cambridge Cybercrime Centre.

own domain knowledge obtained from studying other gaming platforms such as Steam[10] and other gaming-specific media websites and online shops. This resulted in 238 games being identified overall from a set of 630 forums. We discard posts belonging to other forums. These may include posts that we would classify as releases but are not related to a particular game.

**Social Network Analysis** To understand the relevance and impact of the cheats and also the interests of cheat providers, we conduct a Social Network Analysis on the forum data. First, for each community, we build a directed graph $G = \{U, I\}$ that represents the historical interactions ($I$) of the users ($U$). Specifically, each pair of nodes $u_i, u_j \in U$ are connected by an edge $e_{ij}(w) \in I$ where $w$ is the number of times that the user $u_i$ responded or interacted with a post of user $u_j$. Then, we extract classical network centrality measures (degree, in-degree, out-degree and eigenvector) to analyze the impact and relevance of each user in the community. Additionally, we look for posts and threads related to the actual trading of virtual items, goods and services. For this analysis, we focus on the MPGH community, since it provides dedicated boards for the market. We count the number of posts made by cheat providers on these boards, which gives us an indicator of the economic activities of actors that are releasing cheats for free.

### 2.3 Attachment Analysis

We annotate every attachment with the corresponding game according to the board where it is shared. For each attachment where a cheat is released, we extract information about the posts where it was released[11]. Concretely we annotate the actor, the timestamp, whether a post is starting a new thread or if it has been a response to another post (e.g. a request), the number of replies to this post, the number of downloads of the attachment, and the game associated to the post. In most cases, multiple files related to a cheat are embedded into an archived file (e.g. zip or rar). In those cases, we annotate both, the zip file and the extracted files to the same forum. We recursively uncompressed all archive files, using a custom-made password cracking tool to inspect password-protected files. This tool scans the text from the post where the attachment is released, looks for a password reference, and saves all the successful passwords found in a passwords list. This list is used as a dictionary and all the entries are tested until the file is successfully uncompressed. All archive files that can not be uncompressed are discarded.

Apart from the context (i.e. forum-related data), for each attachment, we annotate the following information: the attachment name, its file type, a cryptographic hash (sha256), the author that posted the file, the number, and the list of files in the archive if the file is compressed, if the file contains an executable, if there is encryption, and the entropy [29]. Filetypes are obtained either by

[10] `https://store.steampowered.com` Accessed on 10th May 2021.
[11] Some attachments are duplicated or re-released in different posts.

looking at the file extension or by interpreting their magic number using the tool *libmagic*. Initial exploratory analysis showed that various files within the attachments were not directly related to the actual cheats. These are files used either as auxiliary tools or game and system files used as backups to recover from a corrupted version. To identify these, we look for all windows binary files (exe and dll) that are digitally signed. We analysed all digital certificates and found that none of them belonged to a cheat developer (they were related to trusted developers such as Microsoft, Adobe, etc.). Because of this, we remove these files from the analysis and classify the remaining files into a global category of executable files with sub-categories depending on their kind (Java, Python, PE file, etc.)

### 2.4 Injector Classifier

An initial analysis of the executable files and their corresponding posts revealed that many of them were not actual cheats. As mentioned earlier, while these forums are focused on game cheating, they also include boards related to other topics such as programming, graphics, etc. Our initial exploratory analysis also showed that most of the game boards included several releases that were focused on cheat injection. These 'injectors' or 'loaders' can be used by other cheat developers to inject their cheats into the game memory, allowing them to focus on the cheating behaviour rather than on how to get the cheat into the game. Identifying these injectors quickly can allow game developers to fix or improve the detection capabilities of their anti-cheat engines sooner, reducing the window the game is susceptible to a particular cheat. Because of this, we decide to focus the rest of our analysis on these files. In particular, we develop a machine learning-based classifier capable of detecting if a particular binary has injecting capabilities. For this, we collect a series of injectors as ground truth, we extract static and dynamic properties, build a random tree-based classifier using 5-fold cross-validation.

**Ground Truth Collection** We have conducted further filtering over the 45,338 executables to identify cheat-related attachments solely focusing on injecting or loading cheats in-game memory. We shortlisted a dataset of 2,543 injectors by inspecting the filename and looking for keywords like *injector*, *loader* and *injektor*. After keyword matching against the filename, a manual inspection of the files was conducted to validate the dataset.

To further validate the injector dataset, we use PEfile[12] to extract the functions and libraries imported, and also to extract strings from the binaries. Then, we match these extracted symbols against a list of methods known to be associated with code injection. The list of methods was created from the following sources: i) Feng et al. [10] described in 2008 a list of methods associated with game cheat injection; ii) A book by Nick Cano that includes and extends the methods [4] published in 2016; and iii) the tool *Capa*, an open-source security

[12] `https://github.com/erocarrera/pefile` Accessed on 10th May 2021.

framework from FireEye that automatically identifies malware capabilities from binaries [12], and which includes some of the previously mentioned methods in their set of rules relevant to process injection. Overall our analysis includes 66 methods relevant to injection.

To train our model we include a negative class with files that are not injectors. To achieve this, we include binary files that contain as much diverse behaviours as possible. We use the files extracted from the *NIST NSRL database* [30] as our negative class files. This database collects signatures of known and traceable software applications that are meant to reduce the complexity of law enforcement and cooperate investigations, and therefore are expected to have a diverse set of behaviours without including cheat-injection characteristics.

**Feature Extraction** Game cheats, and injectors in particular, are binary files obtained from *untrusted* sources (the forums we analyse). In the context of game cheating, injectors are programs designed with the purpose of loading and executing the cheat's code, either in the game memory or in a third party library used by the game. Due to the growth of anti-cheat technologies, these injectors implement techniques to remain stealthy and evade detection, such as code obfuscation or encryption of the injector's binary. The analysis of obfuscated code is hard and time consuming. Thus the classifier is trained using features derived from both static and dynamic analysis information. To get these features, we rely on existing information from VirusTotal (VT)[13]. We observe that, in order to improve the trustworthiness of their files and to probe that their cheats are *safe*, cheat providers often upload their binaries to VT, and link the corresponding report in their release post. We take advantage of this behaviour to obtain and extract static and dynamic features for our dataset. We query VT for behavioural reports from our set of injectors and non-injector files. Out of the 2,543 injectors, 1,426 produced behavioural reports containing dynamic analysis information. The behavioural reports along with the static information get parsed and converted into a feature matrix consisting of injectors (samples) as rows and information taken from the reports and static analysis as columns (features). In a similar way, we query VT for reports from non-injector files obtained from the NIST NSRL database. We obtain 1,731 reports that include all our required features. This means that our ground truth set consists of 1,426 injectors and 1,731 non-injectors (good-ware) files. All the files included in the dataset are unique as they have been filtered using their SHA-1 signature.

We group all our features in 7 categories: file operations, registry changes, processes & mutexes, services, functions, strings, and others (which includes a variety of features that cannot be grouped around a common theme). Table 1 shows the number of features from each category. We use one-hot encoding for all our features, except for the number of processes created which is numerical. As an example, each possible loaded module corresponds to a specific feature (e.g. *ole32.dll*) with a value of 1 if the module is actually being loaded by the

[13] `https://www.virustotal.com` Accessed on 10th May 2021.

binary and 0 otherwise. We provide more detailed information about the features in Appendix A.

| Analysis Class | Number of features |
|---|---|
| File Operations | 8,068 |
| Registry Changes | 5,440 |
| Strings | 5,161 |
| Processes & Mutexes | 1,101 |
| Functions | 398 |
| Services | 44 |
| Other | 214 |

Table 1: Summary of the features extracted from the ground truth collection using one-hot encoding.

**Model creation and validation** We use the ground-truth extracted to train a classification model. The accuracy of the model was calculated using K-fold cross-validation [26], with $K = 5$. The dataset was randomly divided to 70% training data and 30% test data. The performance of the model was measured using the f-score which takes into account the precision and recall for each class [6]. We tested different classification algorithms, i.e., SVM, random trees and neural networks, obtaining similar performances. Thus, for our experiments, we used an extremely randomized trees classifier [3]. The classifier's accuracy was tested with a different number of trees, with no significant change in performance. The performance results of our injector classifier are described in §3.2 along with a bias analysis.

## 3 Results

In this section, we present the results of our analysis. We start with a dataset characterization, including the type of files released in the cheats. We continue analyzing the performance of the injector classifier, and some case studies for injectors detected in our dataset. Finally, we present the results obtained from the analysis of the forums, including the demographics of the actors involved, as well as the activity related to the trading of a subset of actors that release more cheats and more injectors.

### 3.1 Dataset Characterization

Table 2 summarizes our dataset. In MPGH there are 86,789 posts with links to attachments, from which we have obtained 168,096 links and downloaded 160,991 attachments (some of the links were duplicated or death links). From

these, 119,715 (74.3%) corresponded to images and thus were excluded from the analysis. From the remaining, 376 files were not processed; 10 files consisting of password-protected files which were not cracked by our password cracking tool (§2.3), 197 files were corrupted and 130 were multi-archives. In UC there are 16,836 posts from where we have downloaded 21,265 attachments (images excluded). A total of 146 files were not processed; 75 archives were skipped as they were multi-part and 71 were corrupted.

| | | MPGH | UC |
|---|---|---|---|
| Structure | Forums | 752 | 227 |
| | Cheat Forums | 555 | 140 |
| | Games | 191 | 118 |
| | Actors | 511,440 | 184,568 |
| Threads in game forums | Overall | 449,832 | 85,454 |
| | With attachments | 31,705 | 4,552 |
| Posts in game forums | Overall | 5,809,108 | 1,203,745 |
| | With attachments | 36,688 | 7,049 |

Table 2: Summary of the data extracted from the Multiplayer Game Hacks (MPGH) and the UnknownCheats (UC) forums.

Out of all the data collected, we focus our analysis on forums that are dedicated to discussing specific games as indicated by the title of the forum, e.g. *Fortnite Hacks & Cheats*. We have mapped 630 forums to 309 game-specific forums out of a total of 238 unique games (see Table 2). We thus filter out attachments posted in forums not related to particular games (11,854 attachments). Overall, we see 5.6M game-related posts and 35K attachments in MPGH, and 1.4M posts and 7K attachments in UC.

Figure 2 shows the top 10 games in the last two years judging by the number of posts discussing each game. The total volume of posts in MPGH is much higher but is decreasing, while in UC it remains stable. This suggests that UC is becoming the principal community for cheats. Moreover, while it is more visible for UC, both communities experienced an increase in the number of posts since the beginning of March 2020. This matches the time when COVID-19 related lockdowns were imposed globally, and it is consistent with existing reports that show an increase in cheating activity by means of Denial of Service (DoS) attacks during the lockdown period [8]. When looking at individual games, we see that CS:GO is the most popular game in the two communities.

Table 3 provides a more detailed breakdown of the different file types found in the attachments. The *EXE* category includes all sorts of executable files, including MS-DOS, scripts, ELF and COFF (Unix), LSB, and Mach-O files. An insight that can be directly obtained from these results is the focus of each community: UC users are more interested in technical discussions around cheats,

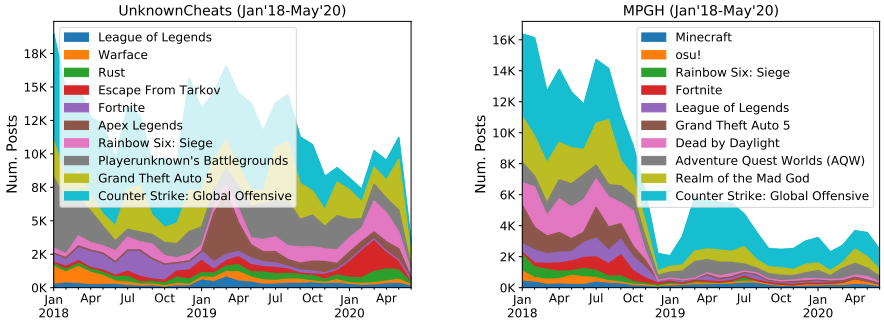


Fig. 2: Top 10 games by number of posts in UnknownCheats (left) and MPGH (right) from January 2018 until May 2020.

and thus they widely share the source code used to create cheats (e.g. 44% of the files from UC are C/C++ files, as opposed to MPGH where only 6% are of these types).

| | MPGH Total(%) | UC Total(%) | | MPGH Total(%) | UC Total(%) |
|---|---|---|---|---|---|
| Documents | 363,167(37.4) | 150,290(37.3) | DLL | 21,760(2.2) | 7,273(1.8) |
| Data file | 164,293(16.9) | 17,964(4.5) | Multimedia | 20,825(2.1) | 3,493(0.9) |
| Image | 163,369(16.8) | 10,513(2.6) | Game files | 14,349(1.4) | 4,100(1.0) |
| Java | 59,779(6.2) | 362(0.1) | Emails | 13,916(1.4) | 9 (0.00) |
| C | 35,975(3.7) | 51,101(12.7) | Scripts | 12,357(1.3) | 5,610(1.4) |
| C++ | 33,024(3.4) | 127,216(31.6) | HTML | 12,051(1.2) | 2,892 (0.7) |
| EXE | 26,704(2.7) | 7,722(1.9) | Other | 28,664(2.9) | 14,099(3.5) |

Table 3: Summary of attachment file types in MPGH and UC.

### 3.2 Injectors Classifier

Injecting code (e.g., in the form of a DLL) in a process is a widely-used method to piggyback into the execution context of a process. While this technique can aid programmers in the process of bypassing the restrictions of an operating system [28], it is also used by legitimate programs (e.g., Anti Viruses).

We evaluate the classifier using K-fold cross-validation as described in §2.4. We report the performance of our classifier using the average f-score measure[6]. Overall, our system reports an f-score of 91%. The model performs well on both classes with 0.94 *sensitivity* (TPR). When comparing the two classes bias we can observe a 0.11 FPR on the injector class and 0.03 FPR on the goodware class which contributes to a *specificity* of 0.97. Figure 3 shows the ROC curve

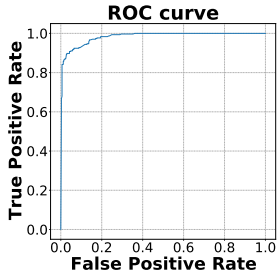


Fig. 3: Receiver Operating Characteristic (ROC) curve for the class of injectors showing 0.98 area under the curve.

and summarizes the trade-off between the FPR and the FPR. Having both high *sensitivity* and *specificity* rate translates to the classifier being able to identify injectors as well as files that are not classed as injectors. However, obtained FPRs indicate that our classifier is better at identifying what is *not an injector* (our goodware, negative class). If appropriately built into an automatic analysis system, our classifier could be used by anti-cheat analysts to quickly discard newly uploaded files that are not of interest, reducing the number of files and posts that need to be manually reviewed.

To better understand the classifier output, we run it against the set of 12,035 binary files for which we have information from the VT sandbox. From these, 45% were classified as injectors. We applied a set of heuristics to them to further understand the output. For example, 512 (9.3%) of these files mention the injection capabilities of the binary either in the thread title (418) and/or in the filename of the archive where the binary is included (253). This provides strong evidence that these files are actual injectors. Also, we confirm that some binaries, although not released as general-purpose injectors, implement injection capabilities. This is the case of an auxiliary tool used to change the MD5 hash of the cheat process before it is analysed by the corresponding anti-cheat engine to evade basic signature-based detection. Since this is done in runtime, the tool needs to inject itself into the game process memory. Another example is one tool that is (was) used to inject code in Flash processes and replace ActionScript Bytecode (ABC) to abuse Flash-based games. This tool was first tagged as malicious by some anti-virus vendors (and then allow-listed) due to its potential to 'inject malicious code into SWF files[14]. Another example of such a file classified as an injector is a tool called LeagueDumper[15]. This tool gets injected into the League of Legends game client and stores memory components of the games' process back to disk for analysis. While this tool is not directly used to inject a game cheat, it helps cheat developers to analyse the game binaries (which are encrypted with a custom packer to avoid this kind of behaviour).

[14] `https://community.mcafee.com/t5/Malware/quot-False-Artemis-4DD89AF63CF7-quot/m-p/521383` Accessed on 10th May 2021

[15] `https://github.com/tarekwiz/LeagueDumper` Accessed on 10th May 2021

### 3.3 Forum analysis

In this section, we analyze the social aspects of the two underground communities studied. We report about the demographics of the actors involved, including the social relations of these actors and their interests in the marketplace section.

**Demographics** As seen in Table 2, the forum dataset is comprised of more than 511K and 184K members in MPGH and UC respectively. However, only a small proportion of these are involved in the actual provision of cheats. We consider a cheat is provided if: i) the attachment contains a binary file (e.g. DLL or EXE) and ii) the attachment is released on a game board. We found that 4,522 actors on MPGH (0.9%) and 2,476 on UC (1.3%) have shared at least one attachment in gaming forums. Figure 4 shows the distribution of actors according to the number of attachments and the number of games where these attachments are shared. A large proportion of authors have shared more than one attachment (around 51% in MPGH and 42% in UC). Also, around 8% of users in MPGH (377) have provided more than 10 attachments (and indeed, a single actor provides a total of 290 attachments). This shows that, while many users participate in the sharing of cheats, the majority of the cheats are shared by a small subset of the community. We also see differences in the specialization of users. Left-hand plots in Figure 4 show the different games for which actors have shared attachments (we only consider those actors providing more than 2 attachments). We observe that most users (65% in MPGH and 60% in UC) are specialized in a single game (e.g. we see a single user in MPGH who has shared 218 different attachments in just one game, or one actor that has shared 286 attachments in just two games). However, there are users that are not specialized in particular games, and indeed we find that UC users tend to be less specialized, with some users sharing attachments in up to 17 different games.

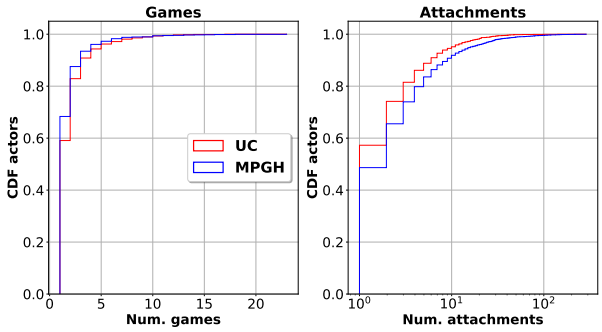


Fig. 4: Distribution of actors by number of attachments (right) and games (left) in MPGH and UnknownCheats (UC).

**Marketplace activity** Our analysis is focused on data that is publicly released in the forums. Nevertheless, it is well known that users in underground forums do share material related to cybercrime activities for free in order to increase their reputation and thus gain an advantage against competitors in the marketplace [15]. Also, underground forums have become improvised marketplaces of all sorts of virtual and online goods [20, 27, 24], and indeed MPGH has a set of sub-forums dedicated to trading. Estimating the actual volume of sales/benefits of a particular user is challenging — most of the actual trades occur by means of private messages and chats. However, the info gathered from these users in the market-related forums serves as a good estimation of the financial activity of these users. Accordingly, we have analyzed the number of posts made in the marketplace by the actors in MPGH that have released at least one binary attachment in gaming forums. Table 4 shows the number of actors posting and the total posts written in marketplace-related sub-forums. It can be observed that the most popular forum among cheat providers is about selling accounts/keys/items. This confirms that cheat providers are also engaged in Real Money Trading, which is a well-known practice to monetize cheats, either individually or as part of a gang or Gold Farming Group (GFG) [18]. Two other popular sub-forums are *Giveaways* and *Scammer Grave*. The former is typically used by forum users to increase their reputation and popularity by offering free goods, services, and even cryptocurrencies. The latter is a miscellaneous forum for reporting and discussing scams. The fact that cheat developers are reporting or discussing scams is another indicator of them being engaged in trading. Table 4 compares the activities of cheat providers with the top 100 actors according to the number of injectors released. Overall, we see that top cheat providers have higher interaction in the marketplace than the injector providers. Also, we observe that, rather than selling actual goods, in general, injector providers are more active in the *Giveaway* section. This might be due to the fact that we classify these actors by looking at the injectors that have been released for free (recall from Section 2.1 that we do not pay for these products and thus all the analysis is based on freely available attachments). Also, we see that 12 of these actors are actively engaged in the trading of eBooks, with an average of around 92 posts per author on this board.

**Social relations** Finally, we analyze the influence or popularity of users providing cheats and providing injectors in the entire community. For such purpose, we apply Social Network Analysis using the techniques described in §3.3. First, we build the graph of the two communities, MPGH and UC. Then, we compute the in-degree centrality metric of all the users in each forum, which indicates, for a given user, how many replies he/she has received, and from how many peers. Then, we compute three ranks: $R_A$, $R_I$ and $R_N$. The first one ranks the actors based on the number of cheats released, the second one ranks them based on injectors, and the latter ranks actors based on their popularity in the community, derived from network centrality measures. For each actor we calculate the quartile within each rank, so we can compare whether higher positions in

| Forum | All actors P/A | #Actors | #posts | Top 100 injector providers P/A | #Actors | # posts |
|---|---|---|---|---|---|---|
| Selling Accounts/Keys/Items | 31.14 | 1,278 | 18.10% | 17.90 | 40 | 8.94% |
| Giveaways | 49.42 | 886 | 19.91% | 25.79 | 43 | 13.85% |
| Scammer Grave | 51.70 | 651 | 15.31% | 39.64 | 33 | 16.33% |
| Buying Accounts/Keys/Items | 11.06 | 603 | 3.03% | 2.35 | 20 | 0.59% |
| User Services | 18.68 | 597 | 5.07% | 12.09 | 23 | 3.47% |
| Trade Accounts/Keys/Items | 8.21 | 488 | 1.82% | 7.36 | 14 | 1.29% |
| eBooks For Sale | 24.44 | 358 | 3.98% | 92.08 | 12 | 13.80% |
| Marketplace Talk | 14.86 | 335 | 2.26% | 6.00 | 11 | 0.82% |
| Elite* | 9.37 | 320 | 1.36% | 3.15 | 13 | 0.51% |
| Marketplace Price Check / Questions | 7.37 | 306 | 1.03% | 15.38 | 13 | 2.50% |
| OTHERS (38) | 11.86 | 3,828 | 28 % | 13.90 | 153 | 37 % |
| TOTAL | 48.62 | 4,522 | 219,854 | 80.10 | 100 | 8,010 |

Table 4: Activity of cheat providers in terms of number of posts and average posts per actor (P/A) in each sub-forum. *Marketplaces for specific games.

one rank correspond with higher positions in the other rank. Figure 5 shows the relations between the quartiles for the three ranks $R_A$, $R_I$ and $R_N$ (denoted $QAi$, $QIi$ and $QNi$ for $i \in [1, 4]$ respectively). It can be observed that most of the users that are in the first quartile of $R_A$ are also in the first quartile of $R_N$, in both MPGH (55%) and UC (53%). This indicates that cheat providers are, in general, popular in their corresponding social network. A similar pattern can be observed for those providing injectors, though with a lower difference (39% and 34%n MPGH and UC respectively). However, there are also various actors which are popular and influencing users (i.e., from $QN1$), and share a few cheats and injectors, or don not share at all. Note that these forums are not used only for trading, but also for exchanging knowledge and meet peers. Thus, being socially influencing is not a sign of being a cheat provider. Instead, we observe that the free provision of cheats and injectors allows users to increase their popularity in the community.

## 4 Related Work

**Underground communities** Underground forums serve for the sharing and trading of illicit products and services, and also for exchange of knowledge [20]. Due to the anonymity and the sense of lack of prosecution, they are an attractive source for initiating into cybercrime activities [22, 24]. Research on such forums allow to understand both old forms of cybercrimes, e.g. hacking [1, 20] or game cheating [14], and also new forms of online fraud, such as e-whoring [15].

Previous works showed that cheating can be contagious within these communities, even when there is a clear social penalty associated with this practice [2, 32]. The pathways into video-cheat gaming hacking were studied by Hughes et al., who analyzed the relationships of actors that are engaged both in general

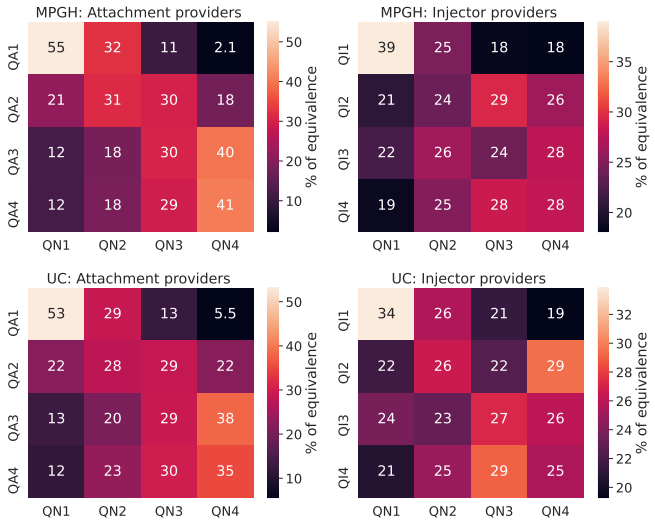


Fig. 5: Relation between number of users in each quartile according to number of attachments (QA, left-side), the number of injectors (QI, right-side) and popularity of the users in the social network (QN)

hacking and cheat development [14]. Fields et al. analyse cheat sites created for Whyville.net, an online gaming site to play casual science games [11]. They highlight the difference between hacks and cheats; and talk about the different types of online forums and their importance in the online gaming culture. Their work uses Grounded Theory [13] and was limited to a single community and 100 posts, which were visited weekly by the researchers to capture data for the study. In our work, we conduct quantitative analysis of two online communities with more than 7M posts overall, and also analyse the attachments shared by forum actors.

**Cheat Detection** Machine learning classifiers based on static and dynamic analysis features have been widely proposed in the security domain for malware detection [23, 5, 16, 7]. Other works have explored the use of machine learning to detect cheats based on the behavioural features exhibited by the players. Liu et al. proposed a method that uses bait targets as honeypots to detect aimbot-based cheats [19]. They performed a field measurement of two games, Counter Strike 1.6 and CS:GO. During one month and two weeks with 440 connections, they were able to identify 43 aimbots. More recently, Witschel and Wressnegger demonstrated that aimbots that introduce randomness while mimicking user improvement are capable of evading commercial anti-cheats [31]. Outside the academic world, some game developers, also provide updates on how their own games are affected by cheaters. In May 2020, Koskinas and Paloetti provided an update on the different cheats that were being reported and detected on *League*

*of Legends* and other games from developer *Riot Games* [17]. They show how they have improved their capability to detect and remove cheaters (from 4% of games in 2015 to 1% of the players in 2020) and how cheaters are better at identifying and reporting other cheaters while playing. To the best of our knowledge, this is the first academic work that proposes static and dynamic analysis of binary files to detect injectors used for video-game cheating.

## 5 Discussion & Conclusions

**Limitations.** Our measurement spans across two of the most popular online communities dedicated to video-game cheating and covers more than 200 games through multiple years. Our work mainly focuses on the analysis of binary files and we do not study how cheats are developed. The analysis of source code files and other scripts could provide valuable insight into the development process, but it is outside the scope of our work. Likewise, our work focuses on analyzing injectors for Windows-based games. We note however that not all game cheats affect Windows-based games. For instance, Pokemon Go, the 8th game within terms of number of cheats releases and it is only available on mobile platforms (Android and iOS). While our implementation can not currently analyze these files, our methodology is agnostic to the platform. Due to the nature of the features we use, our classifier is based on the features that these binaries exhibit today. As it happens in other domains, such as malware [16], these may change in the future as cheaters adapt to new detection techniques (e.g. Riot's Valorant[16]). While concept drift is outside of the scope of this work, the video game ecosystem provides additional data points (e.g. user-reports) that could be used to detect when the classifier becomes *outdated* and needs to be retrained.

**Conclusions.** In this work, we have performed a large-scale measurement of two online forums focused on video game cheating. Our study shows that these forums are widely used to distribute video game cheats, 40K since 2017 and that the majority of these cheats are developed by a small number of contributors that, in most cases, are specialized in a single game. Our results show that there is also a strong cooperation between the members of the community, which sometimes is promoted by financial incentive (i.e. selling cheats, etc. on marketplaces within the forum). Through our exploratory analysis we identify that cheat injectors are an important component gearing the cheating industry, but we also see that this is the weakest point of this economy. Thus, we develop a method to systematically detect injectors uploaded to the forum using both static and dynamic analysis features. Our classifier, which achieves a sensitivity of 0.94, could be used by anti-cheat analyst to quickly identify new injectors uploaded into these communities. Considering that between 2018 and 2020 the average number release posts per day was 7 (for each forum), using our classifier could help reduce the workload of the analyst when inspecting these new releases, preventing new cheats from becoming widespread within their games.

[16] `https://support-valorant.riotgames.com/hc/en-us/articles/360046160933`

## References


1. Allodi, L.: Economic factors of vulnerability trade and exploitation. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security. pp. 1483–1499. ACM (2017)
2. Blackburn, J., Kourtellis, N., Skvoretz, J., Ripeanu, M., Iamnitchi, A.: Cheating in online games: A social network perspective. ACM Transactions on Internet Technology (TOIT) **13**(3), 1–25 (2014)
3. Breiman, L., et al.: Arcing classifier (with discussion and a rejoinder by the author). The annals of statistics **26**(3), 801–849 (1998)
4. Cano, N.: Game hacking: developing autonomous bots for online games. No Starch Press (2016)
5. Chen, Y., Wang, S., She, D., Jana, S.: On training robust pdf malware classifiers. In: 29th USENIX Security Symposium USENIX Security 20). pp. 2343–2360 (2020)
6. Chinchor, N., Sundheim, B.M.: Muc-5 evaluation metrics. In: Fifth Message Understanding Conference (MUC-5): Proceedings of a Conference Held in Baltimore, Maryland, August 25-27, 1993 (1993)
7. Chumachenko, K.: Machine learning methods for malware detection and classification (2017)
8. Clayton, R.: The impact of lockdown on dos-for-hire. Tech. rep., Cambridge Cybercrime Centre COVID Briefing Papers (July 2020), `https://www.cambridgecybercrime.uk/COVID/COVIDbriefing-3.pdf`
9. Duh, H.B.L., Chen, V.H.H.: Cheating behaviors in online gaming. In: International Conference on Online Communities and Social Computing. pp. 567–573. Springer (2009)
10. Feng, W.c., Kaiser, E., Schluessler, T.: Stealth measurements for cheat detection in on-line games. In: Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games. p. 15–20. NetGames '08, Association for Computing Machinery, New York, NY, USA (2008). https://doi.org/10.1145/1517494.1517497, `https://doi.org/10.1145/1517494.1517497`
11. Fields, D.A., Kafai, Y.B.: "stealing from grandma"or generating cultural knowledge? contestations and effects of cheating in a tween virtual world. Games and Culture **5**(1), 64–87 (2010)
12. FireEye: Capa. `https://github.com/fireeye/capa`, `https://github.com/fireeye/capa`, accessed on July 2020
13. Glaser, B.G., Strauss, A.L., Strutzel, E.: The discovery of grounded theory; strategies for qualitative research. Nursing research **17**(4), 364 (1968)
14. Hughes, J., Collier, B., Hutchings, A.: From playing games to committing crimes: A multi-technique approach to predicting key actors on an online gaming forum. In: 2019 APWG Symposium on Electronic Crime Research (eCrime). IEEE (2019)
15. Hutchings, A., Pastrana, S.: Understanding ewhoring. In: 2019 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 201–214. IEEE (2019)
16. Jordaney, R., Sharad, K., Dash, S.K., Wang, Z., Papini, D., Nouretdinov, I., Cavallaro, L.: Transcend: Detecting concept drift in malware classification models. In: 26th USENIX Security Symposium (USENIX Security 17). pp. 625–642. USENIX Association, Vancouver, BC (Aug 2017), `https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/jordaney`
17. Koskinas, P., Paloetti, M.: Anti-cheat in lol (& more) (May 2020), `https://na.leagueoflegends.com/en-us/news/dev/dev-anti-cheat-in-lol-more/`, `https:`

//na.leagueoflegends.com/en-us/news/dev/dev-anti-cheat-in-lol-more/. Accessed on May 2020
18. Lee, E., Woo, J., Kim, H., Kim, H.K.: No silk road for online gamers! using social network analysis to unveil black markets in online games. In: Proceedings of the 2018 World Wide Web Conference. pp. 1825–1834 (2018)
19. Liu, D., Gao, X., Zhang, M., Wang, H., Stavrou, A.: Detecting passive cheats in online games via performance-skillfulness inconsistency. In: 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). pp. 615–626. IEEE (2017)
20. Motoyama, M., McCoy, D., Levchenko, K., Savage, S., Voelker, G.M.: An analysis of underground forums. In: Proceedings of the 2011 ACM SIGCOMM conference on Internet Measurement Conference. pp. 71–80. ACM (2011)
21. Narula, H.: A billion new players are set to transform the gaming industry (Dec 2019), https://www.wired.co.uk/article/worldwide-gamers-billion-players, https://www.wired.co.uk/article/worldwide-gamers-billion-players. Accessed on May 2020
22. National Cyber Crime Unit / Prevent Team: Pathways into cyber crime (Jan 2017), https://www.nationalcrimeagency.gov.uk/who-we-are/publications/6-pathways-into-cyber-crime-1/file, accessed on July 2020
23. Onwuzurike, L., Mariconti, E., Andriotis, P., Cristofaro, E.D., Ross, G., Stringhini, G.: Mamadroid: Detecting android malware by building markov chains of behavioral models (extended version). ACM Transactions on Privacy and Security (TOPS) **22**(2), 1–34 (2019)
24. Pastrana, S., Hutchings, A., Caines, A., Buttery, P.: Characterizing eve: Analysing cybercrime actors in a large underground forum. In: Research in Attacks, Intrusions, and Defenses (RAID). pp. 207–227. Springer, Heraklion, Crete, Greece (2018)
25. Pastrana, S., Thomas, D.R., Hutchings, A., Clayton, R.: Crimebb: Enabling cybercrime research on underground forums at scale. In: Proceedings of the 2018 World Wide Web Conference. pp. 1845–1854 (2018). https://doi.org/10.1145/3178876.3186178
26. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
27. Portnoff, R.S., Afroz, S., Durrett, G., Kummerfeld, J.K., Berg-Kirkpatrick, T., McCoy, D., Levchenko, K., Paxson, V.: Tools for automated analysis of cybercriminal markets. In: Proceedings of 26th International World Wide Web conference (WWW) (2017)
28. Richter, J., Nasarre, C.: Windows via C/C++. Microsoft Press, 5th edn. (November 2007)
29. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal **27**(3), 379–423 (1948)
30. Sherena.johnson@nist.gov: Nist special database 28 (Sep 2020), https://www.nist.gov/srd/nist-special-database-28
31. Witschel, T., Wressnegger, C.: Aim low, shoot high: evading aimbot detectors by mimicking user behavior. In: Proceedings of the 13th European workshop on Systems Security. pp. 19–24 (2020)
32. Woo, J., Kang, S.W., Kim, H.K., Park, J.: Contagion of cheating behaviors in online social networks. IEEE Access **6**, 29098–29108 (2018)

# A Analysis Features

This appendix lists the feature categories used to train the injector classifier along with the number of features within each category. The first column describes the feature category each analysis is part of as seen on Table 1.

| | Analysis name | Number of features |
|---|---|---|
| File Operations | files_opened | 3,938 |
| | files_deleted | 1,817 |
| | files_copied | 955 |
| | files_dropped | 589 |
| | files_written | 432 |
| | files_attribute_changed | 337 |
| Registry Changes | registry_keys_opened | 2,722 |
| | registry_keys_set | 2,603 |
| | registry_keys_deleted | 115 |
| Processes & Mutexes | processes_terminated | 321 |
| | processes_created | 311 |
| | processes_injected | 12 |
| | processes_killed | 1 |
| | processes_tree | 1 |
| | mutexes_opened | 103 |
| | mutexes_created | 352 |
| Services | services_opened | 33 |
| | services_started | 6 |
| | services_created | 3 |
| | services_stopped | 1 |
| | services_deleted | 1 |
| Functions | modules_loaded | 338 |
| | calls_highlighted | 53 |
| | signals_hooked | 7 |
| Strings | text_highlighted | 4,560 |
| | crypto_plain_text | 601 |
| Other | command_executions | 155 |
| | windows_searched | 40 |
| | permissions_requested | 13 |
| | crypto_algorithms_observed | 2 |
| | ip_traffic | 1 |
| | windows_hidden | 1 |
| | memory_pattern_domains | 1 |
| | memory_pattern_urls | 1 |

Table 5: Detailed categorization of features used for the injector classifier.