User Manual

This document provides a user manual, which will instruct the reader in the installation, running and use of the OEMSR version 1.0.

1 Introduction to the OEMSR

The Optimal Economic Model System for Ransomware (OEMSR) is the final deliverable (4.4) of workplan 4 for the RAMSES project. This system is intended to demonstrate the effects of target selection, ransom pricing and ransom strategies on the profitability and threat-level of a known or hypothetical ransomware strain. This will allow scholars and LEAs to analyse the optimal scenario and identify practices that may help disrupt the optimal path towards more effective and/or damaging strains prior to their emergence.

At time of submission, this system is suitable for identifying whether an intelligence network is required to achieve the optimal profit of a given malware, and whether the malware in question causes significant collateral damage (a large body of infections that the criminals have little intention of ever dealing with).

The scope of the OEMSR is Economic Modelling of Ransomware as a Business. It implements population analysis tools to turn survey data of willingness-to-pay values (the value individuals may associate with their files) into variables that allow the derivation of potential profit. This can be achieved for different ransom values and strategies. The aim of this process is to identify how close a selected strategy and price is to the optimal, and to characterise the traits that make a given strategy 'optimal'. It also allows the comparison of strategies, to determine which combination of strategies and price-points will elicit the largest profit from a given population. Due to the definition of 'optimal' depending on the input population, it is not possible to define a blanket 'optimal' ransomware solution, but this tool allows for the identification of a 'population optimal' allowing for targeted solutions to be derived from OEMSR estimations, to protect at-risk populations based on real survey data.

This tool is a foundation for future development. Presently an Economic Model System, it is possible to extend this model with epidemiological models, trust networks and psychological models to explore other elements of the ransomware optimisation process.

1.1 Developers

The following individuals contributed to the development of the OEMSR version 1.0, listed with their roles and contributions:

- Darren Hurley-Smith
    - Research Associate, University of Kent
    - Deliverable management, Python 3 model implementation, Flask implementation, documentation, validation, and quality assurance
- Edward Cartwright
    - Reader, University of Kent
    - Economic theory, mathematical modelling, validation of the former, survey data
- Julio Hernandez-Castro
    - Professor, University of Kent
    - Cybersecurity context for applied economic theory, validation of the OEMSR
- Anna Stepanova
    - Lecturer, University of Kent
    - Economic theory and survey data

2 Start-up, Dependencies and First-Run Guide

This section will guide you through the download, unpacking, dependencies installation and first-run processes of the OEMSR. All examples are for a Windows 10 machine and should also work on Windows 7.

2.1 Directory Set-Up

The zip archive containing the OEMSR should be unzipped to a location of your choice. To simplify this, these examples assume that you have chosen to unpack the files to desktop, in an appropriately named folder.

Viewing the directory on your desktop, you will find 6 python files and 4 sub-directories. The first of these sub-directories '__pycache__' can be ignored. 'Static' contains the ccs stylesheets, fonts, images, JavaScript and results sub-directories. No changes should be made to any of these directories, as the first 4 directories contain content vital for the front-end presentation and function. The results sub-directory contains image and text file outputs associated with any modelling performed using the OEMSR system, as a permanent record of one's analyses.

Returning to the application top-most directory, 'Templates' contains a single '__layout' html file. This may be changed to alter the appearance of the front-end but is not advised for inexperienced users.

'Tutorials' contains a series of video guides to using the tool, intended to supplement this user manual.

The 6 python files form the Flask presentation module and back-end logic of the OEMSR. These should not be altered unless modification by experienced Python 3 developers is intended. It is advised that you back up the OEMSR directory if you engage in development activities with the source code.

2.2 Installing Dependencies

As a Python 3 program, the OEMSR requires that you have Python 3.x installed. It also requires the following modules, installed using pip:

- Flask
- Wtforms
- Matplotlib
- Scipy
- Numpy

These can all be installed using the following instruction from commandline (assuming pip and Python 3 are installed):

```
> py -3 -m pip install flask wtforms matplotlib scipy numpy
```

Once installed, you are ready for your first run of the OEMSR.

2.3 First Run

To run the OEMSR, please ensure you have command prompt running at all times and use an appropriate browser. We recommend Firefox or Chrome, both of which have been tested with our front-end implementation.

To start the localhost:5000 server, type the following into command prompt and hit enter:

```
> py -3 model_front.py
```

This will elicit the response:

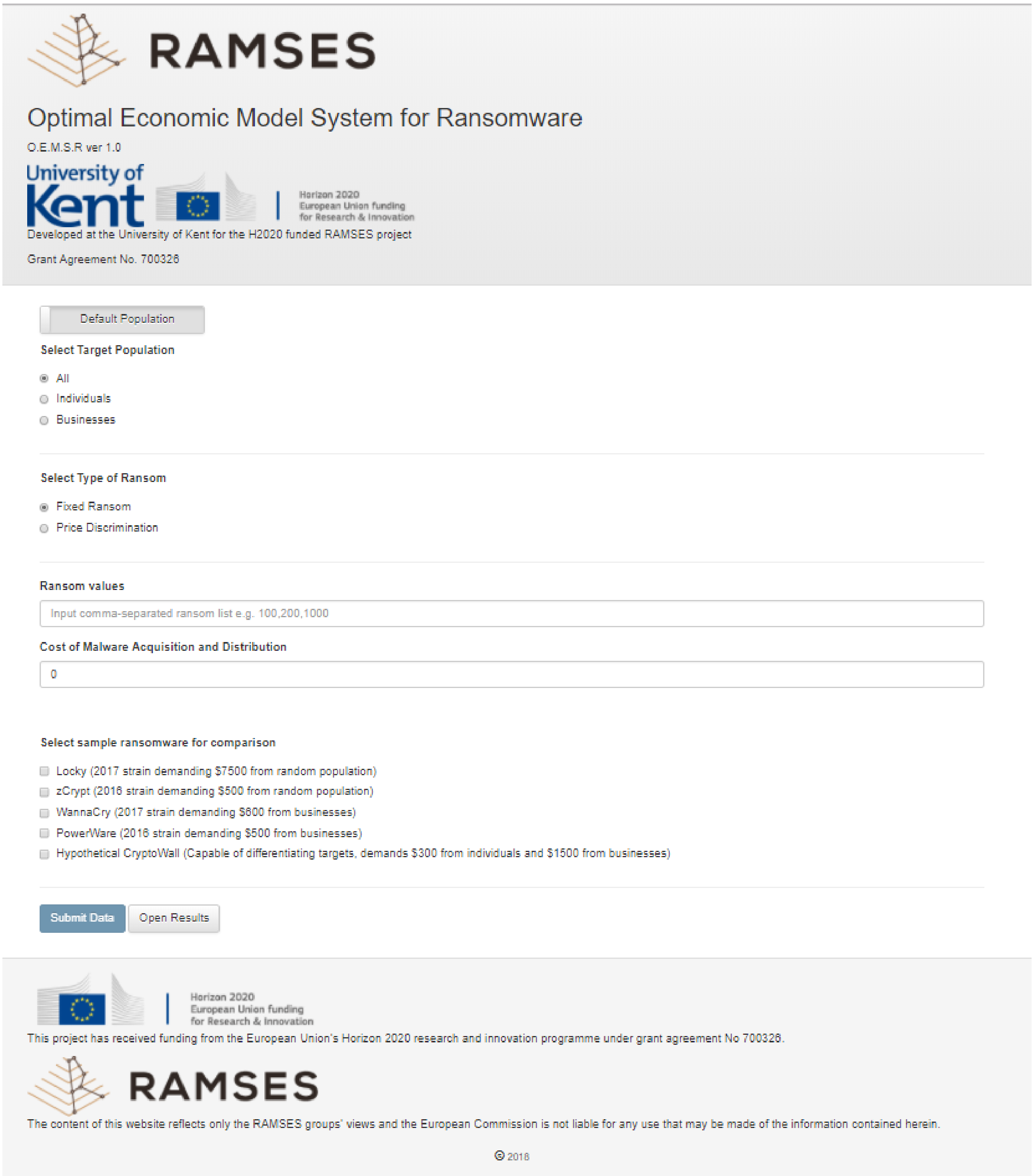
```
* Restarting with stat
```

```
 * Debugger is active!
* Debugger PIN: 183-445-163
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

This means that your OEMSR server is running. Go to your selected browser and navigate to localhost:5000. This will take you to the OEMSR homepage, the default selections page for ransomware and population definitions. You are now ready to define and analyse your first malware example.

3 Using the OEMSR

Figure 1 shows the default view that you will encounter when first starting your OEMSR client.



RAMSES

Optimal Economic Model System for Ransomware

O.E.M.S.R ver 1.0

University of Kent

Horizon 2020
European Union funding
for Research & Innovation

Developed at the University of Kent for the H2020 funded RAMSES project

Grant Agreement No. 700326

Default Population

Select Target Population

● All
○ Individuals
○ Businesses

Select Type of Ransom

● Fixed Ransom
○ Price Discrimination

Ransom values

Input comma-separated ransom list e.g. 100,200,1000

Cost of Malware Acquisition and Distribution

0

Select sample ransomware for comparison

□ Locky (2017 strain demanding $7500 from random population)
□ zCrypt (2016 strain demanding $500 from random population)
□ WannaCry (2017 strain demanding $600 from businesses)
□ PowerWare (2016 strain demanding $500 from businesses)
□ Hypothetical CryptoWall (Capable of differentiating targets, demands $300 from individuals and $1500 from businesses)

Submit Data   Open Results

Horizon 2020
European Union funding
for Research & Innovation

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700326.

RAMSES

The content of this website reflects only the RAMSES groups' views and the European Commission is not liable for any use that may be made of the information contained herein.

© 2018

Figure 1 – Default View

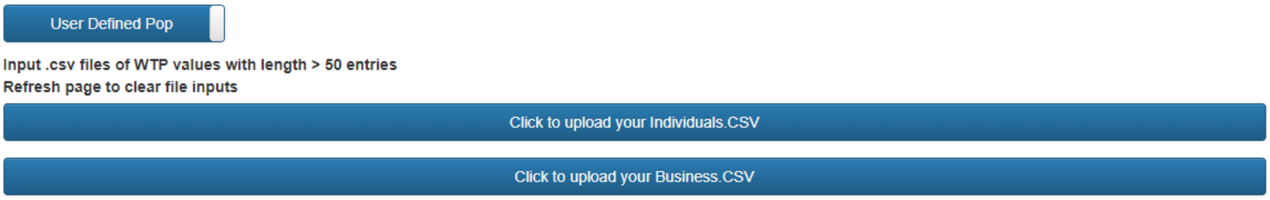
This is a simple web-page, with a small selection of smart features that allow the hiding of irrelevant fields for both input integrity and ease of viewing. There are several easily identified key elements to this, simplest, view.

3.1 Population Definition

The target population is the foundation of this system. Without a target population, the system would not be able to run at all, as it is the individual willingness-to-pay (WTP) of members of a population that allow the profitability of a given malware to be determined and the optimal to be found.

Figure 1 shows the population definition field in its default state, using the system-defined default population. This population is generated from a weighted list of WTP for individuals and businesses, with 1000 individuals and 100 businesses generated for a total population of 1100 for the model population. This is a suitable number as it allows for an acceptable degree of distribution of WTP values throughout the population, whilst not bogging down the model in lengthy generation and attribute derivation computations. Sub-types of individuals or business are not required: the willingness-to-pay of a given individual or business can be used as an indicator of their capacity to pay (which may in turn reflect personal or institutional wealth). Business default WTP values are always set higher than individual WTP, and the distribution of these WTP values can be found in the 'ransom_calc.py' module.

Figure 2 shows the alternative mode of this button.


User Defined Pop
Click to upload your Individuals.CSV
Click to upload your Business.CSV

Input .csv files of WTP values with length > 50 entries
Refresh page to clear file inputs

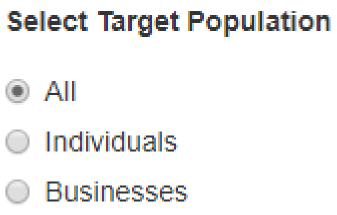
Figure 2 – User Defined Population Mode

User defined populations may be added from CSV files. Each population type (individual or business) should be input as its own, column format, csv file (each WTP on its own column). The only variables required in each CSV are the WTP values of the surveyed or hypothetical population. The resulting population will have a size equal to the number of WTP values: the population is purely defined by the CSV inputs in these cases. If you input one CSV, but leave the other blank, the default will be used for the uninitialized population type. To clear your file selections, refresh the page.

It must be stressed this this population is considered infected with the target malware unless the appropriate defence calculations say otherwise later in the OEMSR process. This is because the OEMSR doesn't implement an epidemiological model at present and must assume that the population targeted has reached the stage at which money will, or will not, change hands. An epidemiological model is slated for future work.

3.2 Population Types

Having selected a target population, you must now determine the way in which your ransomware is intended to interact with it. You can select whether you will target businesses, individuals, or the entire population, as shown in Figure 3.



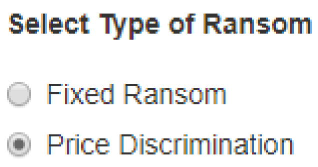
Select Target Population

● All
○ Individuals
○ Businesses

Figure 3 – Target Selection

3.3 Type of Ransom

After selecting how your hypothetical malware will select the target population, you must select the type of ransom. Fixed, the default setting, is by far the most common type of ransom type in real malware. It is simple, requires no specific intelligence and aims to demand a set prices, sometimes with multipliers based on swiftness of payment (with increases for overdue payment). Also shown in Figure 4 is the price discrimination option. This is a more sophisticated type of ransom; wherein sub-strategies can be defined. These strategies detail how multiple ransom values are attributed to the population.



Select Type of Ransom

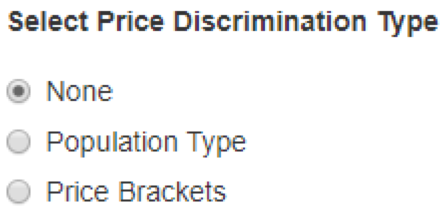
○ Fixed Ransom

● Price Discrimination

Figure 4 – Ransom Type Selection

3.4 Type of Discrimination

Figure 5 shows the sub-types of discriminatory ransoms. This field will not show unless price discrimination has been selected previously. The options include non, population type, and price brackets.



Select Price Discrimination Type

● None

○ Population Type

○ Price Brackets

Figure 5 – Discrimination Type Selection

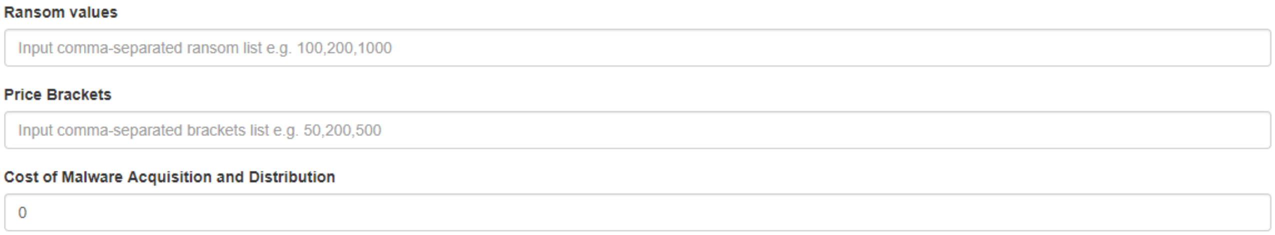
None will randomly assign different ransoms (defined by the user) throughout the target population. This represents poor or no intelligence gathering, and a scatter-shot approach to seeing if multiple ransom values can lead to higher profit. Up to four ransom values can be defined.

Discrimination by population type will allow the definition of two ransoms; the first being presented to the individual population, and the second to the business population.

Price brackets allows the definition of up to four ransoms and four price brackets. Brackets represent an estimation of WTP. For example, brackets (100,500,1000) will sort the population into WTP values of 0-100,101-500, and 501-1000. It is, therefore, not advised to have ransoms set higher than your bracket value for each bracket, as it is unlikely that members of that bracket will be willing to pay.

3.5 Ransom, Brackets and Cost inputs

Figure 6 shows the fully expanded field for ransom, bracket and cost inputs. Price brackets is hidden unless price brackets have been selected in the previous options.



Ransom values

Input comma-separated ransom list e.g. 100,200,1000

Price Brackets

Input comma-separated brackets list e.g. 50,200,500

Cost of Malware Acquisition and Distribution

0

Figure 6 – Expanded Ransom, Price Brackets, and Costs Inputs

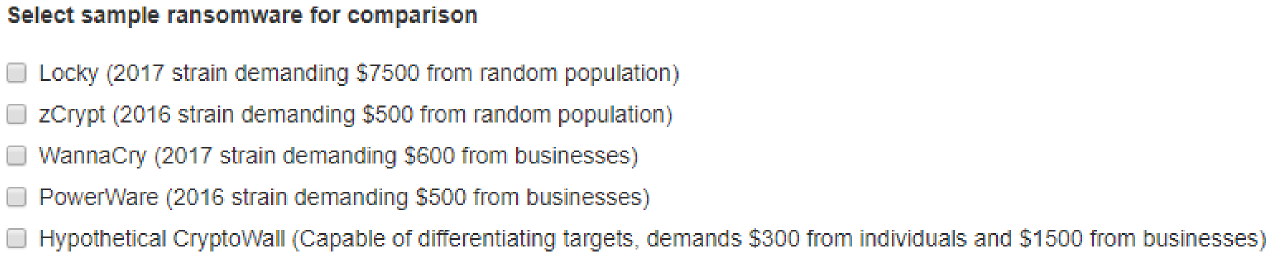
Ransom values are input and comma-separated integers or floats. For example, a fixed ransom may be 100, a discriminatory ransom (no strategy) might be 100,1000,2000, and a discriminatory ransom (pop type) might be 100,1000. If you put in too many values for the given settings, it will select values from the leftmost entry, until it has sufficient values. Attempts to use more than 4 ransoms will cause an overflow in the graphing portion of the model, do not attempt this at this time (InternalError:500 will be thrown).

Price brackets are entered in much the same way, you must have as many brackets as you have ransoms.

The cost of malware acquisition and distribution is a user defined floating point or integer value. It is a single value, equal to the estimated foxed costs of the malware. This is a purely user defined variable but can be expanded in future work to include elements from ransomware epidemiology, Ransomware as a Service (RaaS) data, and other additional modules that define the Optimal Model beyond Economic and Business considerations.

3.6 Example Malware Selection

Figure 7 shows the final selection the user may make before submitting their hypothetical ransomware to the OEMSR. The user may select one or all of the options shown below.



Select sample ransomware for comparison

□ Locky (2017 strain demanding $7500 from random population)

□ zCrypt (2016 strain demanding $500 from random population)

□ WannaCry (2017 strain demanding $600 from businesses)

□ PowerWare (2016 strain demanding $500 from businesses)

□ Hypothetical CryptoWall (Capable of differentiating targets, demands $300 from individuals and $1500 from businesses)

Figure 7 – Example Ransomware Selection

The first four are examples of contemporary malware, from within the last two years (at time of submission). These are based on real data and abstractions of their attributes. Fixed costs are estimated, as real data on the cost of development is extremely difficult to isolate. Further analysis of RaaS will be required to make accurate future estimations of price.

The last example is a hypothetical Crypto Wall implementation. We assume that the developers iterate on this crypto-ransomware, to include target profiling capabilities. We have included this option due to a lack of effective examples in the current literature: ransomware operators still largely send fixed ransoms, and when discriminating, discriminate by only targeting a certain portion of the population instead of setting tiered ransom prices based on assumptions regarding their infected population.

3.7 Results

Having completed all prior steps, the results are obtained by clicking Submit. In Figure 8, the submit button is greyed out because not all required fields have been filled. Filling these fields will allow the button to be pressed, which is indicated by the button returning to a more vibrant shade of blue.

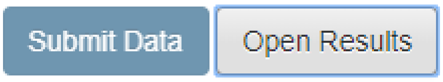


Figure 8 – Submit and Results Buttons

Open results may be clicked regardless of whether the OEMSR has been run. However, the default display, shown in Figure 9, will be shown if there are no results to display. The last results generated will be shown, if the button is pressed after closing the modal view but before the next OEMSR process is run.

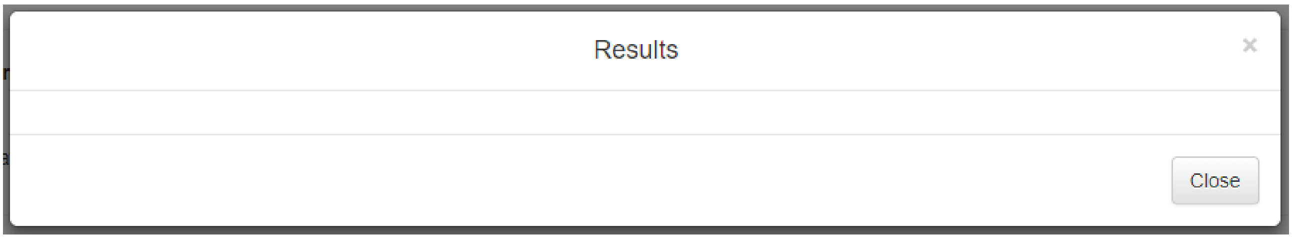
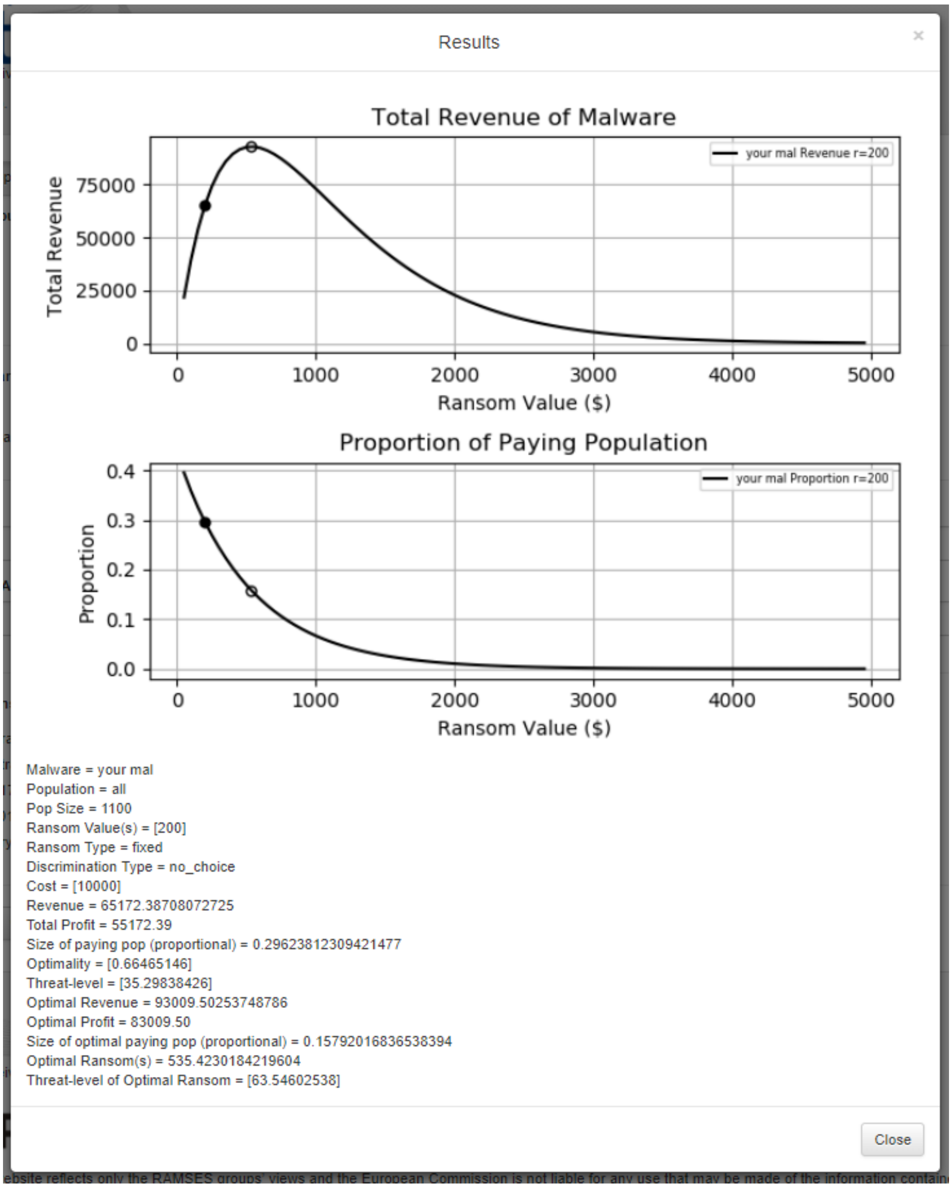


Figure 9 – Default Results Modal

Figure 10 shows a simple results output. This example runs just the user defined malware example over the entire default population. A fixed ransom of $100 is demanded, with $10,000 in fixed costs, and no examples are selected to compare against.


Results
×
Total Revenue of Malware
your mal Revenue r=200
Total Revenue
75000
50000
25000
0
0
1000
2000
3000
4000
5000
Ransom Value ($)
Proportion of Paying Population
your mal Proportion r=200
Proportion
0.4
0.3
0.2
0.1
0.0
0
1000
2000
3000
4000
5000
Ransom Value ($)
Close
ebsite reflects only the RAMSES groups' views and the European Commission is not liable for any use that may be made of the information contain

```
Malware = your mal
Population = all
Pop Size = 1100
Ransom Value(s) = [200]
Ransom Type = fixed
Discrimination Type = no_choice
Cost = [10000]
Revenue = 65172.38708072725
Total Profit = 55172.39
Size of paying pop (proportional) = 0.29623812309421477
Optimality = [0.66465146]
Threat-level = [35.29838426]
Optimal Revenue = 93009.50253748786
Optimal Profit = 83009.50
Size of optimal paying pop (proportional) = 0.15792016836538394
Optimal Ransom(s) = 535.4230184219604
Threat-level of Optimal Ransom = [63.54602538]
```

Figure 10 – Simple Results Modal (Default pop, All, Fixed, $100 ransom, $10,000 costs, no examples)

The graph outputs the continuity of ransoms from $50 to $4950, demonstrating the revenue generated and the proportion of the population that pays to generate that revenue. The filled marker represents the defined ransom results, the hollow marker represents the optimal. This is true of all lines; their respective filled and hollow markers will share the line colour.

A text output of the results, including variables not able to be depicted in the graphs, is provided. Importantly, this includes the profitability (revenue – costs), optimal profitability, and threat-level of the malware in question. Profitability is a good measure of attractiveness for a given malware; it represents the cash-in-hand outcome of the ransomware phase (before any costs associated with extraction of value to fiat currency via cryptocurrencies). The threat-level is a simple indicator of the amount of damage that a given ransomware might do, considering the proportion of people who pay, the population size and the total profit.

Threat is calculated as follows:

$$T = (i/n)*(1-p_q)$$

Variable *i* represents the total profit, *n* is the total population size affected/targeted by the malware, and p_q represents the proportion of individuals who end up paying (thus contributing to *i*). This score is intended to represent the threat of the malware from a collateral damage and likelihood of repetition perspective. Malware that is attractively profitable, requires a small proportion of the population to pay, and therefore can ignore a substantial portion of those they infect (leading to loss of files and disruption of business unless WTP rises to meet high ransom demands) is more of a threat than the alternative.

It should be noted that the optimal malware may be less threatening than a sub-optimal variant. This is because the optimality of a malware implementation is determined by its profit, not the damage it causes (which is a symptom of generating duress, not the intention). As a result, sub-optimal behaviours may cause malware to be more damaging, and the OEMSR allows users to investigate this further, by identifying where optimality is best served by being less damaging, and how the most threatening strains arise.

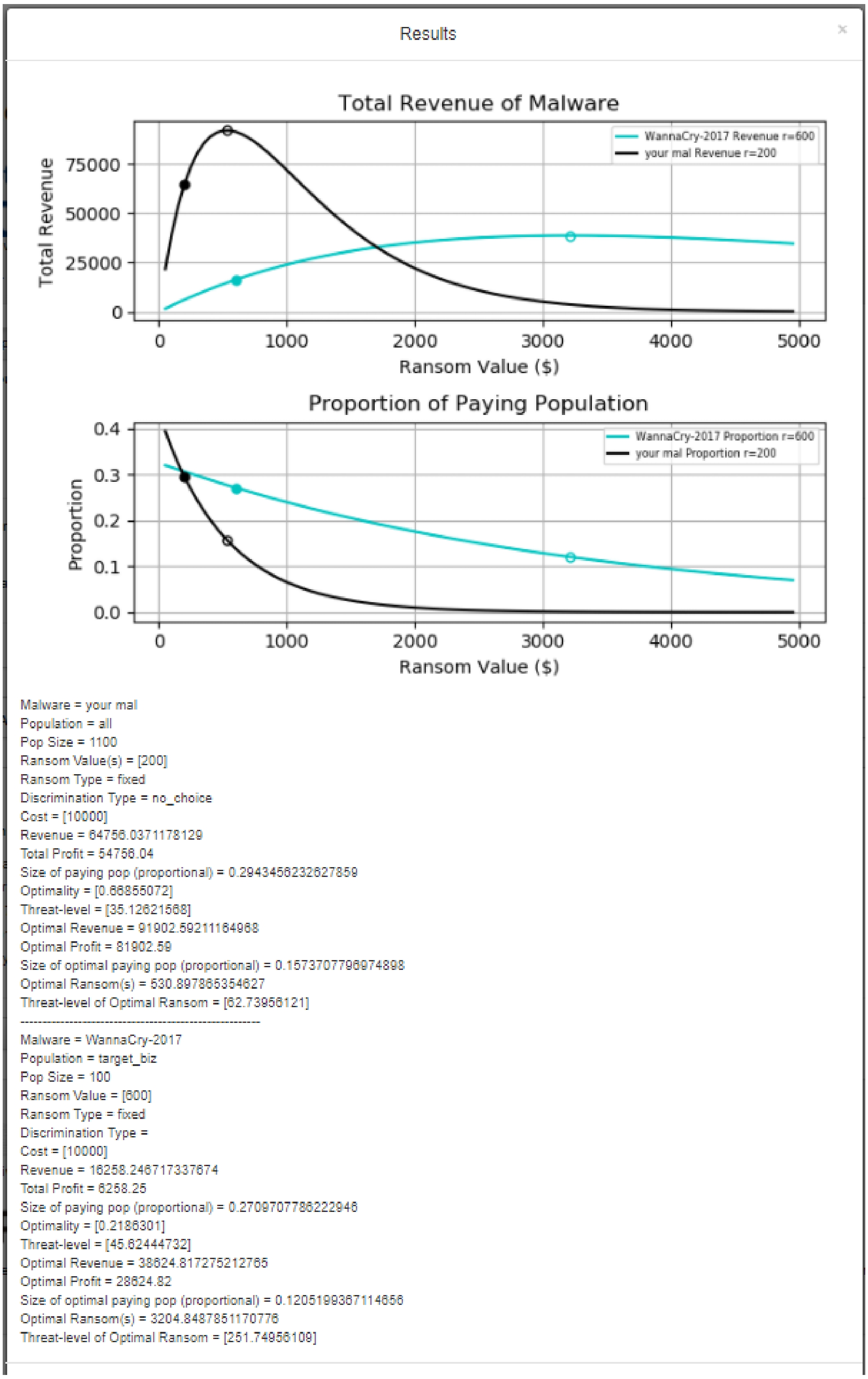
A general trend noted during testing was that optimal implementations generally had threat levels amongst the highest possible for their population and strategy combinations, but higher scores were possible if the decline in profit with sub-optimal paying proportions was sufficiently slow (as the larger population unable/unwilling to pay the ransom would begin to increase the threat faster than the profitability drops off).

Figure 11 (next page) shows the same user defined malware, compared against the WannaCry example. The two different population selection methods are easily seen, as the business focused WannaCry example exhibits a significantly different distribution of profitability and proportion of population willing to pay. The text output captures results from all the modelled malware, showing both the user defined and example malware.

Figure 12 (page after next) shows the output of a user defined price bracketed malware, and the hypothetical CryptoWall implementation that targets each population type with a different ransom. It must be noted that the line style changes with bracketed ransoms, to indicate which line is associated with each subdivision of the population (be it by bracket, randomly or by type). As previously shown, the text output is reproduced for all modelled malware strains. Where proportion, ransom and revenue differ by the population segment associated with a given form of discrimination, square brackets are used to enclose a list of the variables, to provide a breakdown of elements that go on to contribute towards the threat-level and profitability of a malware strain.

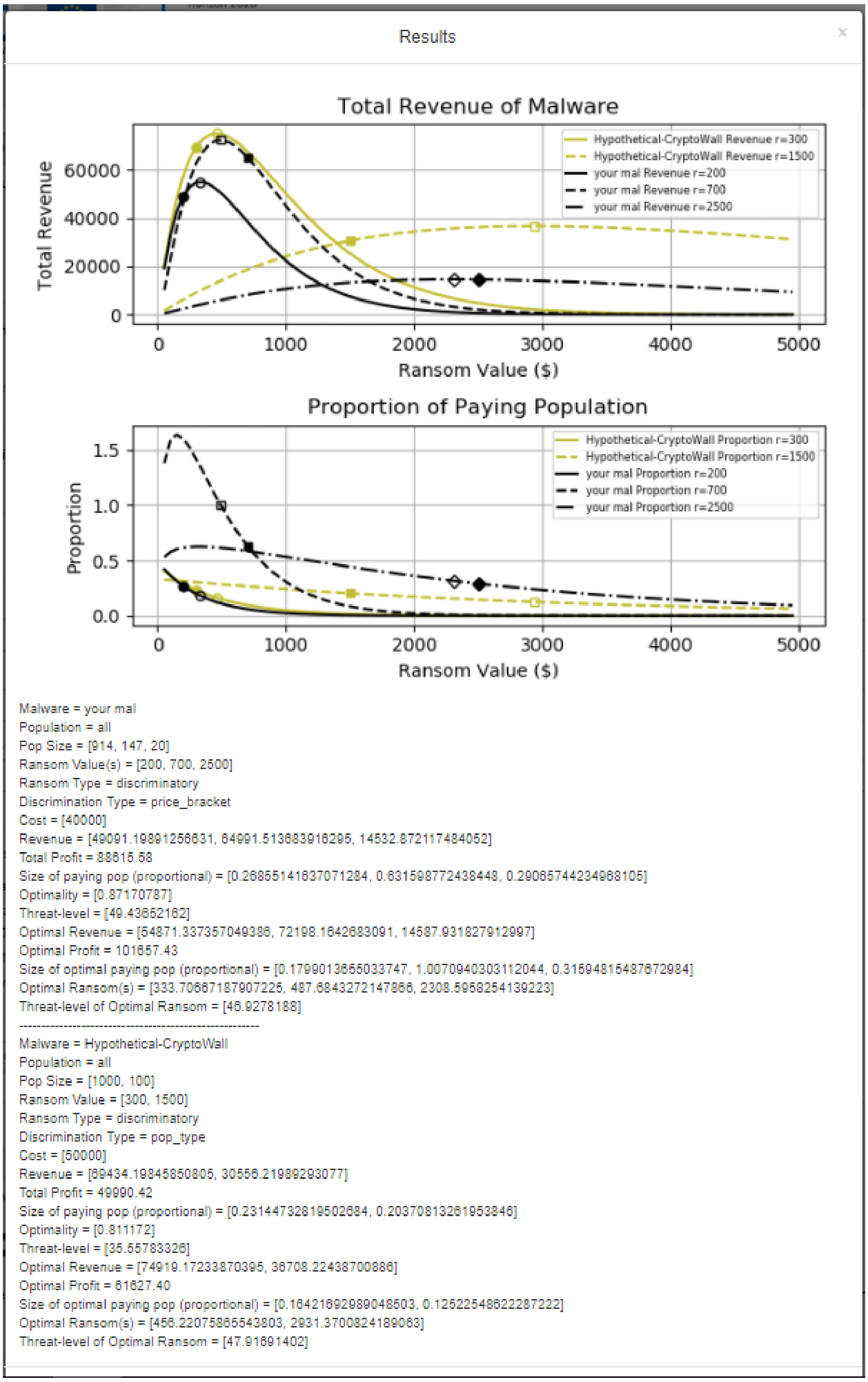
The calculation of the paying proportion is very sensitive to local minima in the population definition process of OEMSR. This results in a line that may exceed 1.0 times the population for some extremely dense WTP distributions, but it is not possible for the optimal ransom to ever exceed 1.0 for this value. Further study is required to isolate and smooth these local minima.

All results are saved to the */static/results/graphs* and */static/results/reports* directories for graphs and text reports respectively. All files are saved in the format *YYYY-MM-DD_UUID.png* (with reports taking extension *txt* instead). The UUID is tied to the report and graph, allowing outputs to be grouped long after the OEMSR session has been closed. These results are saved on every run of the OEMSR, so periodic cleaning of these directories may be required.



```
Malware = your mal
Population = all
Pop Size = 1100
Ransom Value(s) = [200]
Ransom Type = fixed
Discrimination Type = no_choice
Cost = [10000]
Revenue = 64756.0371178129
Total Profit = 54756.04
Size of paying pop (proportional) = 0.2943456232627859
Optimality = [0.66855072]
Threat-level = [35.12621568]
Optimal Revenue = 91902.59211164968
Optimal Profit = 81902.59
Size of optimal paying pop (proportional) = 0.1573707796974898
Optimal Ransom(s) = 530.897865354627
Threat-level of Optimal Ransom = [62.73956121]
----------------------------------------------------------------
Malware = WannaCry-2017
Population = target_biz
Pop Size = 100
Ransom Value = [600]
Ransom Type = fixed
Discrimination Type =
Cost = [10000]
Revenue = 16258.246717337674
Total Profit = 6258.25
Size of paying pop (proportional) = 0.2709707786222946
Optimality = [0.2186301]
Threat-level = [45.62444732]
Optimal Revenue = 38624.817275212765
Optimal Profit = 28624.82
Size of optimal paying pop (proportional) = 0.1205199387114656
Optimal Ransom(s) = 3204.8487851170776
Threat-level of Optimal Ransom = [251.74956109]
```

Figure 11 – Simple example compared with Wannacry (Default pop, Businesses, Fixed, $600, $10,000 costs)


Results
×
Total Revenue of Malware
Total Revenue
60000
40000
20000
0
0
1000
2000
3000
4000
5000
Ransom Value ($)
Hypothetical-CryptoWall Revenue r=300
Hypothetical-CryptoWall Revenue r=1500
your mal Revenue r=200
your mal Revenue r=700
your mal Revenue r=2500
Proportion of Paying Population
Proportion
1.5
1.0
0.5
0.0
0
1000
2000
3000
4000
5000
Ransom Value ($)
Hypothetical-CryptoWall Proportion r=300
Hypothetical-CryptoWall Proportion r=1500
your mal Proportion r=200
your mal Proportion r=700
your mal Proportion r=2500

```
Malware = your mal
Population = all
Pop Size = [914, 147, 20]
Ransom Value(s) = [200, 700, 2500]
Ransom Type = discriminatory
Discrimination Type = price_bracket
Cost = [40000]
Revenue = [49091.19891256631, 64991.513683916295, 14532.872117484052]
Total Profit = 88615.58
Size of paying pop (proportional) = [0.26855141637071284, 0.631598772438448, 0.29065744234968105]
Optimality = [0.87170787]
Threat-level = [49.43652162]
Optimal Revenue = [54871.337357049386, 72198.1642683091, 14587.931827912997]
Optimal Profit = 101657.43
Size of optimal paying pop (proportional) = [0.1799013655033747, 1.0070940303112044, 0.31594815487672984]
Optimal Ransom(s) = [333.70667187907225, 487.6843272147866, 2308.5958254139223]
Threat-level of Optimal Ransom = [46.9278188]
------------------------------------------------------------
Malware = Hypothetical-CryptoWall
Population = all
Pop Size = [1000, 100]
Ransom Value = [300, 1500]
Ransom Type = discriminatory
Discrimination Type = pop_type
Cost = [50000]
Revenue = [69434.19845850805, 30556.21989293077]
Total Profit = 49990.42
Size of paying pop (proportional) = [0.23144732819502684, 0.20370813261953846]
Optimality = [0.811172]
Threat-level = [35.55783326]
Optimal Revenue = [74919.17233870395, 36708.22438700886]
Optimal Profit = 61627.40
Size of optimal paying pop (proportional) = [0.16421692989048503, 0.12522548622287222]
Optimal Ransom(s) = [456.22075865543803, 2931.3700824189063]
Threat-level of Optimal Ransom = [47.91691402]
```

Figure 12 – Complex Results with User Defined Brackets-based scheme compared with Hypothetical Cryptowall Example with Target Differentiation Capabilities (Individuals and Businesses)

4 Summary

This section provides usage instructions, explanations of key processed and output definitions. The content has been aimed at users of low to moderate technical proficiency. More advanced users should refer to the functional definition and technical breakdown, which will provide more detailed information regarding the theoretical and programmatic aspects of the OEMSR.