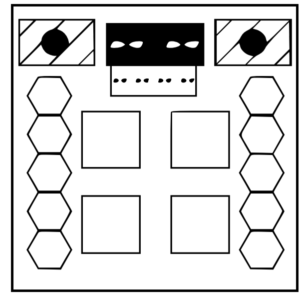
On the Subject of OmegaDestroyer



Not cruel, just a sequel.

This module consists of four pressable displays, two status lights, a two-character display above a four-digit timer, and ten hexagonal buttons numbered 0-9.

If the LEDs begin as colored, or the pressable displays are replaced by a flat, non-pressable cube, you are looking at a different module.

An 8 digit password, using only the numbers 0-9, is required to be entered. Extreme measures have been taken to secure this password.

To block **all** unauthorized access, the password changes **every 2 seconds.**

When the module is initialized or fully reset, the timer will show 210 (not 0). **Buttons cannot be pressed when the timer is 210.** The pressable "main" displays will show R_V, the module's raw value, except the digits will be scrambled.

1: Basic Functionality

The timer, referred to by the value t, will increment by 1 every 2 seconds, which will change the raw value. **However, as soon as the timer changes from 1899 to 1900, the module will fully reset. This takes slightly less than 1 hour.**

The bottom-left main display can be pressed to toggle the timer being "split", which means it and the main displays won't visibly update. **Note that the module will not stop updating, even though it will appear to do so.** In addition, the timer will show in blue while it is split, and all shown values will update to their current states immediately upon becoming "unsplit".

The ten hexagonal buttons will be labeled with digits 0 to 9, and they can be pressed to input their corresponding digit. Input digits are colored magenta, and they don't display/calculate in the wrong order. Digits that have yet to be input, or sometimes** all eight digits of the input, display as "-".

To clear all input and show the digits of R_V again, press the bottom-right main display when its backlight is not yellow/red. **To submit the currently input password, press the top-right main display.** Submitting the correct password **(calculated when submit is pressed)** will solve the module, but strikes will occur when inputting too many digits or submitting a wrong password.

Strikes clear user input and briefly pause the module, but they don't cause it to fully reset unless MWYTH (the module's hard mode) is active (see Appendix MWYTH).

The top-left main display can be tapped to mute or unmute the module. Whenever the module is muted, TL will have a cyan backlight. **Some sounds will still play when the module is muted, such as the sounds for solving and striking.**

TL can also be held for 3 to 10 seconds to give BR a yellow backlight for 1 second. Pressing BR in this state will fully reset the module, which may be useful in the event a new code is desired immediately.

DANGER: DO NOT HOLD THE TOP-LEFT MAIN DISPLAY FOR OVER TEN SECONDS. This will give BR a red backlight for 1 second. Do not press BR when it has a red backlight unless you are prepared, as doing that activates MWYTH.

2: Quadratic Deciphering

The value of R_v is controlled by the following quadratic equation:

$$R_v = (\alpha t^2 + \beta t + \omega) \bmod 100{,}000{,}000$$

α is between 1 and 9,999 inclusive.

β is between 11,111 and 8,888,888 inclusive.

In addition, β has no 9s or non-leading 0s as digits.

ω is between 10,000,000 and 99,999,999 inclusive. **Nothing else is guaranteed.**

Finally, the KYBER*, when taken modulo 16, controls the order of R_v's digits on the main displays as follows:

- The **nth** main display in reading order always displays the **nth** pair of digits (including leading 0s to make it 8 digits) of R_v, but it displays the pair in reverse order (e.g. 1s then 10s places for BR instead of 10s then 1s places) if and only if the **nth** most significant bit of the KYBER modulo 16 (converted to binary, including leading 0s to make it 4 bits) is a 1.

The KYBER, abbreviated as $\underline{k}$, is equal to 16 times the very top display's number, plus a hidden random number between 1 and 14 inclusive. (The hidden number is the only part controlling R_v's digit ordering, as the other part modulo 16 is trivially zero.) **Both parts of the KYBER re-randomize every 210 steps of $\underline{t}$, which is every 7 minutes. A grace period where the previous $\underline{k}$ is used for submission calculations exists only in Twitch Plays, lasting 10 steps of $\underline{t}$ (20 seconds).**

** Key You'd Best Extract Reasonably*

- **The KYBER absolutely does not scramble anything that is not R_v.**
- The number buttons' backlights count down time until re-randomization by changing from green to yellow to red. **The KYBER resets as soon as the top two buttons turn red, so when the bottom two turn red, you need to be close.**
- A sound will play whenever the KYBER re-randomizes.

Due to patterns in the displays caused by this switching, and general patterns in possible sequences of R_v, **the values of $\underline{\alpha}$, $\underline{\beta}$, $\underline{\omega}$, and $\underline{k}$ can all be calculated unambiguously given enough data and effort.** If you are having difficulty with this, see Appendix OM-G-D for tips.

Once you have determined $\underline{\alpha}$, $\underline{\beta}$, and $\underline{\omega}$, prepare the following section. When you get a $\underline{k}$ you think will last long enough to finish the module, do the calculations.

3: Security Enciphering

First, split $\underline{k}$ into its prime factors, along with a single 1. Take the last digit of the sum of these factors, and call it $\underline{x}$. **The last digit of the submitted time must match $\underline{x}$, or the module will strike upon submitting.**

- Example A: 28 splits to 1*2*2*7, and 1+2+2+7=12. The last digit is 2, so x=2.
- Example B: 13 is prime, so it's 1*13, and 1+13=14. The last digit is 4, so x=4.
- Example C: 1 has no prime factors, so it's just 1. The last digit is 1, so x=1.

Next, encrypt the submitted R_v (including leading 0s to make it 8 digits) using the following three ciphers in order, eventually finding the ciphered value C_v.

3.1: Alpha Cipher (Autokey Inverse)

Use $\underline{\alpha}$ as the start of the key. The rest of the key will have to be procedurally calculated, as it is identical to the output of this cipher.
Take the start of the key, placing it below the first digits of R_v so the digits line up in columns. For each column where the key digits are known, add the input and key digits, taking the last digit as the output.
Once an output digit is calculated, it can be appended to the key to allow more output digits to be calculated. Eventually, the key will be long enough to encrypt all of R_v and complete the cipher.

3.2: Beta Cipher (Bifid w/ 3*3)

Make a sequence of the digits 1 through 8 in order, followed by either 0 if $\underline{k}$ is strictly below 1000, or 9 otherwise. Also make an empty 3*3 grid, then fill the start of it with the distinct digits of $\underline{\beta}$ in order of their first occurrence.

Fill the rest of the grid using the initial sequence, skipping over any digit that was already added to the grid, so it has 9 unique digits.
Take the digits of the alpha-encrypted R_v in order, and find their position in the table in (column, row) format [the top-left corner is (0, 0)]. If the only digit absent from the table appears, use (0, 0) as its position.
Concatenate the positions to a single string of length 16, and split this string into 4 rows of 4 characters each. Now read the string column by column (top to bottom, left to right) and treat that as 8 new positions. Find the digits in those positions and concatenate them together to obtain the encrypted value.

3.3: Omega Cipher (Enigma)

Split ω into four parts as follows:

- The first digit and (the first digit minus 1) are connected via plugboard.
- The second digit is a reflector. Reflectors use the layouts of rotors, but vertically flipped so 0-9 (left to right) is on the top, and with no *s.
- The next three digits are rotors from bottom to top (added in that order).
- The last three digits are the initial alignments of the rotors in order.

If you are attempting to add a layout that was already used, even if it was the reflector, instead add the lowest unused layout. Once it's fully set up, use the enigma on the beta-encrypted R_v as in Black Cipher. **The enigma's output is C_v.**

The layout of the top row before the plugboard swap is 0123456789, and rotor layouts can be found on the following page.

4: Final Calculation

Once you have C_v, the following function can be used to calculate F_v, which is the password to solve the module:

$$F_v = (C_v + \alpha*\beta + k*t*\omega) \bmod 100{,}000{,}000$$

Use leading 0s if necessary to make the input F_v exactly 8 digits. Submitting anything other than an 8-digit password strikes.

5: Extra Facts

- Cyber12 wanted me to make this module, so I did.
- I used concepts from several other modules to make this, but...
- Section 3 has changed significantly from the original idea.
- Thanks to MaddyMoos for the model!
- **A 4-second safety check exists in case the timer malfunctions in any way.**

Enigma Rotor/Reflector Layouts

Layout 0

6*	2	0	9	1	7	5	8	4	3
0*	1	2	3	4	5	6	7	8	9

Layout 1

7	8*	1	0	6	4	2	9	3	5
0	1*	2	3	4	5	6	7	8	9

Layout 2

9	6	4*	2	5	3	0	8	1	7
0	1	2*	3	4	5	6	7	8	9

Layout 3

5	0	3	7*	1	8	9	4	6	2
0	1	2	3*	4	5	6	7	8	9

Layout 4

3	9	7	5	2*	1	4	6	0	8
0	1	2	3	4*	5	6	7	8	9

Layout 5

8	7	6	1	0	9*	3	5	2	4
0	1	2	3	4	5*	6	7	8	9

Layout 6

1	3	5	8	7	4	0*	2	9	6
0	1	2	3	4	5	6*	7	8	9

Layout 7

2	4	9	6	8	0	1	3*	7	5
0	1	2	3	4	5	6	7*	8	9

Layout 8

4	9	8	2	3	6	7	1	5*	0
0	1	2	3	4	5	6	7	8*	9

Layout 9

7	5	6	4	9	2	8	0	3	1*
0	1	2	3	4	5	6	7	8	9*

Remember: for reflectors, remove the *s and vertically flip the layout.

Appendix OM-G-D: OmegaDestroyer Tips

Tip Set Alpha: Quadratic Patterns

Assuming you know k, you can start by gathering a set of unswapped R_V values at equally spaced t intervals. Call the difference between these intervals dt. **You will need to gather at least four of these values, otherwise you will not be able to detect invalidity of the sequence, which will be very important later.**

Put these values in a list from left to right, and call that list L_A. Construct the list of (right-minus-left) differences between adjacent L_A elements and call it L_B, then construct the list of (right-minus-left) differences between adjacent L_B elements and call it L_C.

If a difference between two terms in either list appears to be negative, add 100,000,000 to terms in L_A to change terms in L_B until the difference becomes positive. This must be done to account for possible changes to the quadratic growth caused by taking the result modulo 100,000,000.

If you've done everything right, L_C should only have one unique number in it- call that number dr. If a number some multiple of 100,000,000 higher than dr appears, you added too many 100,000,000s to L_A.

To obtain α, calculate **dr/(2 * dt^2)**. This should always be a positive integer.

Now that α is known, subtract αt^2 from each term in L_A, and add 100,000,000 to any negative results until they become positive. This will make the new L_A use the equation **(β * t + ω) mod 100,000,000**, which can let you calculate β and ω similar to the analagous values in the R_V equation of [Password Destroyer].

Tip Set Beta: Random K Component

Unfortunately, you don't know what k is, and it might even change. To prevent the changing being an issue, get your data shortly after k has changed to maximize the delay until it changes again.

Start by guessing what the random component of k was for the data, and try the procedure from Tip Set Alpha. Since you guessed a random k, it's unlikely that what you think is a set of unswapped R_V values is actually quadratic.

If that's the case, L_C is likely to have multiple unique numbers, which will indicate that you guessed wrong and need to guess again. This likelihood will increase rapidly with bigger data sets, but it's still **very** big with 4-5 values.

If you're worried about this likelihood failing, check all 14 possible values for the random component of k- if only one creates a valid sequence, it's guaranteed to be the correct value. **(If no k values work, there's an error in your data/math.)**

Finally, if you're really stumped with a certain set of values, it might be faster to get a new set with everything except k being the same, or even to fully reset the module so everything is different, compared to figuring out the tricky set.

Oh, and remember to quickly check k near the start of the 7-minute period you plan to submit a password in, so you can use it in section 3's calculations. This can be done quickly by looking at a displayed value and the time it appeared, and comparing it to what the R_v formula outputs for that time. Good luck!

Tip Set Omega: Ciphering Tips

Prepare as much as possible before doing the ciphers. You will need to be fast.

- It is recommended to choose a t value as late as possible before either the KYBER resets, or if applicable, the Twitch Plays grace period ends.
- α, β, and ω should be determined well before attempting to submit.
- As mentioned in Tip Set Beta, the first thing that will need to be done in the 7-minute submission period is finding k.
- Having an equation for R_v already set up for when you determine x (and therefore the t you plan to submit at) will allow you to find R_v faster.
- The start of the Alpha and Beta Cipher keys can be determined in advance. Even though there's an unknown part, this will save time.
- The **entire** Omega Cipher enigma layout can also be determined in advance.
- **As a 10% guess/last resort:** You can guess what x is to prepare even more.

Appendix MWYTH: Mode Where You Think Harder

Assuming you didn't get REALLY unlucky, I warned you about this.

MWYTH will activate when one of two things happens:

- BR is pressed when it and no other main display has a red backlight, or...
- R_v becomes anything that some possible KYBER can turn into 55363537**

When MWYTH activates, the main displays and numbered buttons will all turn solid red (and not respond to being pressed), the very top display will count down from 10 seconds, and the timer will say "OHNO". Also, an alert tone will play five times, and the module will undergo a full reset (which doesn't deactivate MWYTH) when the 10 seconds ends. The main displays will then show yellow R_v digits with black backlights, **and the module will act like it's muted.**

There is no way to deactivate MWYTH without striking, but the module can still be solved in MWYTH at an increased difficulty. When the module strikes while in MWYTH, it will restart the 10-second countdown period, but with TL showing the text “EZ” in green and without an alert tone. If TL is pressed during this post-strike countdown, the main displays will turn solid white, and MWYTH will deactivate and revert its changes. If TL is not pressed in time, MWYTH will not deactivate and another strike will be required to attempt deactivation again.

TL does something different in MWYTH, with it no longer being able do anything it did before. It now toggles between having the main displays show the KYBER-scrambled R_v in yellow, and a two-digit PAIN*** on each display in cyan.

*** If one of these MWYTH-activating numbers appears, it displays as all “-”s.*

**** Pentuple Authentication Instance Number (Yes, these are pains to deal with.)*

Let P1 through P4 be the PAINs in reading order, and let Q1 through Q4 be the second (aka “ones place”) digits of P1 through P4 respectively. While calculating C_v, perform the following modifications at their specified times:

- Once you have completed Alpha Cipher with $\underline{\alpha}$, repeat Alpha Cipher with the start of the key being the current PAIN (initially P1). Use the output of the last Alpha Cipher as the input of the current one. Perform this step for all four PAINs in order before continuing to Beta Cipher.
- Once you have made the 16-character coordinate string in Beta Cipher and split it into rows, examine each quadrant of the split-up string (TL, TR, BL, BR). For each digit in all four quadrants, add the first (aka “tens place”) digit of the PAIN displayed in the same quadrant. Modulo the sums by 3, and use these sums as the coordinates for the rest of Beta Cipher.
- Once you have set up the enigma for Omega Cipher, temporarily ignore the rotor turning mechanics. While ignoring the rotor turning mechanics, shift the reflector left Q1 times, the bottom rotor left Q2 times, the middle rotor left Q3 times, and the top rotor left Q4 times. Once these shifts are done, stop ignoring the rotor turning mechanics and use the enigma as usual, but with the modified starting position.

Once C_v is calculated, calculate and submit F_v as usual to solve MWYTH.