

Automatic Signature generation

Measurement vs. Signatures

- Signature based tools look for known signatures
 - In packet headers or packet payload
 - Have to know the attack before containing it
- Measurement based tools
 - Try to decipher anomalous events to flag new attacks or outbreaks
 - Normally only detect, identification requires more work

Measurement vs. Signatures

- Can we use measurements to detect, identify and automatically generate signatures ?
 - Can detect new attacks
 - Use finer control provided by signatures to contain the attack packets

Measurements

- Looked at information in packet headers
- What about information in packet payloads?
- Worms may have same payloads
- Can we look for repeating patterns in content to detect, identify new attacks?

Content Analysis -motivation

- Worms need to be detected quickly
 - Signature needs to be generated fast
- Worms
 - Part of attack payload invariant
- Fast spreading worm
 - Same content being sent from multiple senders to multiple receivers
- Look for same content with address dispersion

Worms -content

- Typical worm's content invariant
- Random fill text or random cyclic shift and encryption can make content polymorphic
 - Decryption code may have to be invariant
- Assume part of content invariant
 - I identify this as worm signature!

Difficulties in content analysis

- Variable payload lengths
- Unknown patterns/strings to look for
- Could be encrypted
- Could be polymorphic
 - Attack code may be at different parts of the payload

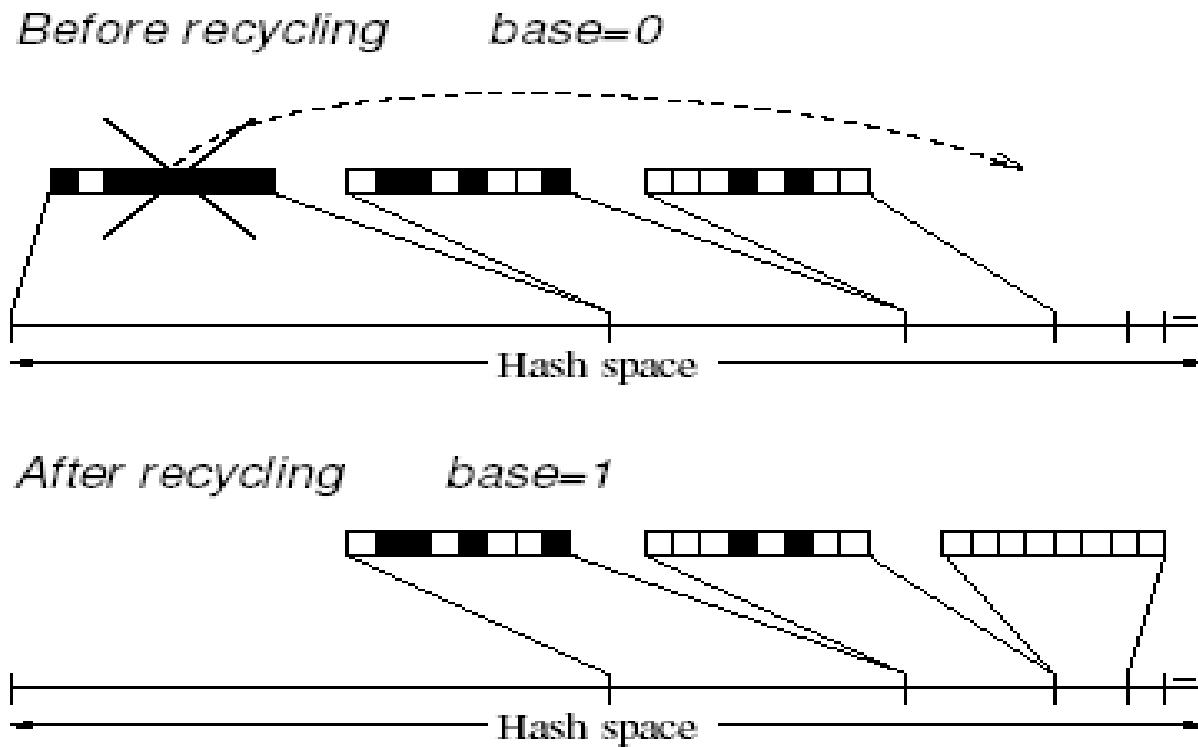
Content Analysis

- Estimate number of flows with same content
- Could use probabilistic techniques discussed earlier
- Use scaled bitmaps
 - Reuse bits for other purposes as bitmap fills up

Scaled bitmaps example

- Hashspace: 64, have 32 bits
 - Throw away half the hash values
 - Multiply the number of set bits by 2 to get an estimate
- When bitmap fills up (count >64)
 - Increase hashspace: 128, use only $\frac{1}{4}$ of hash values
 - Multiply by 4 to get an estimate...

Scaled bitmaps



Content signatures

- Look for constant size content fragments, say 64 bytes
- Many 64 byte fragments in a packet
- Simple cyclic shift can fool the system
 - If fragments are not overlapping
- A packet of length l , fragment length b , will have $l-b+1$ fragments
 - 1000 byte packet, 40 byte fragment, 960 fragments!!!

Content signatures

- Use hashing to reduce memory requirements
- Use multi-stage filters to look for contents that are repeating
- Lots of hashes to compute per packet
 - Need simple hashing technique
- Add destination port and protocol to content to reduce false +ves

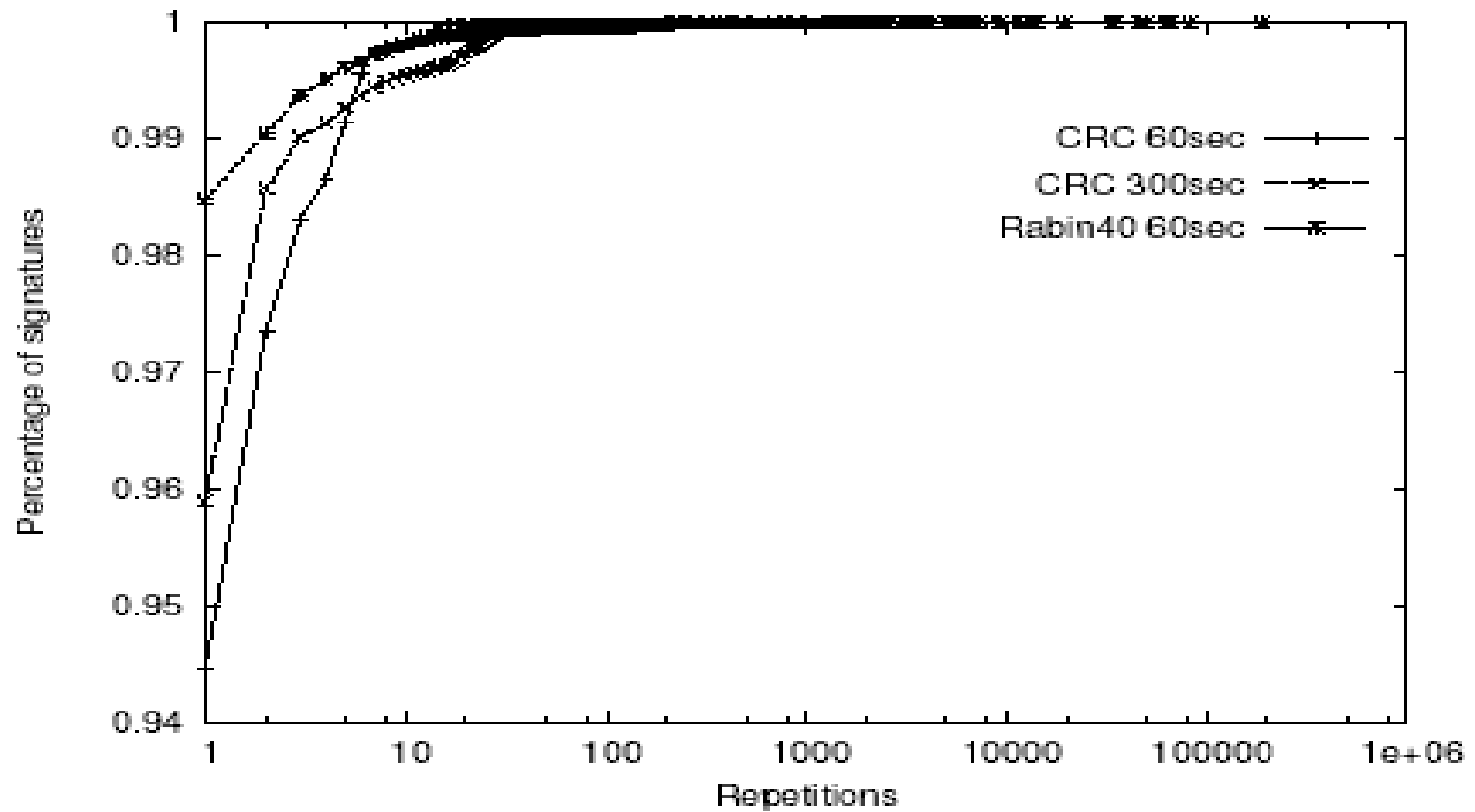
Content hashes

- Use Rabin fingerprints
- Easy to compute
- Allow computing the next fragment's fingerprint incrementally from current fragment
 - Reduce processing requirements
- Processing is still high

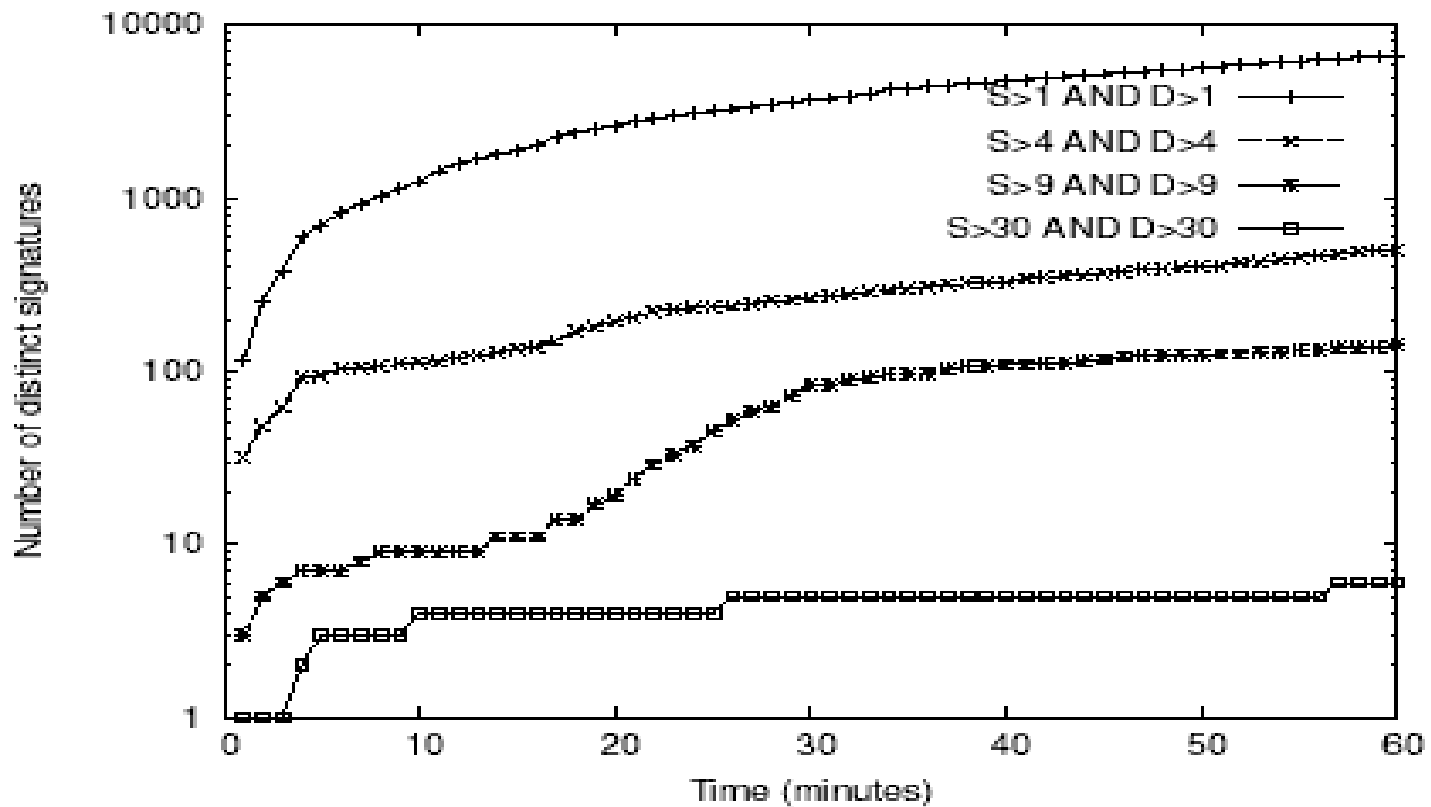
Value Sampling

- Use sampling to reduce processing requirements
- Keep track of only certain fingerprints
 - Reduces memory/processing reqs.
- Worm length = x , fragment length = b , sampling freq = f , $\text{pr}(\text{track}) = 1 - (1-f)^{x-b+1}$
- $f = 1/64$, $b = 100$, $\text{pr} = 55\%$, $b = 200$, $\text{pr} = 92\%$, $b = 400$, $\text{pr} = 99.6\%$

Prevalence Threshold



Address Dispersion



Processing costs

	Mean	Std. Dev.
Component wise breakdown		
Rabin Fingerprint		
First Fingerprint (40 bytes)	0.349	0.472
Increment (each byte)	0.037	0.004
Multi Stage Filter		
Test & Increment	0.146	0.049
AD Table Entry		
Lookup	0.021	0.032
Update	0.027	0.013
Create	0.252	0.306
Insert	0.113	0.075
Overall Packet		
Header Parsing & First Fingerprint	0.444	0.522
Per-byte processing	0.409	0.148
Overall Packet with Flow-Reassembly		
Header Parsing & Flow maintenance	0.671	0.923
Per-byte processing	0.451	0.186

Evaluation

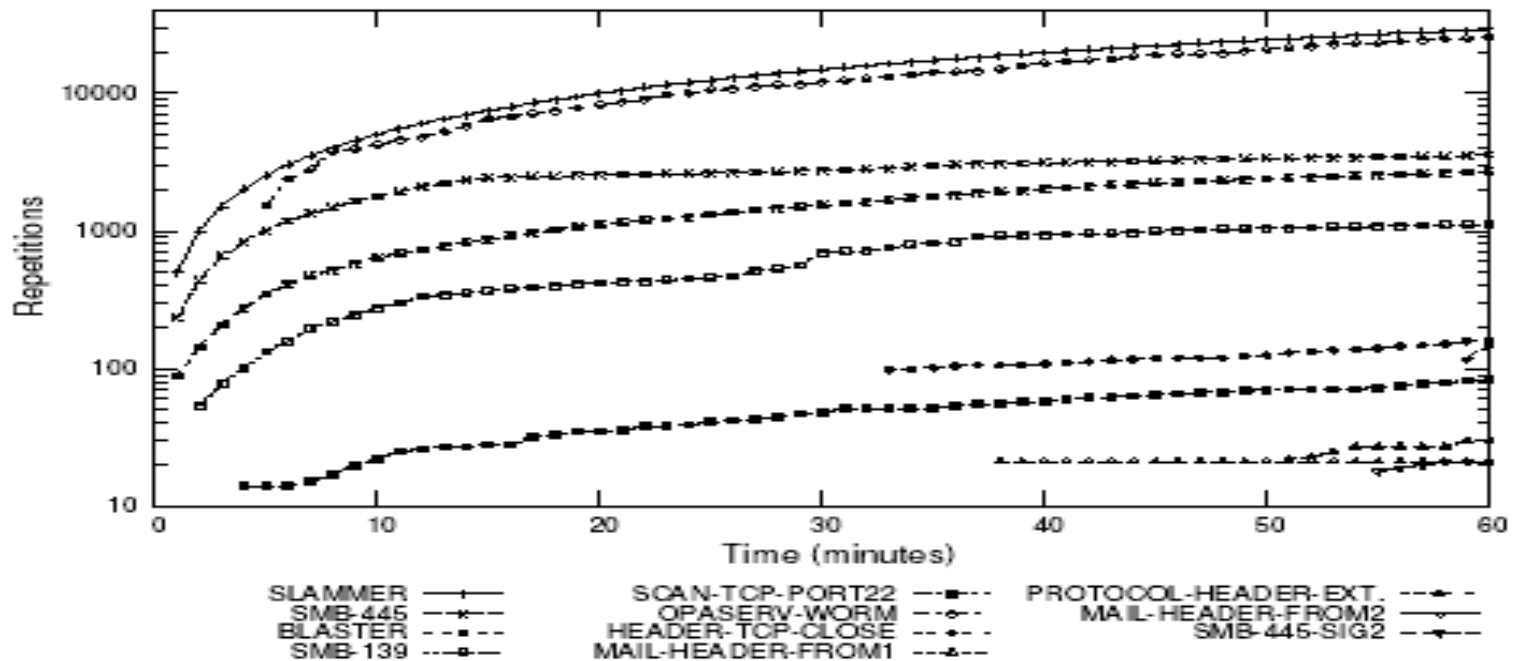


Figure 11: Count of worm-like signatures as a function of time. This graph uses a content prevalence threshold of 3 and an address dispersion threshold of 10 (i.e. $d > 10$ and $s > 10$).

Comparison to other schemes

- Address Dispersion considered elsewhere?
 - Flow counting in flowscan (not content based)
 - Address distribution analysis in Netviewer (no content considered)
- Address dispersion is useful for worm detection!!

Content Signatures

- Can we turn content analysis on after an attack detection signal by address dispersion alone?
 - Project ideas??
- Can we only look at content signatures of invalid addresses?
 - To reduce processing requirements?

Whitelists

- Possible for standard web requests, standard mailer headers to cause repetitive patterns
- Will need whitelists to reduce false +ves
- This list could be expanded over time

References

- [1] S. Singh et al, "Automatic Worm Fingerprinting", USENIX OSDI , Dec. 2004