A Modular Architecture for Personalized Learning Content in Anti-Phishing Learning Games

Rene Roepke[1] Vincent Drury[2] Ulrik Schroeder[1] Ulrike Meyer[2]

Abstract: While game-based anti-phishing education earned a lot of attention in the last years, it can only attest to minor successes. Since developed games usually contain manually created and curated content, problems can occur when learners are faced with content they cannot easily relate to. This may hamper the motivation of learners and thus influence the learning experience negatively. Existing work proposes the personalization of learning games to address these problems, but does not go beyond a conceptual contribution. This paper provides an implementation of a personalization pipeline for two learning game prototypes and presents the modular, component-based architecture.

Keywords: Personalization; Game-based Learning; Content generation; Learner modeling

1 Introduction

As game-based learning emerges as a scalable, motivational and effective approach in security education for end-users, different anti-phishing learning games have been developed and reviewed by various researchers [DS16, HASB16, TMJ17, Ro20a].While these games aim for raising users' awareness and educating them on the recognition of malicious URLs or emails, the game content is often manually created with a games' target group in mind. Although developers can make an effort in creating suitable learning game content, the heterogeneity of learners makes it hard to fulfill requirements of different individuals. In case of anti-phishing education, where learners are presented URLs or emails of particular services, a lack of personalized learning content leads to various potential shortcomings. Unknown services can hamper raising awareness since more cognitive processes are needed for learners to transfer learned knowledge to their daily activities. In addition, for unknown services, a learner might be unable to decide whether a given URL is benign or not, due to a lack of reference. Both might have negative impact on motivation and the learning experience.

A solution to the aforementioned shortcomings is the personalization of learning game content towards a learner's individual characteristics. By adapting the presented services,

[0] This research was supported by the research training group "Human Centered Systems Security" sponsored by the state of North Rhine-Westphalia.

[1] RWTH Aachen University, Learning Technologies Resarch Group, Ahornstr. 15, 52074 Aachen, Germany [roepke,schroeder]@cs.rwth-aachen.de

[2] RWTH Aachen University, Research Group IT-Security, Mies-van-der-Rohe Str. 15, 52074 Aachen, Germany [drury,meyer]@itsec.rwth-aachen.de

URLs or emails to those the learners know, e.g. by considering their browser history, installed applications and more, the learning game could be tailored to individual learners [Ro20b]. The contribution of this paper lies in the implementation of a modular, component-based architecture for personalized learning games for anti-phishing education.

2 Related Work

In recent years, different contributions have been made to the fields of personalized learning and game-based learning. Concerning the intersection of these fields, Streicher and Smeddinck [SS16] provide an overview on the most prominent terms and concepts. They explicitly distinguish between *personalization*, *customization* and *adaptivity*, since they are often used interchangeably. While personalization describes "the act of changing a system to the needs of a specific individual user" [SS16] and customization if it is the needs of a user group, adaptivity describes solely the ability of a system to change over time and can consequently be used to achieve personalization or customization. Based on this distinction, we were only able to find examples of personalized learning games, which are based on adaptivity and not personalization of content [KEJ13, LKR08, Ki06].

Bakkes et al. [BTP12] present an extensive literature-based overview on personalized gaming. They emphasize the importance of learner modelling, as it is a requirement for personalized game experiences, and describe different types of adaptation within games, e.g. difficulty scaling and game mechanics adaptation. While it is more common for gameplay or difficulty to be adapted based on specific learner characteristics (e.g. learning styles [KEJ13]), the personalization of content is less well studied. In a previous publication, we presented the idea of a personalization pipeline for game-based anti-phishing education [Ro20b]. As it was only conceptual work, no implementation was presented.

To supplement these findings, we broaden our search to content personalization in other learning environments and in particular, implementations of content personalization. Here, Bezza et al. [BBM13] provide an overview on different methods for content personalization in e-learning systems. They distinguish between *inductive* (i.e. without user intervention) and *deductive* (i.e. with user intervention) user modeling. After presenting their classification scheme for content personalization methods, they review existing contributions.

Regarding the implementation of personalized learning systems, Ismail et al. [IB18] provide an abstract, reusable software architecture identifying four main compontents: (1) learner unit, (2) knowledge unit, (3) personalization unit and (4) presentation unit. These components are utilized with regards to modelling the user and personalizing access to learning resources. Ismail et al. [IB18] provide abstract, conceptual descriptions of each component and note that implementations may differ based on the implementation context. Lastly, they map exemplary personalized learning systems to their proposed architecture.

Although existing work in the fields of personalized learning and game-based learning does not provide explicit examples for content personalization in learning games but rather for adaptive, personalized gameplay (e.g. [Ki06, LKR08]), suitable approaches can be drawn from related fields. Our work builds on the idea of a personalization pipeline [Ro20b] and the reusable architecture for personalized learning systems by Ismail et al. [IB18].

3 Concept of the Personalization Pipeline

In previous work [Ro20b], we presented the idea of a personalization pipeline for anti-phishing learning games. The pipeline consists of three parts processing data about the learner in order to provide personalized learning game content. The three parts are: (1) data collection, (2) content generation and (3) content delivery. The pipeline is intended to precede a game and provide adaptations to gameplay or the configuration of a game. Each part of the personalization pipeline provides input for the next part.

For *data collection*, two different approaches can be followed: manual or automated. Both approaches provide a learner model as their output to the content delivery module. For anti-phishing learning games, the learner model consists of knowledge about used services and visited websites.

To generate suitable game content, the *content generation* module offers different content generators, e.g. a URL generator. These content generators are queried by the level generator and provide content ready to be embedded into a level definition. Depending on the type of game, additional content generators can be implemented, e.g. an email generator for games about email phishing.

For *content delivery*, the generated level definitions serve as input to a level controller. The level controller provides an interface to the game, generating levels depending on the current game state, i.e. different levels based on the learner's progress in the game. The interface between level controller and game is the only connection of the personalization pipeline with the game, making each component of the personalization pipeline modular and easy to replace with different implementations.

Depending on the type of game, the implementation of the pipeline can vary, and thus, we want to solidify the idea of a personalization pipeline and provide an implementation and architecture with an interface to two game prototypes.

4 Implementation

Our implementation of the personalization pipeline follows a modular, component-based design where each stage of the pipeline is represented by one module consisting of multiple components. Following the single-responsibility principle, each component serves a specific task or purpose. Data flow is managed with simple interfaces between components. Figure 1 depicts a component diagram of the architecture as well as data flow between different components or modules.

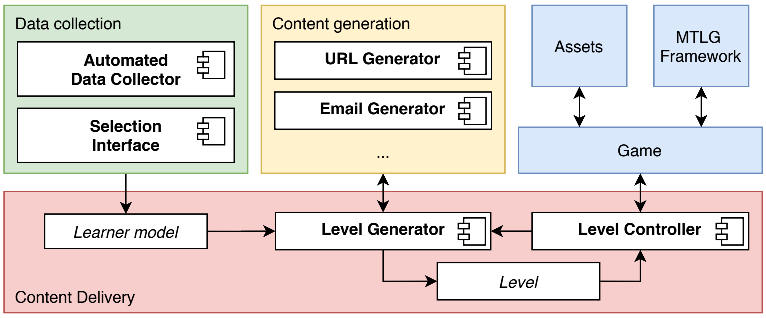


Fig. 1: Component diagram of the proposed architecture. Modules of the pipeline have different colors and include multiple components. Arrows represent data flow.

We plan to evaluate the pipeline making use of two existing game prototypes for anti-phishing education called “All sorts of Phish” and “A Phisher’s Bag of Tricks”. Both game prototypes address the structure of URLs as well as common manipulations for phishing purposes. While learners have to recognize the type of URL and sort them accordingly in the first game, the second game (see Figure 2) requires learners to create malicious URLs themselves by applying different manipulation rules.

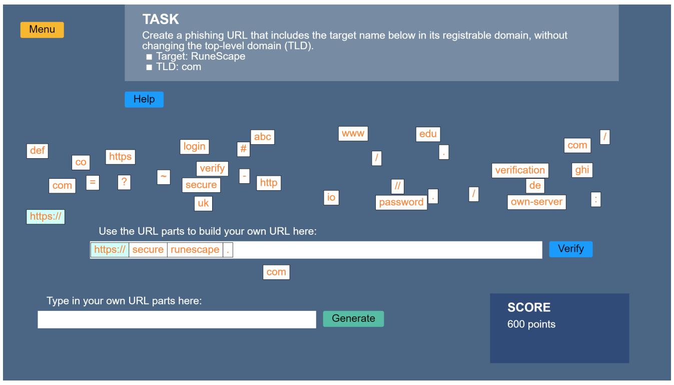
The games are developed using the Multi Touch Learning Game framework MTLG[3], which supports game development using the HTML5 Canvas element and native JavaScript. The games are implemented using the Model-View-Presenter pattern, while making use of different core components of MTLG via the (revealing) module pattern [Os12]. Both games are delivered using a simple web server, they run fully on the client side requiring no further server capacities. The only exception is a logging module, that can be set up to forward event log data to a remote server for evaluation purposes.

4.1 Data collection for learner modeling

All personalization efforts begin with the creation of a learner model, which is implemented in the data collection module. Since we mainly distinguish between two different approaches for data collection, i.e. manual and automatic, two components are provided in our data collection module.

The current game prototypes support manual, deductive learner modeling implemented in a *selection interface*, that shows pre-defined and popular services that the learners can then select or deselect, depending on their familiarity with the services. Services are divided into different categories (e.g. cloud storage, shopping, payment services) and displayed using pagination. The selection screen is developed as an independent component, that can

[3] `https://mtlg-framework.gitlab.io/`, last accessed 08-12-2020



TASK

Create a phishing URL that includes the target name below in its registrable domain, without changing the top-level domain (TLD).

- Target: RuneScape
- TLD: com

Use the URL parts to build your own URL here:

https:// secure runescape .

Type in your own URL parts here:

SCORE

600 points

Fig. 2: Level of the game “A Phisher’s bag of tricks”. The player has to create a malicious URL by combining available URL parts via drag-and-drop

be added to existing games to facilitate the simple deductive creation of learner models. Contrasting this manual selection is the automated, inductive approach, which uses a browser extension to collect services directly from a learner’s browser history. This approach is not yet suitable for large-scale studies, as there are major privacy concerns when collecting the browser history of end-users. We aim to handle these issues in the future by making sure that user data never leaves the browser.

The output of either approach is a learner model specified as a `json` object, that includes information about the learner’s familiarity with a number of services. If additional aspects need to be considered for the model, further data collection components can be integrated. As for now, the learner model does not actively involve in-game data. However, it could be extended, e.g. by collecting knowledge about the player’s progress or covered content to reflect the current state of the learner during the game.

4.2 Rule-based content generation

The next step in the personalization pipeline is content generation. In the current game prototypes, URLs can be generated dynamically to ensure a large variety of content, even for learner models with only few known services. The necessary functionality is provided by the *URL generator* component, which is used to transform benign services and their URLs into malicious URLs imitating those services. It can also create additional benign URLs. The generator takes service descriptions (consisting of a name and URL) as input, performs a number of rule-based modifications on the URLs, and outputs a number of malicious or benign URLs and details about their creation process. The rules are based on patterns that were extracted from real-world examples of benign and phishing URLs: Benign rules are based on URLs from the login pages of popular websites (retrieved from Alexa[4]), while malicious rules are based on a study of related work, and verified using the popular database Phishtank[5]. Several rules can be applied in sequence to create more complex and realistic URLs. The generators can also be exchanged to generate different types of content, for example, an *Email generator* component is planned for a future game prototype.

4.3 Content delivery via a level controller

The last step in the pipeline is content delivery. Conceptually, the content delivery module is the interface available to the game to request personalized content. While the pipeline was initially described as a three-step process, here, the content delivery module takes on the role of a controller, collecting necessary information with the help of several new components. First, the *level generator* component is used to generate personalized levels by embedding generated content into explicit level definitions for the games. The actual interface between delivery module and game is implemented in the *level controller*, which triggers the creation of new levels upon request by the game and returns appropriate level definitions. This cyclic relation allows for on-demand level generation during the game. Level definitions are tailored towards the specific game and include all information necessary to create a playable level, including specific task descriptions, conditions required to clear the level and the URLs that are to be used. Finally, the game handles the translation of level definitions into actual playable levels, by creating the required views and game logic.

5 Discussion

Besides our previous conceptual contribution [Ro20b], we considered the work of Ismail et al. [IB18] as a basis of our implementation, since they propose a reusable software architecture for personalized learning systems. Although the architecture is not specifically designed for personalized learning games, Ismail et al. describe abstract main components reusable in different implementation contexts which can be mapped to our architecture.

First, the “learner unit” in the work of Ismail et al. [IB18] can be directly mapped to our learner model, as both are responsible to maintain data about the learners. Next, the “presentation unit” maps to our two games, as they are the environment in which a learner interacts with the personalized content. While we argue that these two units can be easily mapped to components in our architecture, mapping the “knowledge unit” and “personalization unit” requires the clarification of a crucial difference between traditional e-learning environments and learning games. While Ismail et al. [IB18] consider learning resources to be courses, e-books, and similar, learning resources within a game environment are tutorials and tasks, which in particular include the generated learning game content.

[4] https://www.alexa.com/topsites, last accessed on 09-12-2020

[5] https://www.phishtank.com/, last accessed on 09-12-2020

Therefore, the mapping of a “knowledge unit” is more vague, but best fits the generator components and game assets. Lastly, the “personalization unit” cannot be mapped ideally either, as it includes both data collection and content delivery. We argue that this distinction improves our architecture, since our modular approach allows the exchange of individual data collection and even level generation components.

Currently, the implementation of the pipeline and prototype games focuses only on the personalization of services and URLs that appear in the games, other parts of the games are fixed. In particular, progress in the game and level difficulties are not adapted to the learner. This would require a notion of difficulty for the different levels, as well as an approach to update the learner model to reflect progress, strengths and weaknesses of the learner. Even though both of these requirements can already be modeled using the conceptual architecture of the personalization pipeline, they are not currently implemented, as they are not the focus of the research project.

Another open question is how to improve the data collection module. Currently, only the manual selection is used for creating a learner model. The integration of the automated approach for data collection needs to be evaluated carefully in regards to user privacy. In particular, even though the games themselves do not leak any user information, the logging functionality needs to be adjusted to not leak any information on URLs or services. With both approaches implemented and functioning, the hybrid approach, that uses the output of the automated approach as input to the manual selection, is also an alternative that is left to be analysed in more detail.

To summarize, the current prototypes show, that the personalization pipeline proposed previously [Ro20b] can be implemented using a modular, reusable architecture. The next step will be to evaluate the personalized games, to compare them to their non-personalized versions, as well as compare different data collection and personalization approaches regarding their effect on learning outcomes and the learner’s experience.

6 Conclusion and Future Work

In this paper, we presented an architecture for personalized anti-phishing learning games. To this end, we implemented a personalization pipeline consisting of three stages: data collection, content generation and content delivery. Our results show that the modular approach to pipeline and game creation makes it possible to retrofit personalized content into existing games, and facilitates improving and changing different components as required by the specific game.

For future work, we intend to evaluate the game prototypes and compare the personalized games to the non-personalized setting, to answer the underlying research question whether there are differences in learning outcomes and gaming behavior. Furthermore, we intend to create the automated data collection as an alternative approach to the manual selection interface and compare the approaches from a learner’s perspective.

Bibliography

[BBM13] Bezza, Assma; Balla, Amar; Marir, Farhi: An approach for personalizing learning content in e-learning systems: A review. In: 2013 Second International Conference on E-Learning and E-Technologies in Education (ICEEE). pp. 218–223, 2013.

[BTP12] Bakkes, Sander; Tan, Chek Tien; Pisan, Yusuf: Personalised Gaming: A Motivation and Overview of Literature. In: Proceedings of The 8th Australasian Conference on Interactive Entertainment: Playing the System. IE '12, Association for Computing Machinery, New York, NY, USA, 2012.

[DS16] Dewey, Chad M.; Shaffer, Chad: Advances in information SEcurity EDucation. In: Int. Conf. on Electro Information Technology. IEEE, Grand Forks, pp. 133–138, 2016.

[HASB16] Hendrix, Maurice; Al-Sherbaz, Ali; Bloom, Victoria: Game Based Cyber Security Training: are Serious Games suitable for cyber security training? Serious Games, 3(1):53–61, 2016.

[IB18] Ismail, Heba; Belkhouche, Boumediene: A Reusable Software Architecture for Personalized Learning Systems. In: 2018 International Conference on Innovations in Information Technology (IIT). pp. 105–110, 2018.

[KEJ13] Khenissi, Mohamed Ali; Essalmi, Fathi; Jemni, Mohamed: Toward the personalization of learning games according to learning styles. In: 2013 International Conference on Electrical Engineering and Software Applications. pp. 1–6, 2013.

[Ki06] Kickmeier-Rust, Michael D.; Schwarz, Daniel; Albert, Dietrich; Verpoorten, Dominique; Castaigne, J-L; Bopp, Matthias: The ELEKTRA project: Towards a new learning experience. M3–Interdisciplinary aspects on digital media & education, pp. 19–48, 2006.

[LKR08] Law, Effie Lai-Chong; Kickmeier-Rust, Michael D.: 80Days: Immersive digital educational games with adaptive storytelling. University of Graz, 2008.

[Os12] Osmani, Addy: Learning JavaScript Design Patterns: A JavaScript and jQuery Developer's Guide. O'Reilly Media, Inc., 2012.

[Ro20a] Roepke, Rene; Koehler, Klemens; Drury, Vincent; Schroeder, Ulrik; Wolf, Martin R.; Meyer, Ulrike: A Pond Full of Phishing Games - Analysis of Learning Games for Anti-Phishing Education. In (Hatzivasilis, George; Ioannidis, Sotiris, eds): Model-driven Simulation and Training Environments for Cybersecurity. Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 41–60, 2020.

[Ro20b] Roepke, Rene; Schroeder, Ulrik; Drury, Vincent; Meyer, Ulrike: Towards Personalized Game-Based Learning in Anti-Phishing Education. In: 2020 IEEE 20th International Conference on Advanced Learning Technologies (ICALT). pp. 65–66, 2020.

[SS16] Streicher, Alexander; Smeddinck, Jan D.: Personalized and Adaptive Serious Games. In (Dörner, Ralf; Göbel, Stefan; Kickmeier-Rust, Michael; Masuch, Maic; Zweig, Katharina, eds): Entertainment Computing and Serious Games, volume 9970 of Lecture Notes in Computer Science, pp. 332–377. Springer International Publishing, Cham, 2016.

[TMJ17] Tioh, Jin-Ning; Mina, Mani; Jacobson, Douglas W.: Cyber security training a survey of serious games in cyber security. In: 2017 IEEE Frontiers in Education Conf. (FIE). IEEE, Indianapolis, pp. 1–5, 2017.