ERW-Radar: An Adaptive Detection System against Evasive Ransomware by Contextual Behavior Detection and Fine-grained Content Analysis

Lingbo Zhao, Yuhui Zhang, Zhilu Wang, Fengkai Yuan, Rui Hou

Institute of Information Engineering, Chinese Academy of Sciences

Background

■ Traditional ransomware

Encrypt a large number of files in short time, and demand ransom from victims to restore the encrypted files

Anomalous I/O behaviors

- ➢ High frequency of I/O requests
- ➢ Specific behavior patterns (e.g., read-encrypt-delete)
- ➢ High entropy values of files

Background

Traditional ransomware

Encrypt a large number of files in short time, and demand ransom from victims to restore the encrypted files

Anomalous I/O behaviors

- ➢ High frequency of I/O requests
- ➢ Specific behavior patterns (e.g., read-encrypt-delete)
- ➢ High entropy values of files

Evasive ransomware

Successfully encrypt the files and bypass ransomware detection systems

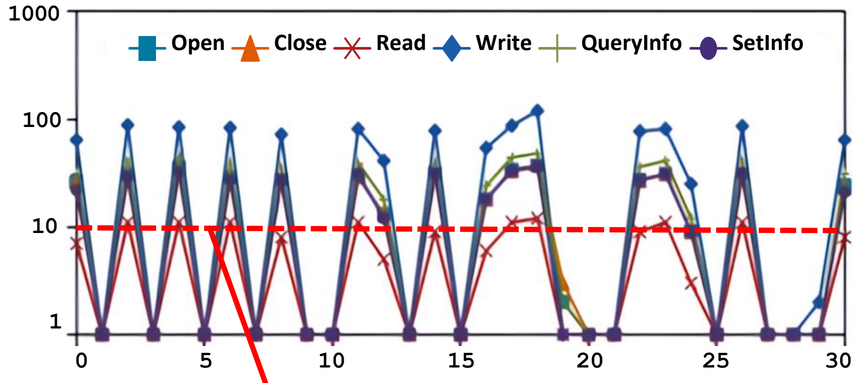
Encryption efficiency

Weakened or even hidden malicious features of I/O behaviors

Background

■ Evasive ransomware-a

- **Evasive strategy:** adjust I/O strategy to make operations less intensive or regular
- **E.g.,** split short-duration encryption tasks and run sub-tasks intermittently



I/O frequency

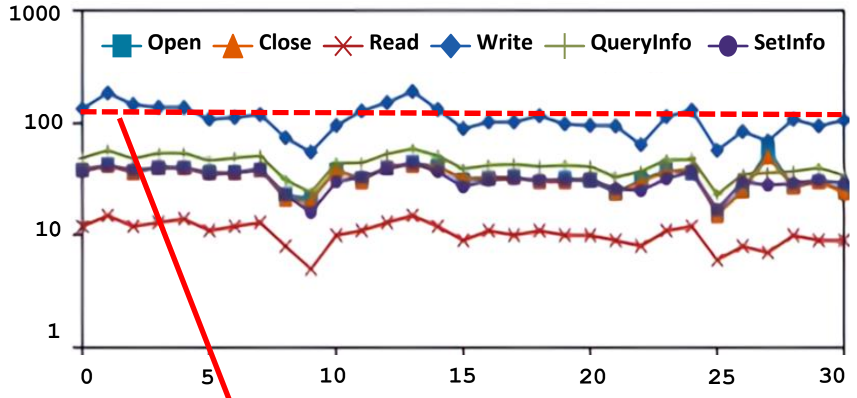
Existing approaches

Features:

- Number of read, write, delete … operations
- Sequence of operations

Limit:

- Rely heavily on predefined features or specific patterns (fixed detection threshold)

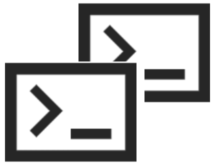


Detection threshold

Background

■ Evasive ransomware-b

- **Evasive strategy:** achieve encryption goal by imitating benign programs
- **E.g.**, extract behavior templates and use them to orchestrate attack



Template A					
Time	Proc	Operation	Path	Entropy	R/W Size
T_0+t_i	P_i…	QueryInfo	Dir_1…	/	/
T_0+t_j	P_j…	Read	F_1…	/	4096B
T_0+t_k	P_k…	DirControl	F_2…	/	/
T_0+t_l	P_l…	Create	F_3…	/	/
T_0+t_m	P_m…	Write	F_4…	6.12	1024B
T_0+t_n	P_n…	CleanUp	F_5…	/	/
…	…	…	…	…	…

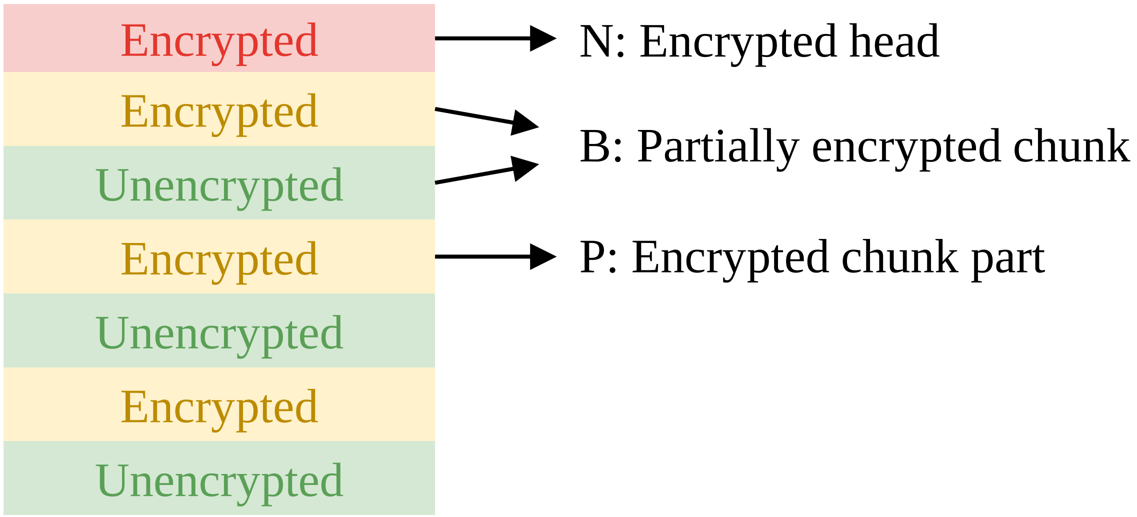
Limit of existing approaches:

- Assume that ransomware behaves very differently from benign programs due to frequently encrypting files and erasing original content

Background

■ Evasive ransomware-c

- **Evasive strategy:** use partial encryption or pad low-entropy data to reduce entropy values
- **E.g.**, The Blackcat ransomware, which adjusts N, B, and P to implement various encryption methods, including Full Pattern, Head Only Pattern, Dot Pattern, and Smart Pattern



Limits of existing approaches:

- Rely heavily on high entropy values of encrypted files

Threat model

Evasive ransomware

- Evade detection systems by adopting mainstream evasive techniques rather than simplifying encryption operations (directly deleting files)

Crypto ransomware

- Encrypt the majority of file content to ensure that files cannot be read or recovered by victims, rather than using non-encryption content hiding techniques (setting passwords for files)

Exclusion ransomware

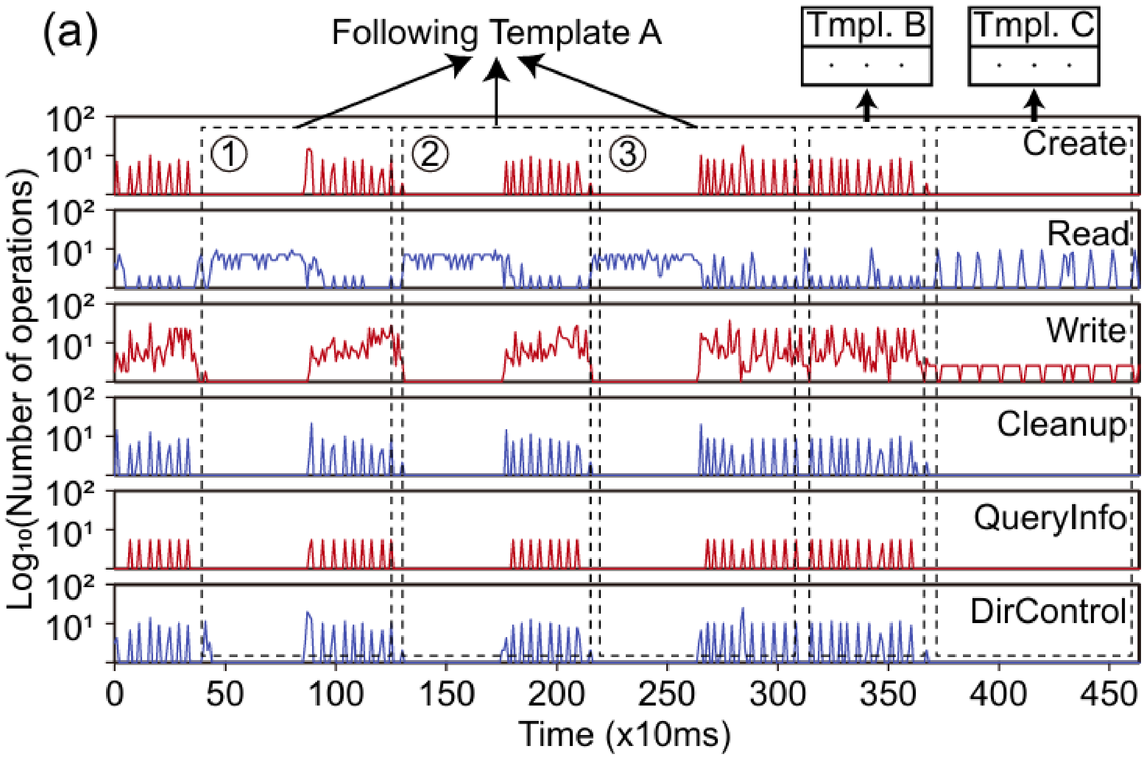
- Encrypt at extremely slow speeds (one byte/hour speed)

System shutdown

- The detection system is monitored and managed in secure environment (entrust the management of ERW-Radar to the ASP, which has the highest privilege of the system)

Can we design a system that can defend against these attacks?

Observation & Opportunity-1



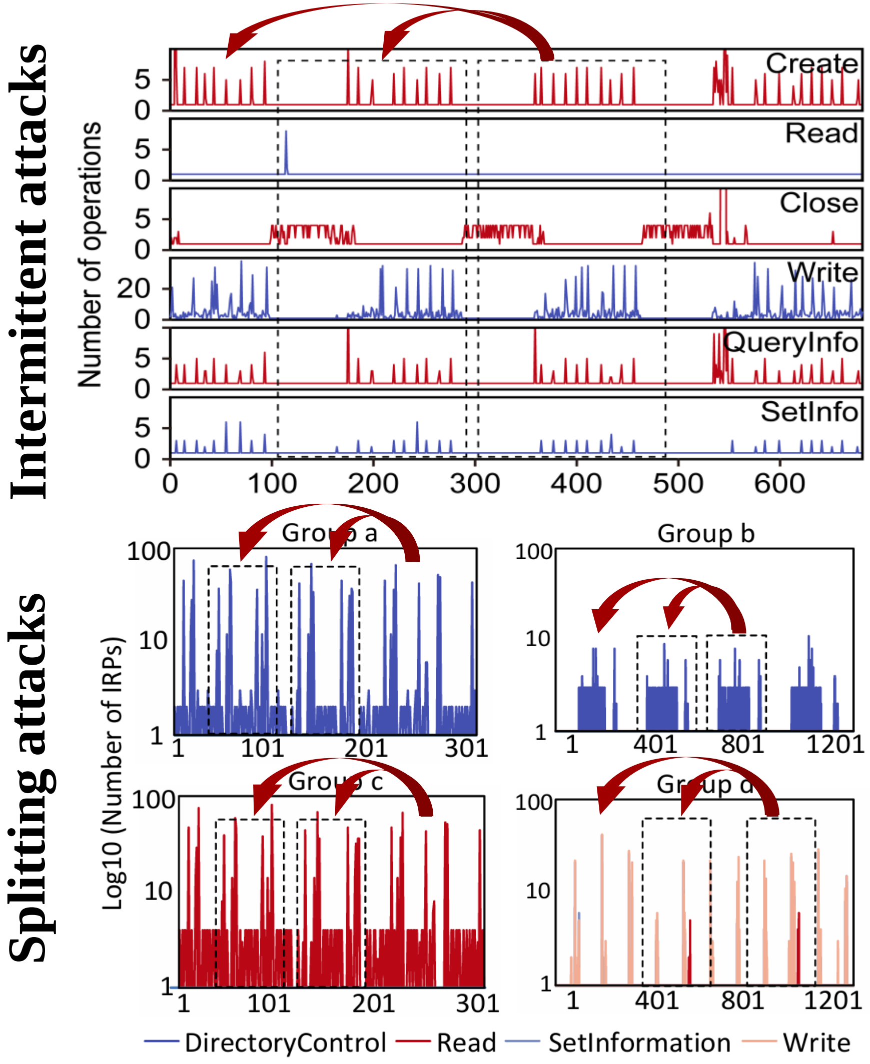
Imitation attacks

Observation:

Step 1: A series of similar behavior segments (①②③)

Step 2: A series of similar operations over a period of time

Observation & Opportunity-1



Observation:

Step 3: Repetitive behavior segments are also observed in splitting attacks and intermittent attacks

Opportunity:

- **Evasive ransomware repetitively executes similar operations based on various templates, which results in its I/O behaviors exhibiting a unique repetitiveness characteristic over the long term**

Observation & Opportunity-2

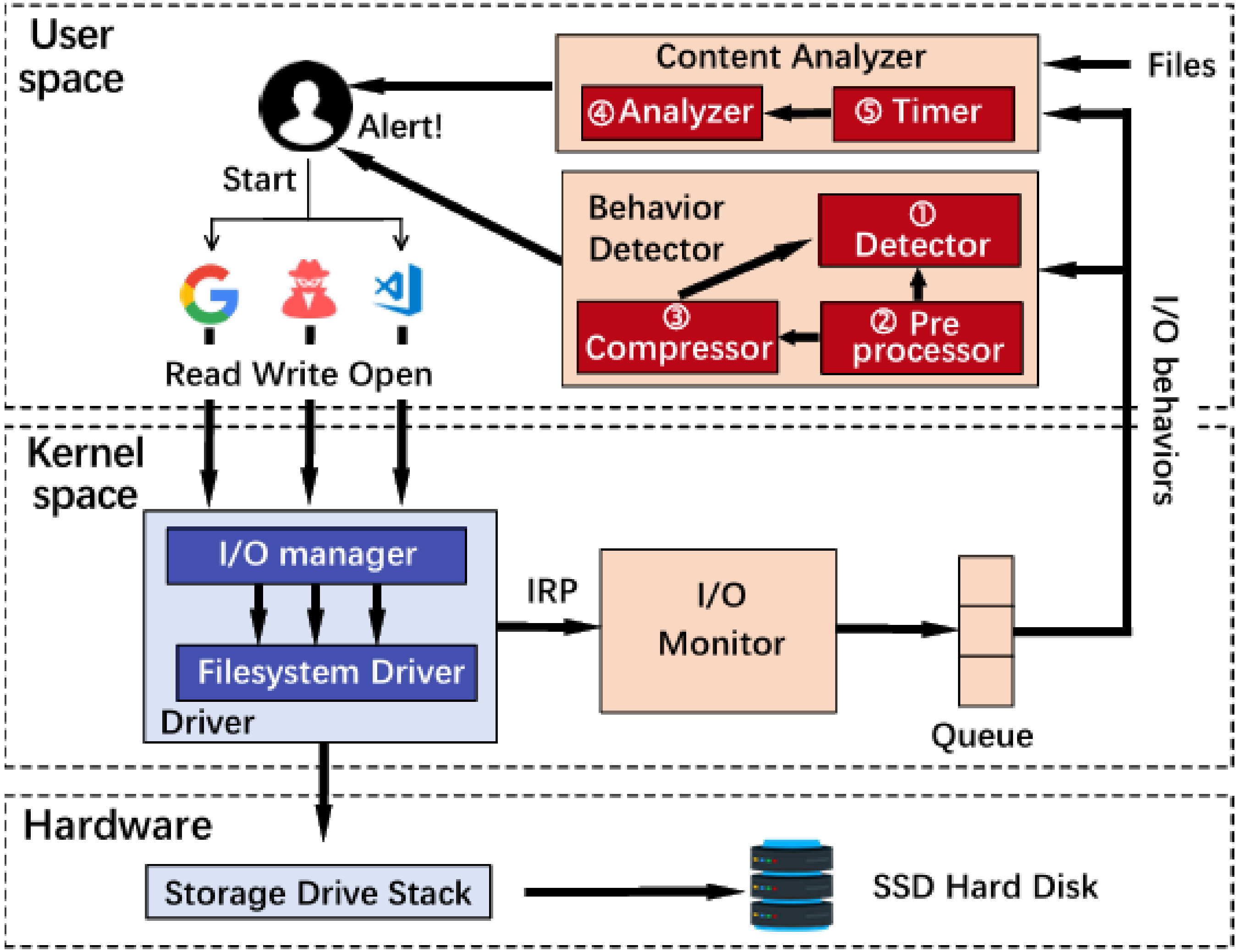
Observation:

- **Step 1: The $\chi 2$ test results can distinguish between encrypted files and benignly modified files to some extent**
- **Step 2: The probability distribution of bytes**
  - **uniform in encrypted files**
  - **slightly fluctuate in files padded with low-entropy data**
  - **exhibit numerous peaks in benignly modified files**

Opportunity:

- **Combining the probability distribution of bytes with the $\chi 2$ test helps distinguish encrypted files more accurately, especially when the differences in the $\chi 2$ test are not significant**

Our approach: ERW-Radar



Component

- ✓ I/O Monitor
- ✓ Behavior Detector
- ✓ Content Analyzer

Function

- ✓ extract behavior information
- ✓ process behavior detection
- ✓ file content analysis

I/O Monitor-Challenge 1

How to extract I/O behaviors？

Challenge 1-1	➢ There are over 30 types of IRPs ➢ The frequency of IRPs reaches up to 3,000/s	➡	**Parsing all IRPs incurs a huge overhead**

I/O Monitor-Challenge 1

How to extract I/O behaviors？

Challenge 1-1	➢ There are over 30 types of IRPs ➢ The frequency of IRPs reaches up to 3,000/s	**Parsing all IRPs incurs a huge overhead**
Challenge 1-2	➢ Behavioral information is extracted in the kernel space ➢ Frequently transmitting it to user space	**Excessive context switching affects system performance**

I/O Monitor-Challenge 1

How to extract I/O behaviors?

Challenge 1-1	➢ There are over 30 types of IRPs ➢ The frequency of IRPs reaches up to 3,000/s	Parsing all IRPs incurs a huge overhead
Challenge 1-2	➢ Behavioral information is extracted in the kernel space ➢ Frequently transmitting it to user space	Excessive context switching affects system performance

✓ **To alleviate the burden of the system, the key is to reduce the parsed IRPs and the frequency of transmitting data**

I/O Monitor-Solution 1

Lightweight and customized information extraction

- **Parse IRPs that are most relevant to ransomware behaviors**
- **step 1:** Let most IRPs pass through directly except for file-related operations
- **step 2:** Filter out those IRPs with low frequency

I/O Monitor-Solution 1

Lightweight and customized information extraction

◆ **Parse IRPs that are most relevant to ransomware behaviors**

➢ **step 1:** Let most IRPs pass through directly except for file-related operations

➢ **step 2:** Filter out those IRPs with low frequency

◆ **Cache the behavioral information in the queue and send it periodically**

➢ **Eight types of IRPs are retained:**

✓ IRP_MJ_CREATE
✓ IRP_MJ_READ
✓ IRP_MJ_WRITE
✓ IRP_MJ_CLEANUP
✓ IRP_MJ_ QUERY_INFORMATION
✓ IRP_MJ_ SET_INFORMATION
✓ IRP_MJ_ QUERY_VOLUME_INFORMATION
✓ IRP_MJ_ DIRECTORY_CONTROL

➢ **The behavioral information**

✓ Timestamp
✓ Process ID
✓ Type of IRP
✓ Full path of file/directory
✓ Size of write/read buffer

Behavior Detector-Challenge 2

How to detect the behavioral repetitiveness accurately?

Challenge 2-1

System noise

- **repetitive behavior segments are not always entirely consistent**

→ **Rule-based solutions are not feasible (pre-define patterns or features)**

Behavior Detector-Challenge 2

How to detect the behavioral repetitiveness accurately?

Challenge 2-1	**System noise** ➢ **repetitive behavior segments are not always entirely consistent**	➡	**Rule-based solutions are not feasible (pre-define patterns or features)**
Challenge 2-2	**No periodicity** ➢ **intervals of repetitive behavior segments are not always entirely consistent**	➡	**Periodicity-based solutions are not feasible (periodic functions)**

Behavior Detector-Challenge 2

How to detect the behavioral repetitiveness accurately?

Challenge 2-1	**System noise** ➢ **repetitive behavior segments are not always entirely consistent**	**Rule-based solutions are not feasible (pre-define patterns or features)**
Challenge 2-2	**No periodicity** ➢ **intervals of repetitive behavior segments are not always entirely consistent**	**Periodicity-based solutions are not feasible (periodic functions)**

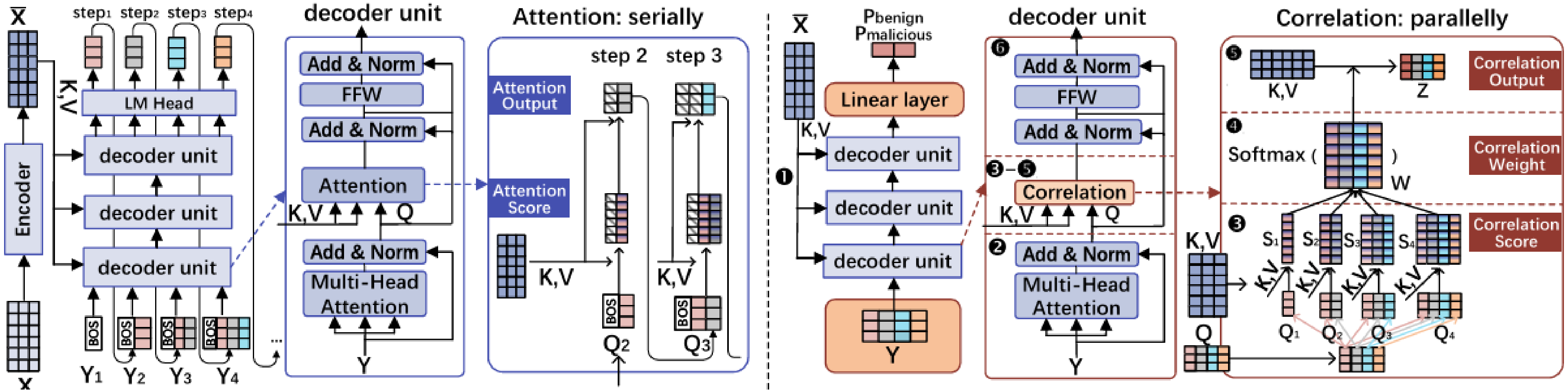
✓ **The recent behavior segments provide precise information about the ongoing operations, the historical ones offer contextual references, their correlation can reflect the behavioral repetitiveness**

Behavior Detector-Solution 2

Correlation mechanism

◆ **Key idea: Context-based detection approach**

- **Step 1:** break down the latest behavior sequence into progressive sub-sequences
- **Step 2:** conduct sequence-wise analysis with historical sequence



Behavior Detector-Challenge 3

How to determine a feasible size for the detection window?

Challenge 3-1	**Short detection window** ➢ **Fail to provide enough contextual information** ➢ **High efficiency**	➡	**Attackers weaken encryption behaviors to evade detection**

Behavior Detector-Challenge 3

How to determine a feasible size for the detection window?

Challenge 3-1	**Short detection window** ➢ **Fail to provide enough contextual information** ➢ **High efficiency**	➡	**Attackers weaken encryption behaviors to evade detection**
Challenge 3-2	**Long detection window** ➢ **High computational latency** ➢ **More behavior information**	➡	**Cause more file loss**

Behavior Detector-Challenge 3

How to determine a feasible size for the detection window?

Challenge 3-1	**Short detection window** ➢ Fail to provide enough contextual information ➢ High efficiency	**Attackers weaken encryption behaviors to evade detection**
Challenge 3-2	**Long detection window** ➢ High computational latency ➢ More behavior information	**Cause more file loss**

✓ **A variable-length detection window can balance detection cost and security guarantees**

Behavior Detector-Solution 3

Dynamic detection windows

◆ **Key idea: Adjust the window size based on recent detection results (idle)**

□ **Case 1: Both behavior detection and content analysis prompt positive results consistently**

- **Inference:** The window size is sufficient to distinguish between malicious and benign behaviors effectively.
- **Approach:** A shorter but more feasible size is probably needed to detect ransomware.

Behavior Detector-Solution 3

Dynamic detection windows

◆ **Key idea: Adjust the window size based on recent detection results (idle)**

□ **Case 1: Both behavior detection and content analysis prompt positive results consistently**

- **Inference:** The window size is sufficient to distinguish between malicious and benign behaviors effectively.
- **Approach:** A shorter but more feasible size is probably needed to detect ransomware.

□ **Case 2: The two results remain inconsistent**

- **Inference:** The window size is too small, making malicious behaviors hard to identify or resulting in too many FPs.
- **Approach:** Expanding the detection window might capture more contextual information to help identify ransomware accurately.

Content Analyzer-Challenge 4

How to analyze the write buffers of IRPs?

Challenge 4	**In the kernel space** ➢ **Limited computing and storage resources**	➢ **Fail to conduct the χ2 test and probability distribution evaluation** ➢ **Fail to store excessive content for accurate analysis**

Content Analyzer-Challenge 4

How to analyze the write buffers of IRPs?

Challenge 4	**In the kernel space** ➢ **Limited computing and storage resources**	➢ **Fail to conduct the $\chi 2$ test and probability distribution evaluation** ➢ **Fail to store excessive content for accurate analysis**

✓ **Instead of analyzing the write buffers individually, randomly analyzing a series of file segments is more effective in capturing encrypted file content**

Content Analyzer-Solution 4-1

Fine-grained analysis

Insight

- Perform content analysis in user space at the granularity of file segments

Target

- A series of segments with different size from modified files

-the $\chi 2$ test: $\boldsymbol{\sum_{i=0}^{k} \frac{(N_i - E_i)^2}{E_i}, \quad k = 255}$

-the probability: $\boldsymbol{p(i) = \frac{N_i}{L}}$

- $E_i = \frac{L}{256}$
- N_i is the actual number of samples assuming value i
- L is the file content length

Content Analyzer-Solution 4-1

Fine-grained analysis

Insight

- Perform content analysis in user space at the granularity of file segments

Target

- A series of segments with different size from modified files

-the $\chi 2$ test: $\sum_{i=0}^{k} \frac{(N_i - E_i)^2}{E_i}$, $k = 255$

-the probability: $p(i) = \frac{N_i}{L}$

- $E_i = \frac{L}{256}$
- N_i is the actual number of samples assuming value i
- L is the file content length

✓ **Compared to entropy value, these features are fine-grained indicators because they consider both the overall randomness and the detailed probability distribution of file content**

Content Analyzer-Solution 4-2

Idle analysis

Insight

- To lessen I/O traffic in a busy system, we can trigger content analysis at idle I/O cycles

Time

- Sustained high intervals between I/O requests imply the arrival of an appropriate analysis time

-I/O Trend: $\frac{x_t+(1-\beta)x_{t-1}+(1-\beta)^2x_{t-2}+\cdots+(1-\beta)^n x_{t-n}}{1+(1-\beta)+(1-\beta)^2+\cdots+(1-\beta)^n}$

- β is the exponential smoothing factor
- x_i is the interval between I/O requests at time i and $i-1$
- n is the number of I/O requests

Content Analyzer-Solution 4-2

Idle analysis

Insight

- To lessen I/O traffic in a busy system, we can trigger content analysis at idle I/O cycles

Time

- Sustained high intervals between I/O requests imply the arrival of an appropriate analysis time

-I/O Trend: $$\frac{x_t+(1-\beta)x_{t-1}+(1-\beta)^2x_{t-2}+\cdots+(1-\beta)^n x_{t-n}}{1+(1-\beta)+(1-\beta)^2+\cdots+(1-\beta)^n}$$

- β is the exponential smoothing factor
- x_i is the interval between I/O requests at time i and $i-1$
- n is the number of I/O requests

✓ **Given that the latest values of time intervals carry greater weight relative to historical values, I/O Trend is sensitive to reflecting the recent trend of I/O busyness**

Summary of Challenges and Solutions

High overhead of I/O behavior extraction → Lightweight and customized information extraction

Accurate detection of behavioral repetitiveness

Enough contextual information VS. computational latency

Analyze write buffers of IRPs

Summary of Challenges and Solutions

High overhead of I/O behavior extraction → Lightweight and customized information extraction

Accurate detection of behavioral repetitiveness → Correlation mechanism

Enough contextual information VS. computational latency

Analyze write buffers of IRPs

Summary of Challenges and Solutions

Challenge	Solution
High overhead of I/O behavior extraction	Lightweight and customized information extraction
Accurate detection of behavioral repetitiveness	Correlation mechanism
Enough contextual information VS. computational latency	Dynamic detection window
Analyze write buffers of IRPs	

Summary of Challenges and Solutions

High overhead of I/O behavior extraction → Lightweight and customized information extraction

Accurate detection of behavioral repetitiveness → Correlation mechanism

Enough contextual information VS. computational latency → Dynamic detection window

Analyze write buffers of IRPs → Fine-grained analysis

Analyze write buffers of IRPs → Idle analysis

Evaluation – Dataset

Evasive	Ransomware	Samples	Traditional	Ransomware	Samples
Family	Number	Rate	Family	Number	Rate
• $ANI^{MSOffice}$ • ANI^{WPS} • ANI^{MSEdge} • $ANI^{Firefox}$ • ANI^{Chrome} • ANI^{WinRAR} • ANI^{7zip} • ANI^{Golang} • ANI^{Rustc} • ANI^{VS}	10	1.10%	Revil Cerber Chaos Darkside Mespinoza Mountlocker Wannacry Xorist HelloXD Virlock	205 201 196 57 24 19 19 9 9 7	22.53% 22.09% 21.54% 6.26% 2.64% 2.09% 2.09% 0.99% 0.99% 0.77%
⋆ $Blackcat$	81	8.90%	Diavol	6	0.66%
⋆ $Blackbasta$	14	1.54%	Karma	5	0.55%
⋆ $Lockfile$	10	1.10%	Voidcrypt	5	0.55%
⋆ $Play$	5	0.55%	Badrabbit	5	0.55%
⋆ $Lockergoga$	5	0.55%	Zepplin	3	0.33%
◇ $Conti$ ◇ $Ryuk$ ◇ $SplittingProto.$	3 2 2	0.33% 0.22% 0.22%	Other 8 families	8	0.88%
Tot. 18 families	132	14.51%	Tot. 23 families	778	85.49%

Ransom Data

- Traditional ransomware
- Evasive ransomware
- Wild samples & prototypes

Benign Data (Common programs)

- Document editing
- Web browsing
- Programming
- Compressing

Evaluation – Performance

Ransomware		ERW-Radar			ERW-Fixed			ERW-Feat			ERW-Trans			ERW-ShieldFS		
	Families	Recall		Overall	Recall		Overall	Recall		Overall	Recall		Overall	Recall		Overall
Imitating	ANI^{MSEdge}	94.25%	Averaged detection recall: **95.23%**	Recall **96.24%** FPR **5.36%** Accuracy **96.65%**	94.03%	Averaged detection recall: 94.85%	Recall 94.98% FPR 6.79% Accuracy 95.54%	69.88%	Averaged detection recall: 79.49%	Recall 88.82% FPR 7.95% Accuracy 88.36%	88.35%	Averaged detection recall: 88.51%	Recall 91.10% FPR 8.79% Accuracy 91.33%	37.38%	Averaged detection recall: 47.58%	Recall 73.85% FPR 8.92% Accuracy 74.49%
	$ANI^{Firefox}$	91.01%			89.39%			64.65%			82.97%			27.21%		
	ANI^{Chrome}	94.21%			93.95%			77.77%			87.03%			42.34%		
	$ANI^{MSOffice}$	97.98%			96.57%			82.98%			90.03%			42.99%		
	ANI^{7zip}	93.79%			94.02%			80.20%			88.87%			50.82%		
	ANI^{WinRAR}	96.94%			96.03%			83.27%			89.79%			44.78%		
	ANI^{VS}	96.33%			95.93%			90.97%			89.96%			74.61%		
	ANI^{Rustc}	97.00%			95.79%			79.38%			88.53%			51.87%		
	ANI^{WPS}	98.33%			97.97%			86.28%			91.07%			56.23%		
Intermittent	Play	96.07%	Averaged detection recall: **97.00%**		94.97%	Averaged detection recall: 95.59%		97.12%	Averaged detection recall: 93.22%		93.00%	Averaged detection recall: 92.21%		89.23%	Averaged detection recall: 77.96%	
	Blackcat	96.84%			94.35%			89.03%			89.13%			70.98%		
	Blackbasta	97.94%			96.44%			96.99%			92.65%			90.85%		
	Lockfile	96.85%			95.33%			87.61%			93.67%			69.63%		
	Lockergoga	97.28%			96.88%			95.36%			92.58%			69.12%		
Splitting	Conti	96.16%	Avg. recall: **96.31%**		94.23%	Avg. recall: 95.36%		90.85%	Avg. recall: 90.76%		91.95%	Avg. recall: 91.80%		86.43%	Avg. recall: 80.31%	
	Ryuk	97.46%			96.75%			92.48%			93.21%			82.57%		
	SplittingProto.	95.30%			95.10%			88.96%			90.25%			71.94%		
*	Tradit. RW	96.12%			94.11%			91.79%			91.89%			89.56%		

- ✓ Compared to ERW-ShieldFS, ERW-Radar increases recall by 6.56% (traditional ransomware) and 27.56% (evasive ransomware)
- ✓ Compared to ERW-Trans, ERW-Radar increases accuracy by 5.32%

Evaluation – Cost

Evasive RW.	Time(ms)	IRPs(Num.)	Files(Num.)	Bytes(KB)	Traditional RW.	Time(ms)	IRPs(Num.)	Files(Num.)	Bytes(KB)
ERW-Radar					ERW-Radar vs. ERW-ShieldFS				
$ANI^{MSOffice}$	383	5	1	4	Badrabbit	370 vs. 290	65 vs. 40	3 vs. 3	67 vs. 42
ANI^{WPS}	392	3	1	3	Darkside	27 vs. 135	1 vs. 27	1 vs. 2	1 vs. 27
ANI^{MSEdge}	690	10	3	9	Diavol	400 vs. 269	963 vs. 599	8 vs. 5	1009 vs. 628
$ANI^{Firefox}$	695	4	1	4	HelloXD	137 vs. 970	1 vs. 175	1 vs. 10	1 vs. 175
ANI^{Chrome}	625	4	1	6	Karma	132 vs. 367	10 vs. 54	1 vs. 3	9 vs. 49
ANI^{WinRAR}	855	10	3	9	Mespinoza	101 vs. 189	34 vs. 67	3 vs. 4	37 vs. 73
ANI^{7zip}	847	120	6	131	Mountlocker	57 vs. 340	18 vs. 79	2 vs. 4	20 vs. 88
ANI^{Golang}	820	45	4	40	Revil	135 vs. 450	5 vs. 98	1 vs. 4	3 vs. 59
ANI^{Rustc}	945	9	3	12	Cerber	104 vs. 378	1 vs. 47	1 vs. 3	1 vs. 47
ANI^{VS}	1918	6	1	12	Virlock	398 vs. 279	890 vs. 563	8 vs. 5	874 vs. 553
Lockergoga	1350	1290	8	1252	Voidcrypt	126 vs. 568	17 vs. 124	2 vs. 5	35 vs. 255
Blackcat [1]	410	17	1(+3)	27(+0.009)	Xorist	1600 vs. 1879	703 vs. 862	11 vs. 16	726 vs. 890
Blackbasta	752	789	6	680	Zeppelin	1536 vs. 2075	693 vs. 839	6 vs. 9	712 vs. 862
Play [1]	260	19	1(+2)	20(+0.006)	Wannacry	121 vs. 561	1 vs. 130	1 vs. 10	1 vs. 130
Lockfile	320	34	2	33	Chaos	79 vs. 138	4 vs. 167	1 vs. 3	2 vs. 84
Conti	475	990	7	1249					
Ryuk	325	895	7	800	others	154 vs. 223	21 vs. 79	2 vs. 3	34 vs. 128
SplittingProto	255	85	2	62					

Evasive ransomware

- Time: 0.68s IRPs: 240.83
- Files: 3.22 Data: 0.24M

Traditional ransomware

- Time: 0.24s IRPs: 149.00
- Files: 2.26 Data: 0.15M

Conclusion

Questions: Evasive ransomware

?? Adjust I/O strategy to make operations less intensive or regular

?? Achieve encryption goal by imitating benign programs

?? Use partial encryption or pad low-entropy data to reduce entropy values

Conclusion

Questions: Evasive ransomware

?? Adjust I/O strategy to make operations less intensive or regular

?? Achieve encryption goal by imitating benign programs

?? Use partial encryption or pad low-entropy data to reduce entropy values

Our approach: ERW-Radar

✓ **A context-based detector for real-time behavior detection**

✓ **A fine-grained analyzer for file analysis at idle I/O**

✓ **Adaptive strategies to balance accuracy and efficiency**