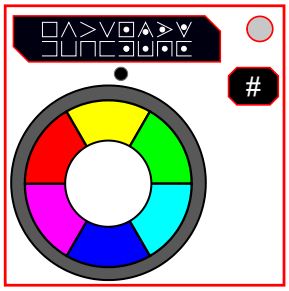
On the Subject of Unfair's Revenge



Now it's more unfair because it took 2 months for him to release it only to get backlash. And another month because of that. And a hellish clear on stream.

This module has two displays. The display on top shows the encrypted message. There's also a strip of LEDs which will light up to show the current stage.

The display on the right can be clicked to toggle between showing the Module ID, in white, or strikes the module is keeping track of, in red. Both of these are shown in Roman numerals.

For all operations involving STRIKES, always refer to the number in red on the module itself.

The module encrypts a string of four three-letter-long instructions with four different ciphers, using different **keys** for each. Enter the correct combination of inputs to disarm the module.

ALL ciphers referring to the alphabetical position refers to the A1Z26 standard for each letter unless stated otherwise.

- The basic order of the given encrypted text is the following: **Original -> Playfair Ciphered -> Affine Ciphered -> Caesar Ciphered -> Pigpen Ciphered.** Reverse the order to obtain the **original** instruction string.

Key A

1. Start with the bomb's serial number.
2. Transform each letter into its numerical equivalent (A=1,B=2,etc.).
    - Make this a single string of digits.
    - Ignore the first character if its numerical equivalent is 20 or above.
3. Remove the last digit if either the 4th or the 5th character of the serial number are vowels.
    - You should only do this once, even in the case both characters are vowels.
4. Convert this number into **hexadecimal.** Refer to *Appendix D3K2H3X* for instructions.

5. Now read the string of hexadecimal digits as a string of decimal digits and letters. Going from left to right, for every digit:
    - If the digit is followed by another digit and they form a number in the range 10-26, convert the pair into its alphabetical equivalent.
    - Otherwise, convert the single digit into its alphabetical equivalent, or skip it if it is a zero.
6. Transform the Module ID, the number of port plates and the number of battery holders into their alphabetical equivalents, using step 5 if necessary.
7. Append these three characters at the end of the result of the previous conversion.
8. **This is Key A.**

Key B

Obtain Key B from the following table using the month and day of the week of when this module was activated:

		Month											
		Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Day	Mon	ABDA	FEV	DBHC	BLD	DBIE	AFEF	AFCG	CQH	DEAI	FEAA	EFAB	DECC
	Tue	ABDB	FEW	DBHD	BLE	DBIF	AFEG	AFCH	CQI	DEAA	FEAB	EFAC	DECD
	Wed	ABDC	FEX	DBHE	BLF	DBIG	AFEH	AFCI	CQA	DEAB	FEAC	EFAD	DECE
	Thu	ABDD	FEY	DBHF	BLG	DBIH	AFEI	AFCA	CQB	DEAC	FEAD	EFAE	DECF
	Fri	ABDE	FEZ	DBHG	BLH	DBII	AFEA	AFCB	CQC	DEAD	FEAE	EFAF	DED
	Sat	ABDF	FEBG	DBHH	BLI	DBIA	AFEB	AFCC	CQD	DEAE	FEAF	EFB	DEDA
	Sun	ABDG	FEBH	DBHI	BLA	DBIB	AFEC	AFCD	CQE	DEAF	FET	EFBA	DEDB

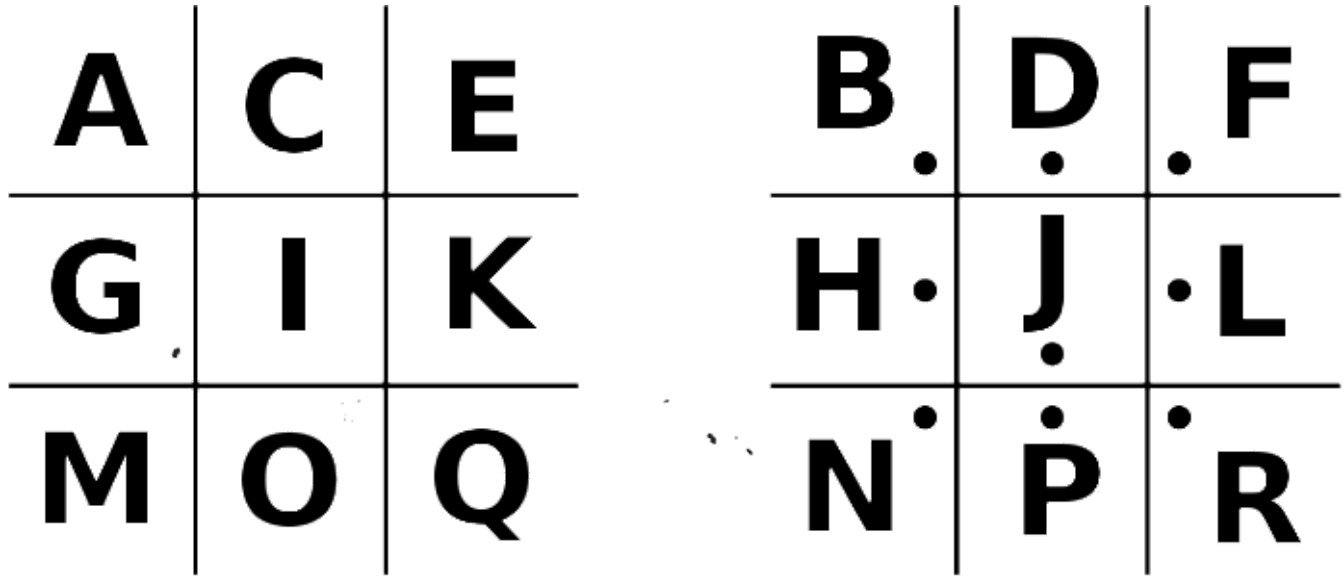
Key C

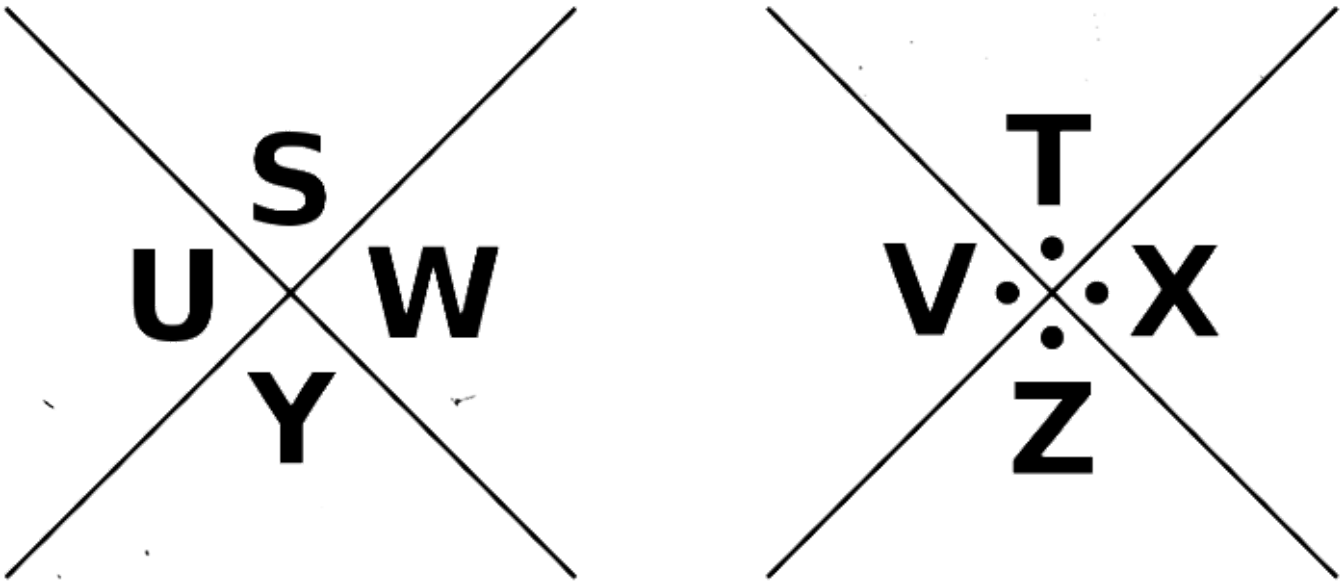
Use a Playfair Cipher to encode Key A using Key B as the keyword. **This is Key C.**

Refer to *Appendix PL4YF4112 101* for instructions.

Solving — Step 1: Pigpen Cipher

Convert the symbols on the top screen into letters using the chart below.





Solving — Step 2: Caesar Cipher

Calculate an offset used for the Caesar Cipher. Start with 0 and perform all operations in the following table for each matching conditon.

Condition	Operation
For every **port type**	−2
For every **port plate**	+1
For every **consonant** in the serial number	+1
For every **vowel** in the serial number	−2
For every **lit** indicator	+2
For every **unlit** indicator	−2
For every **battery**	−1
No batteries	+10
No ports	×2
31 or more modules	÷2

Drop any remainders and/or decimals when using the division operation. I.E, if you get −1.5 after division, turn −1.5 into −1 instead.

To decipher the message, shift every letter on the screen forwards by this many letters in the alphabet if the offset is negative, backwards if positive. Wrap back to the other side of the alphabet if you have to go backwards from **A** or forwards from **Z**.

Solving — Step 3: Affine Cipher

Condition	Operation
For each BOB, CAR, or CLR indicator:	+1 if lit, −1 if unlit
For each FRK, FRQ, MSA, or NSA indicator:	+2 if lit, −2 if unlit
For each SIG, SND, or TRN indicator:	+3 if lit, −3 if unlit
For the number of batteries:	+4 for odd, −4 for even
There are port plates with parallel port:	+5 each, −4 if paried with serial port
There are port plates with DVI-D:	−5 each, +4 if paired with Stereo RCA

Calculate an offset used for the Affine Cipher. Start with 0 and perform all operations from the table above in this section for each matching condition. If the offset is negative, make it positive. This will be referred to as X for this step.

The alphabetic position of each letter is multiplied by (2X + 1).
26 is subtracted from the product until it falls within the range [1, 26].
This is the alphabetic position of the encrypted letter.
To decrypt, add 26 to the alphabetical postion of the encrypted letter until it is divisible by (2X + 1), then divide it by (2X + 1) to get the alphabetical position of the unencrypted letter.
However if X is a number in relation to 13n + 6, the message is encrypted with Atbash cipher instead. The next section explains how to decrpyt/encrypt a message in Atbash.

The table underneath can be used to quickly decrypt each letter for Affine Cipher if needed. X = 0 denotes to the first row in the table and the decrypted letter, X = 1 for the 2nd row for the encrypted letter, X = 2 for the 3rd row, etc.

Affine Encryption Table																								
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
C	F	I	L	O	R	U	X	A	D	G	J	M	P	S	V	Y	B	E	H	K	N	Q	T	W	Z
E	J	O	T	Y	D	I	N	S	X	C	H	M	R	W	B	G	L	Q	V	A	F	K	P	U	Z
G	N	U	B	I	P	W	D	K	R	Y	F	M	T	A	H	O	V	C	J	Q	X	E	L	S	Z
I	R	A	J	S	B	K	T	C	L	U	D	M	V	E	N	W	F	O	X	G	P	Y	H	Q	Z
K	V	G	R	C	N	Y	J	U	F	Q	B	M	X	I	T	E	P	A	L	W	H	S	D	O	Z
The message to decrypt would result in very ambiguous conditions if using X = 6.																								
O	D	S	H	W	L	A	P	E	T	I	X	M	B	Q	F	U	J	Y	N	C	R	G	V	K	Z
Q	H	Y	P	G	X	O	F	W	N	E	V	M	D	U	L	C	T	K	B	S	J	A	R	I	Z
S	L	E	X	Q	J	C	V	O	H	A	T	M	F	Y	R	K	D	W	P	I	B	U	N	G	Z
U	P	K	F	A	V	Q	L	G	B	W	R	M	H	C	X	S	N	I	D	Y	T	O	J	E	Z
W	T	Q	N	K	H	E	B	Y	V	S	P	M	J	G	D	A	X	U	R	O	L	I	F	C	Z
Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z

Solving — Step 3A: Atbash Cipher

Do not use this if X from the table provided in Step 3 is not 13n + 6!

Each letter is encrypted to the alphabetical position of (27 - P), where P is the alphabetical position of the unencrypted letter.
To decrypt simply get the alphabetical position of (27 - E), where E is the alphabetical position of the encrypted letter.

Solving — Step 4: Playfair Cipher

Use a Playfair Cipher with Key C as the keyword to decrypt the string you just deciphered.

Refer to *Appendix PL4YF4112 101* for instructions.

You now have the original message.

Solving — Step 5: Executing the Instructions

Again, the message consists of 4 instructions. Execute the instructions left to right.

Tap the small screen on the right to toggle between showing the module ID or the number of strikes obtained so far, which is shown in red. Both of these are shown in Fixed Roman Numerals.

Instructions:

'%' refers to the module (remainder) operation.

Inner Center refers to the white button in the middle.

Outer Center refers to the gray circular frame around the colored buttons.

The colored buttons are always in this order, starting from the NW colored button and going clockwise around **Inner Center:** Red, Yellow, Green, Cyan, Blue, Magenta.

- **PCR:** Press the Red button.
- **PCG:** Press the Green button.
- **PCB:** Press the Blue button.
- **SCC:** Press the Cyan button.
- **SCM:** Press the Magenta button.
- **SUB:** Press **Outer Center** when the seconds digits on the countdown timer match.
- **MIT:** Press **Inner Center** when the last digit on the seconds timer is $(m + c + s) \% 10$, with m being the Module ID, c being the number of times a colored (R, G, B, C, M, Y) button has been pressed since the last strike on this module (or since the beginning if there are no strikes) and s being the current stage, starting with 1.

Refer to *Appendix PRIM3* for a list of prime numbers.

- **PRN:** Press **Inner Center** if Module ID % 20 is a prime number; otherwise press **Outer Center.**
- **CHK:** Press **Outer Center** if Module ID % 20 is a prime number; otherwise press **Inner Center.**
- **BOB:** Press **Inner Center.**
    - If there are exactly 4 batteries in 2 holders and there is a lit BOB as the only indicator on the bomb, **skip the rest of the instructions** as the module will disarm itself instead.
- **REP** or **EAT:** Repeat the last input, or press **Inner Center** if this is the first instruction. Ignore timing constraints.
- **STR** or **IKE:** Starting from Red (0) at the NW, count as many colored buttons clockwise as there are strikes and press the resulting button.
    - For these instructions, refer to the Strike Counter on the small screen of the module itself.
- **SIG:** Press **Inner Center.** Then press the cyan button and **skip the next instruction.** If this instruction is unable to skip the next instruction, or this is the very last instruction, only the **Inner Center** press is needed.
- **PVP** or **NXP:** Start from the last colored button you pressed (or the NW button if you have not pressed any yet). Go (counter-clockwise if PVP / clockwise if NXP) until you get to a button that is a primary color (R, G, B), then press that button.
- **PVS** or **NXS:** Start from the last colored button you pressed (or the NW button if you have not pressed any yet). Go (counter-clockwise if PVS / clockwise if NXS) until you get to a button that is a secondary color (C, M, Y), then press that button.
- **OPP:** Press the button that is diametrically opposite to the last button you pressed. If your last button pressed was **Outer Center** then press **Inner Center** and vice versa. Otherwise, if this is the first instruction, press **Outer Center.**

Appendix — D3K2H3X

Follow these steps to convert an integer to hexadecimal:

1. Divide the number by 16. Obtain the remainder and quotient.
2. Convert the remainder into a hexadecimal digit. See the corresponding table for a quick reference.
3. Repeat steps 1 and 2 with the quotient as the new number. Keep repeating until the quotient is zero.
4. Reverse the order of the hexadecimal digits obtained.
5. Remove leading zeros.

DEC	HEX
0-9	0-9
10	A
11	B
12	C
13	D
14	E
15	F
16	10
17	11
26	1A
...	...

Appendix — PR1M3

- A prime number is referred to a number that is only divisible by 1 and itself. 1 is not considered prime even though it is divisible by 1 and itself.
- Prime numbers (to 20): 2, 3, 5, 7, 11, 13, 17, 19

Appendix — PL4YF411 101

- Create a 5×5 matrix of letters. Start with your **keyword** and fill the rest with the unused letters of the alphabet. Each letter must occur only **once** in the matrix, so only add the first occurrence. 'J' and 'I' are interchangeable.
- In the following text, use the instructions marked (d) when decrypting and those marked (e) when encrypting.
- Split the **message** into character pairs. If you cannot form a pair, add an **'X'**. If the characters are the **same**, transform the second character into an **'X'**. For each pair:
    - If the letters appear on the same row of your matrix, replace them with the letters to their immediate left (d)/right (e) respectively, wrapping around to the other side of the row if necessary.
    - If the letters are on the same column of your matrix, replace them with the letters immediately above (d)/below (e), wrapping to the other side of the column if necessary.
    - If the letters are on different rows and columns, replace each of them with the letter on the same row but in the column of the other letter in the original pair.
- Drop any final X's that don't make sense and locate any I's that should be J's.