Automatic Challenge Generation for Teaching Computer Security

Ricardo de la Rocha Ladeira, Rafael Rodrigues Obelheiro
{ricardo.ladeira@ifc.edu.br, rafael.obelheiro@udesc.br}

4 de outubro de 2018

Agenda

Introduction

Cybersecurity Games

Goals

Automatic Challenge Generation

Evaluation

Conclusion

Introduction

- Computer Security is a current theme.

Introduction

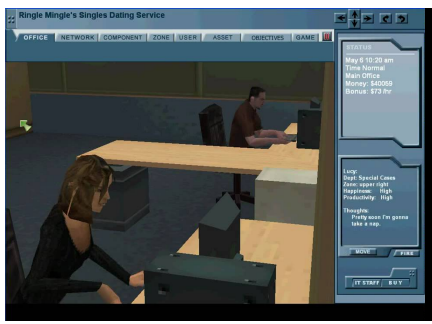
- Need for **cybersecurity culture**.
- Little knowledge of the general public and little manpower.

- Actions are needed in the context of formal education.
  - Research opportunity
  - Increase the acquisition of skills in Computational Security
  - Increase interest in the area and attract professionals.
  - **Games** can contribute to this.

Cybersecurity Games

- **Games** are an important pedagogical tool for Computer Security
  - Motivate
  - Teach

- Challenges, board games, videogames, attack and defense and others. Each one with different resources and target audience.



Cybersecurity Games

- **Challenges** (Treasure Hunt): problems solved with processes and tools, typically without interaction with other players.
  - Reverse engineer a file
  - Find a hidden file
- Flexible in complexity, resources, linearity etc.
- Difficulties
  - Problem-building requires specialized knowledge (which is scarce). Activity usually **manual** and laborious.
  - Reuse of problems
    - Loss of surprise factor
    - Sharing answers

Goals

- **General**: automate the problems generation for security competitions, obtaining unique instances of problems, in a way parameterizable by the organizer of the competition.

- **Specific**:
  - Model a treasure hunting competition;
  - Evaluate the competition effect in academic context; and
  - Measure students' perception of satisfaction in the activity.

Automatic Challenge Generation

Competition and Tool

- Individual competition, non-linear, challenge type.
- Find the secret word
- Ranking
- The tool
  - is used by the competition organizer, who chooses the exercises (simple and/or composite) of the competition;
  - generates unique instances of problems for each player;
  - automatically configures the DBMS;
  - sends instances of problems to the web server, accessible by application.

Automatic Challenge Generation

Challenge Generator

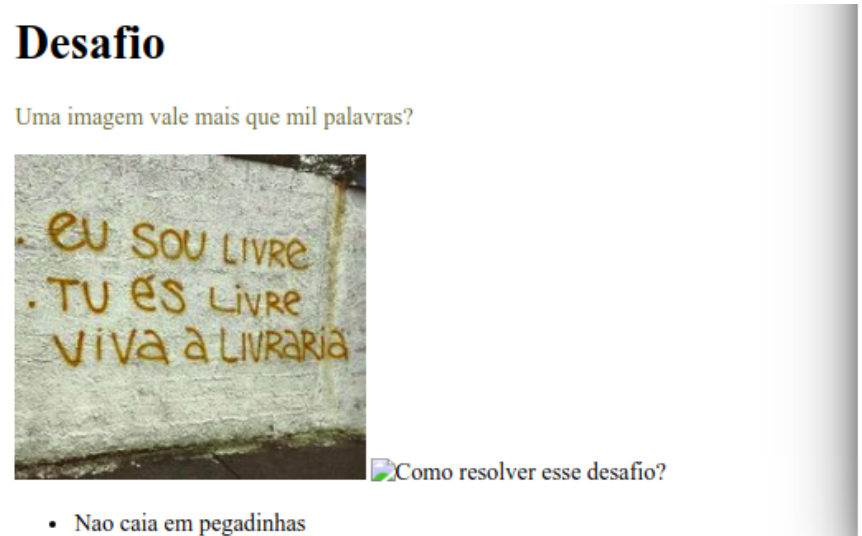
Figure: execution of the challenge generator script.

```
----------
Treasure Hunt!
----------
Informe a quantidade de DESAFIOS: 6
Informe a quantidade de JOGADORES: 10
----------
Vamos criar os desafios!
----------
Lista de problemas disponíveis:
1: (De)codificação de arquivo em base64
2: (Des)criptografia de Cifra de César
3: Comentário em código-fonte de página HTML
4: Comentário no arquivo robots.txt
5: (De)codificação de caractere ASCII para inteiro
6: Descompilar binário e obter fonte Java
7: Descompilar binário e obter fonte Python
8: Esteganografia em imagens
Obs.: escolha 1 ou 2 problemas. Exibiremos uma mensagem de erro se a composição
não existir.
----------
Informe o(s) problema(s) do desafio 1:
```

Automatic Challenge Generation

Challenge Generator

Figure: simple problem.



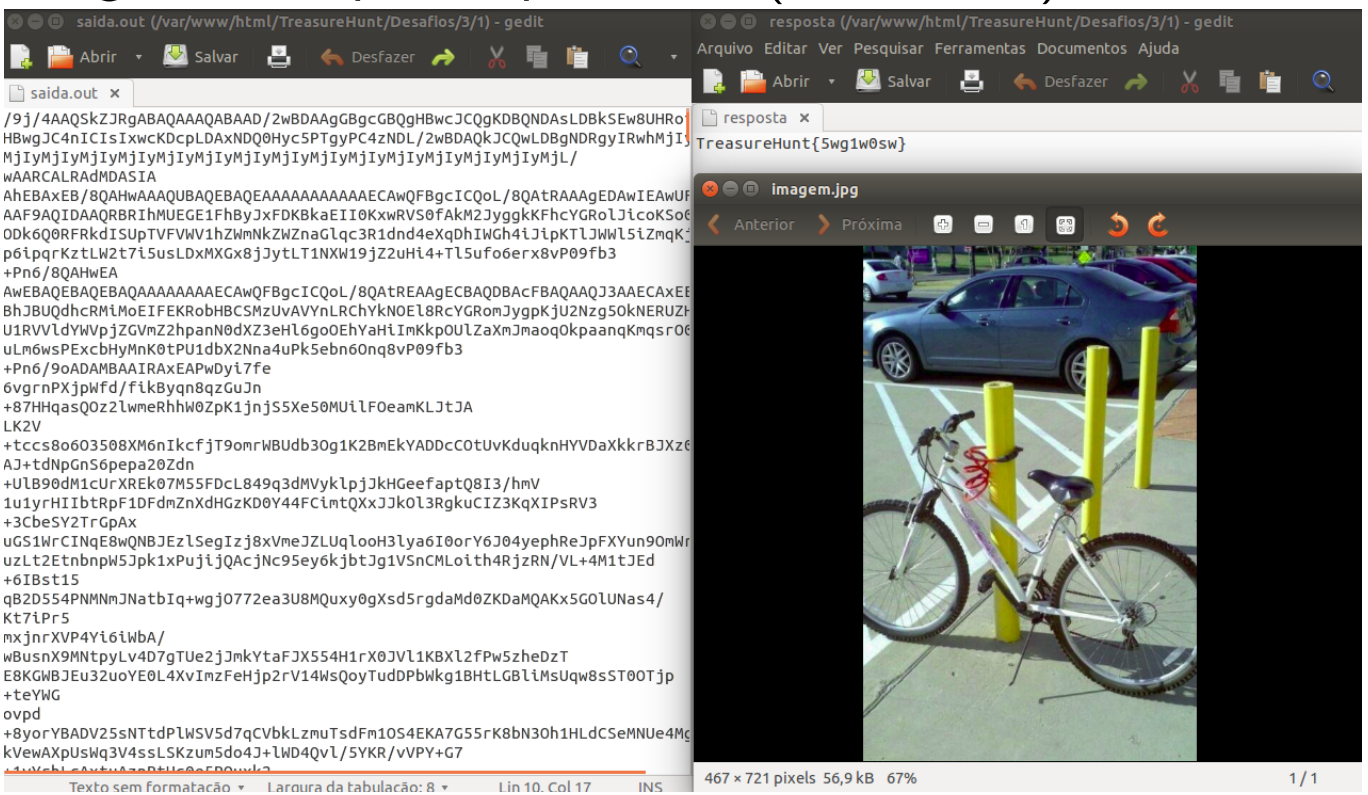

```
22          <img
    src="http://img.ibxk.com.br/2016/03/31/3120433266
    w=1040" alt="Imagem do desafio">
23          <img alt="Como resolver esse desafio?">
24          <ul> <!-- TreasureHunt{ago0tl7n2nbo} -->
25              <li>Nao caia em pegadinhas</li>
26              <li>Fique atento</li>
27              <li>Procure nao se distrair</li>
```

```
23          <img alt="Como resolver esse desafio?">
24          <ul>
25              <li>Nao caia em pegadinhas</li>
26              <li>Fique atento</li>
27              <li>Procure nao se distrair</li> <!--
    TreasureHunt{fx1vkee8iyta} -->
28              <li>Nao leia este item</li>
29              <li>Cuidado para nao perder tempo lendo textos
```

Automatic Challenge Generation

Challenge Generator

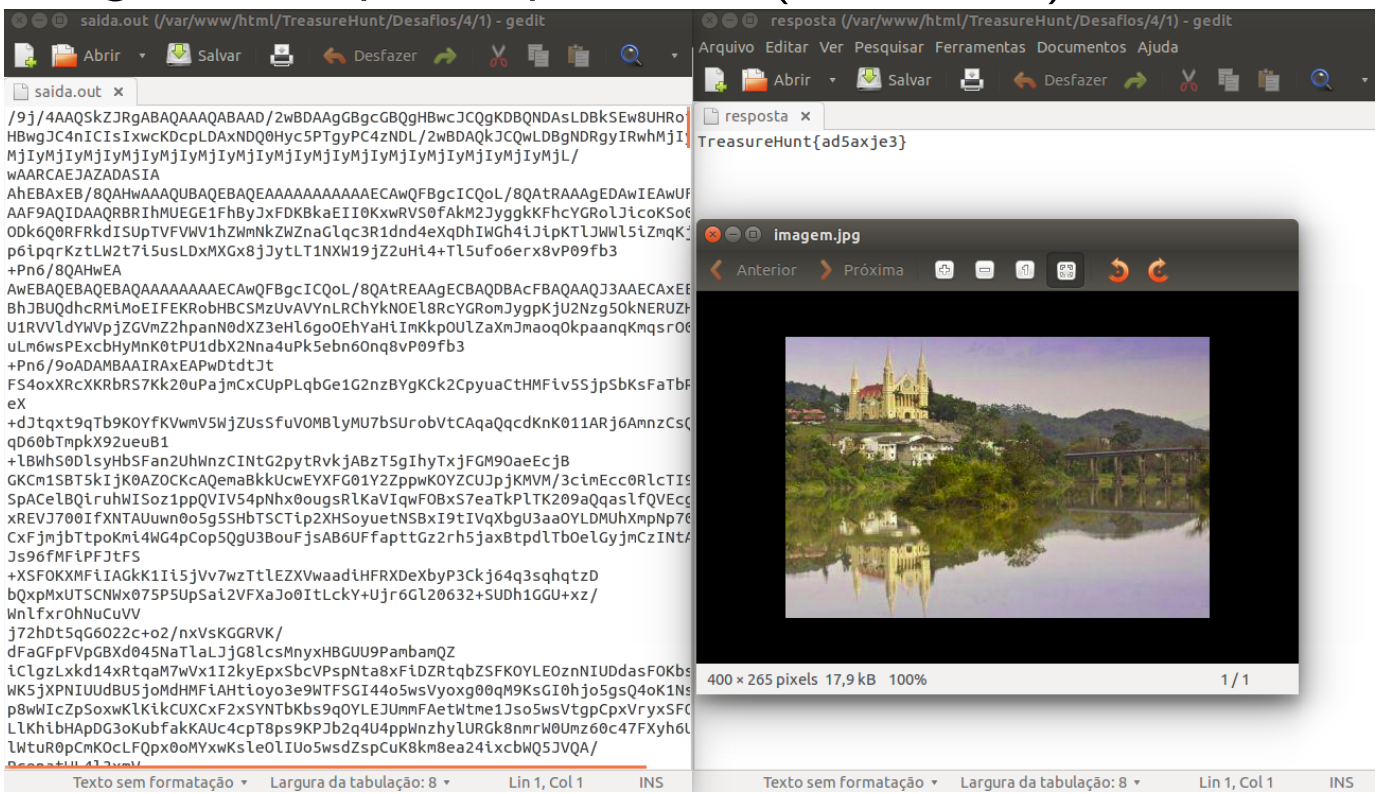
Figure: composite problems (instance 1).



Automatic Challenge Generation

Challenge Generator

Figure: composite problems (instance 2).



Automatic Challenge Generation

Submission System

Figure: response submission interface and individual detailed scorecard. .

Evaluation

Execution

- Competition applied twice, in 3 classes.
  - C1: All students received problems with the same techniques.
  - C2.1 and C2.2: one group received problems with the same techniques as in C1, and the other with different techniques.
- Performance analysis and questionnaire responses on satisfaction with the activity.
- 30 students participated.
- Applied in November 2017.

Evaluation

Results

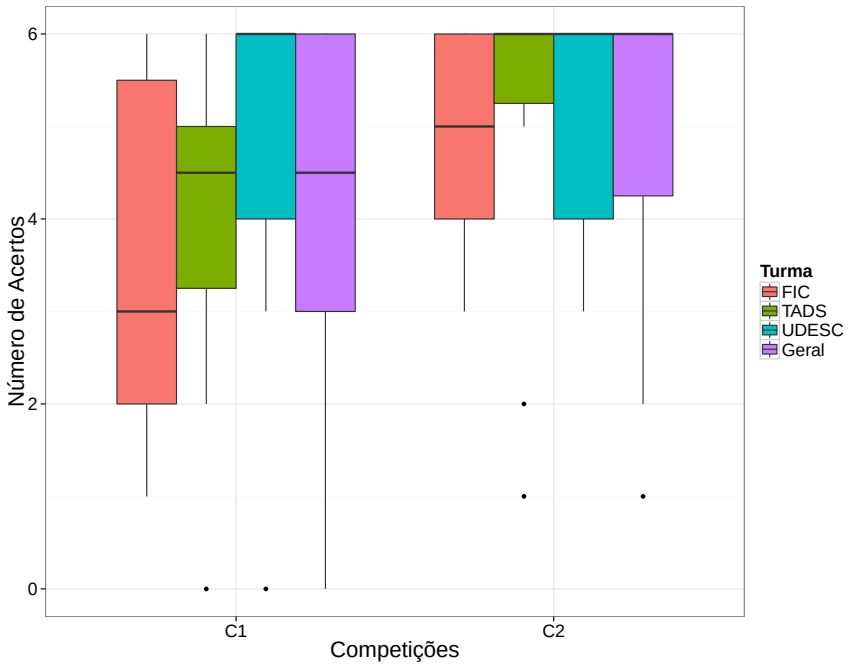


Figura: hits in C1 and C2.

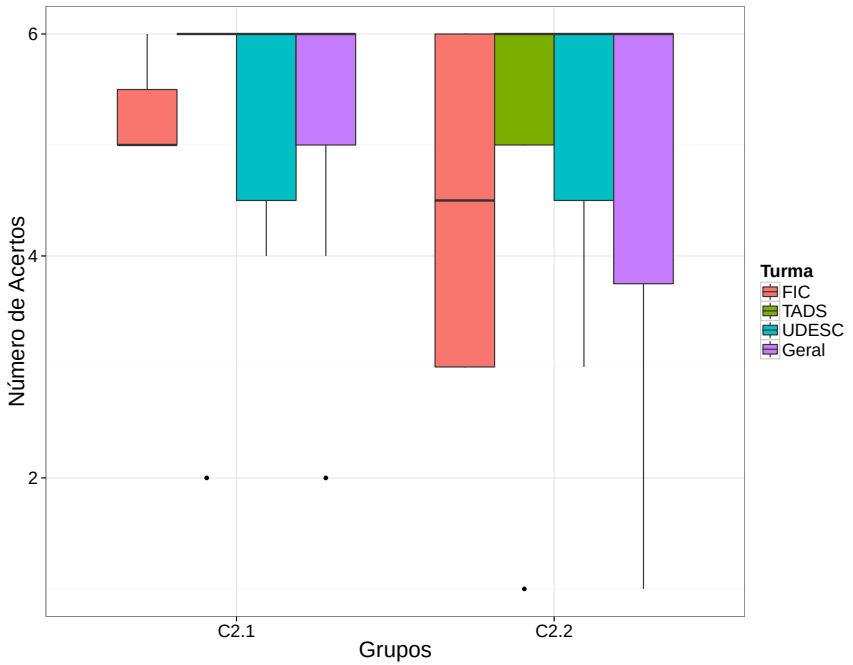


Figura: hits in C2.1 and C2.2.

- Performance with statistically significant difference between C1 and C2.

Evaluation

Questionnaire Results

Tabela: Summary of the questionnaire results.

Question	**Attribute**	**General**		**Evolution**	**Significant?**
		pre	post		
1.1	Satisfaction	4,03	4,52	+0,49	**Yes** ($p = 0{,}0076$)
1.5	Satisfaction	4,77	**4,79**	+0,02	No ($p = 0{,}81$)
1.6	Interest	2,13	2,79	+0,66	**Yes** ($p = 0{,}018$)
1.7	Satisfaction	4,03	4,45	+0,42	**Yes** ($p = 0{,}012$)

Tabela: Questions.

Question number	**Question**
1.1	Games and competitions make me more motivated to learn than traditional classes.
1.5	Practical cybersecurity exercises increase understanding about this area.
1.6	I feel sufficiently prepared (to start) to participate in cybersecurity competitions.
1.7	I think cybersecurity competitions increase the appeal of the area to the general public.

Evaluation

Results and Discussion

- Performance improved from `C1` to `C2`
  - In time
  - In hits
- `C2.1` and `C2.2` with no statistically significant difference.
- Questionnaire results show that the activity was well received by the students.
- There was no response sharing.

Conclusion

- Viability of the prototype of an automatic generation tool of a treasure hunt competition.
  - Equivalent problems
  - Different instances
  - Different classes of problems
  - Composition of techniques
- Randomization efficacy needs to be studied deeply.
- Questionnaires results indicate that the activity was well received, but the sample was considered small.

Conclusion

Future Prospects

- Improve TreasureHunt tool.
- Add ID to competition.
- Expand
  - the amount of techniques;
  - the number of tools per technique;
  - the amount of composition levels;
  - the target audience.
- There are features in development.

Automatic Challenge Generation for Teaching Computer Security

Ricardo de la Rocha Ladeira, Rafael Rodrigues Obelheiro
{ricardo.ladeira@ifc.edu.br, rafael.obelheiro@udesc.br}

4 de outubro de 2018

Appendix

Existing Challenges

- Repository analysis of challenges (CTF, 2017).
- Competitions held between january 2016 and march 2017.
- **Competitions analyzed**: 84
- **Exercises analyzed**: 1250
- **Composite problems**: 86 (6,9%)
- **Problem classes**:
  - Codification/Cryptography
  - Reverse Engineering
  - Forensics
  - Miscellaneous
  - Web
- Most competitions were **non-linear**.
- About **200 techniques found**.

Appendix

Related Work

Table: Comparative between the developed tool and related work.

Work	**Automatic Generation**	**Problems composition**	**Problems uniformity**	**Classes of problems**
PicoCTF (BURKET *et al.*, 2015)	problems	×	✓	▶ Reverse Engineering ▶ *Web* ▶ Miscellaneous ▶ Codification/ Cryptography
MetaCTF (FENG, 2015)	competition	±	×	▶ Reverse Engineering
SecGen (SCHREUDERS et al., 2017)	competition	±	×	▶ *Web* ▶ Forensic ▶ Miscellaneous ▶ Codification/ Cryptography
TreasureHunt	competition	✓	✓	▶ Reverse Engineering ▶ Forensic ▶ Miscellaneous ▶ Codification/ Cryptography

Appendix

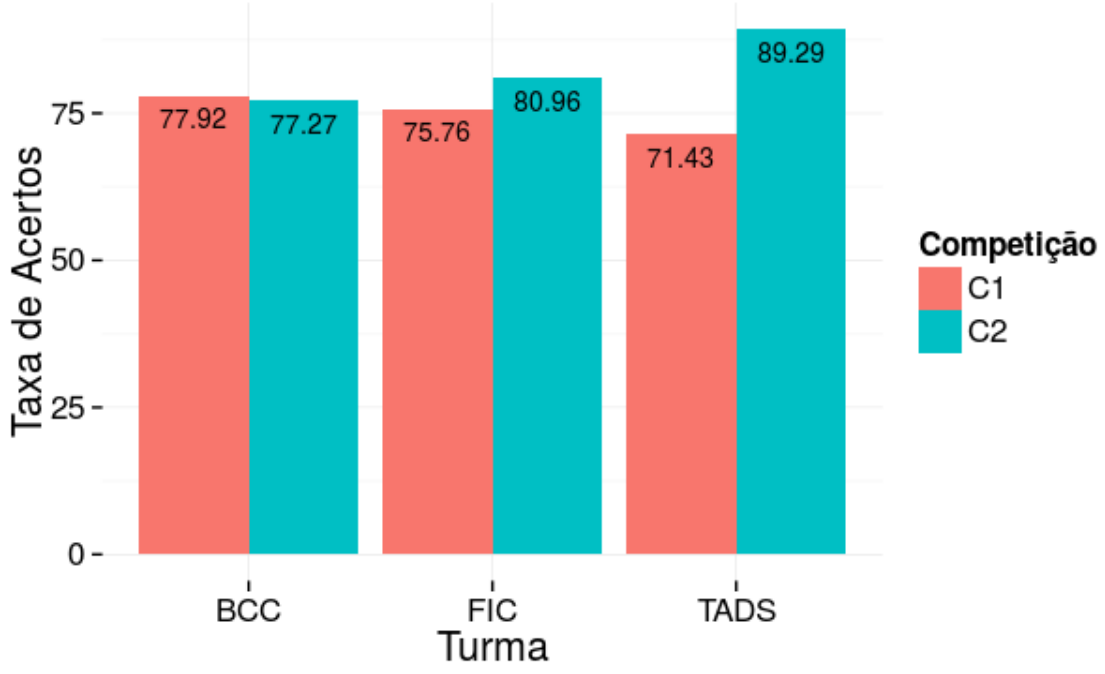
Results of Questionnaires - Method

- Pre-test: 30 answers.
- Post-test: 29 answers.
- Likert scale questions
  - Strongly disagree . . . Totally agree
  - Very demotivating . . . Very motivating
- Consistency assessed by Cronbach's alpha coefficient.
- Statistical difference verified by Wilcoxon test for unpaired samples.

Appendix

Results

Figure: correct submissions rate (in %) per class.

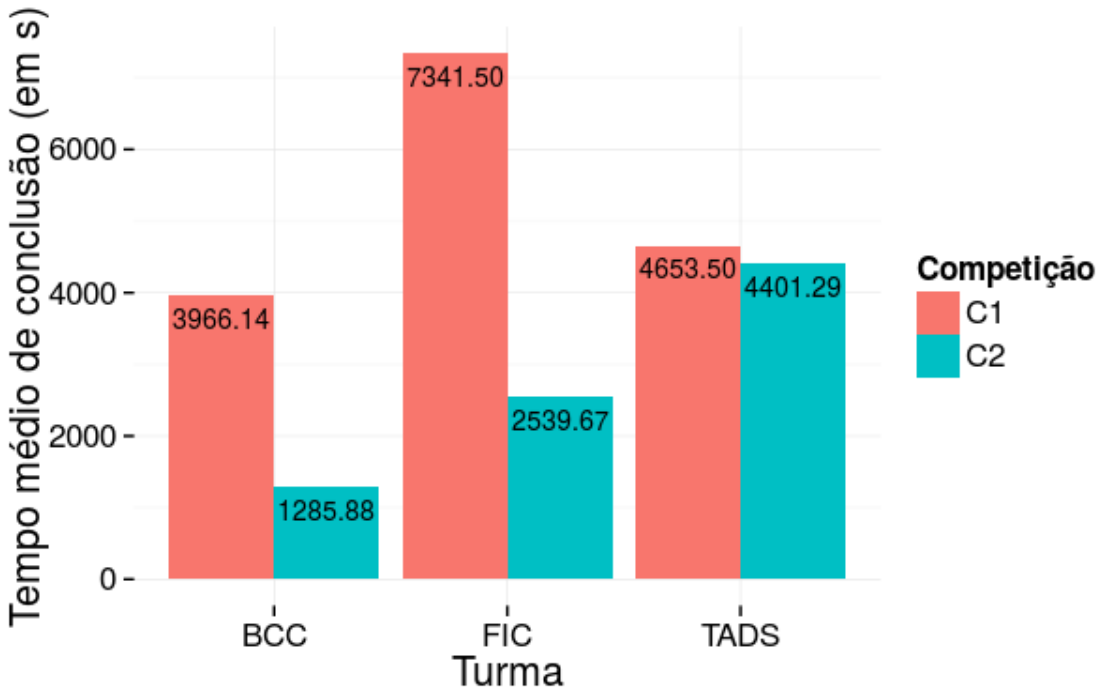


There were no occurrences of response sharing.

Appendix

Results

Figure: average time (in s) to complete the activity.



C1 and C2 are different statistically. It was not possible to say that C2.1 and C2.2 are different.

Appendix

Observation

- Curiosity of students not enrolled;
- Competition factor;
- Relaxation;
- Errors in `outguess` tool.

References

- BURKET, J.; CHAPMAN, P.; BECKER, T.; GANAS, C.; BRUMLEY, D.; Automatic Problem Generation for Capture-the-Flag Competitions. 2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15). ACM, Washington DC, USA. 2015.
- CTF write-ups repository. Disponível em: <https://github.com/ctfs>. Acesso em: 27 fev. 2017.
- FENG, Wu-chang. A scaffolded, metamorphic ctf for reverse engineering. In: 2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15). USENIX Association. 2015.
- SCHREUDERS, Z. C. et al. Security scenario generator (secgen): A framework for generating randomly vulnerable rich-scenario vms for learning computer security and hosting ctf events. In: USENIX ASSOCIATION. USENIX. [S.l.], 2017.

Automatic Challenge Generation for Teaching Computer Security

Ricardo de la Rocha Ladeira, Rafael Rodrigues Obelheiro
{ricardo.ladeira@ifc.edu.br, rafael.obelheiro@udesc.br}

4 de outubro de 2018