TCIPG

Detection/Interdiction of Malware Carried by Application-Layer AMI Protocols

David M. Nicol and Vignesh Babu

GOALS

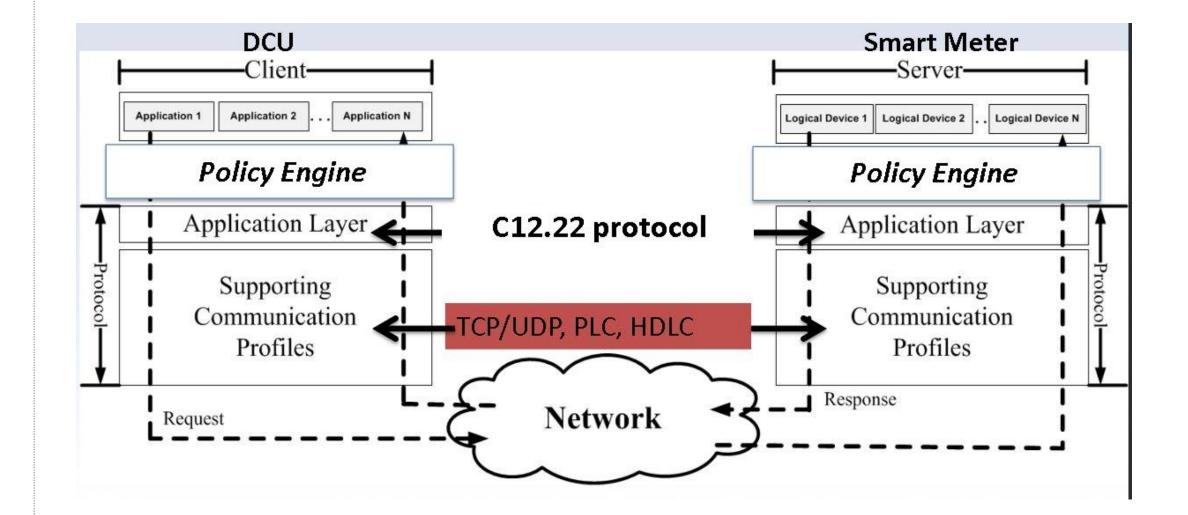
- Modify and integrate a previously proposed set of policies to screen malicious application-level traffic in ANSI C12.22 protocol payloads.
- Develop policies to detect the presence of x86 executables in application-level traffic.
- Evaluate the effectiveness of the policy engine.
- Evaluate the performance impact of building the policy engine in AMI applications.

PREVIOUS WORK

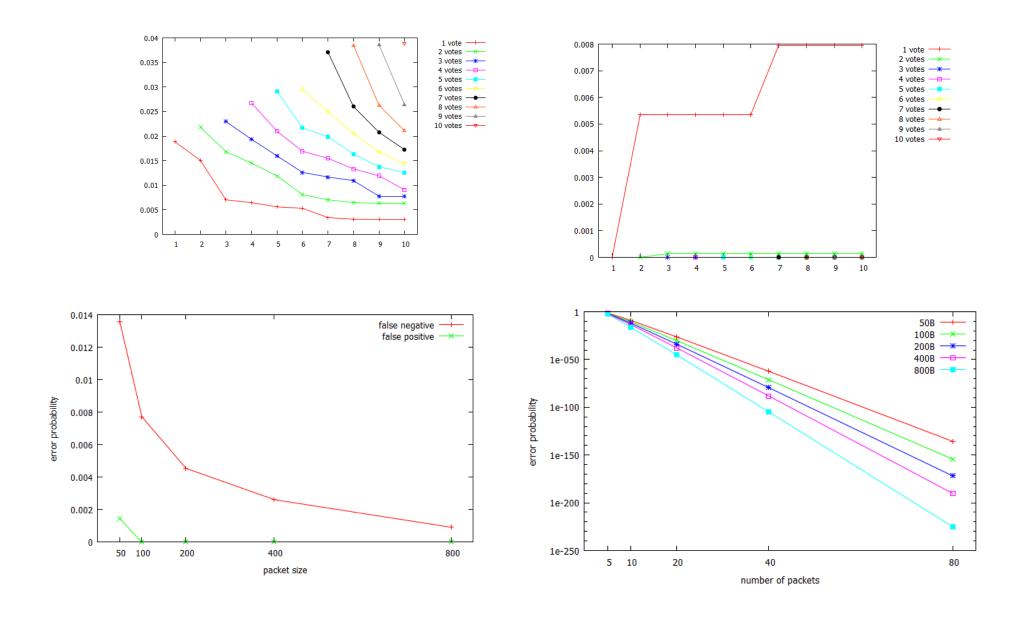
- Integrated a prototype policy engine with an open-source implementation of DLMS/COSEM.
- Design of policy rules tuned to DLMS/COSEM protocol and detection of ARM executables.
- The gathered experimental results show that the prototype policy engine is effective, with low error rates.

RESEARCH PLAN

- Developed a general framework for executing policy rules on the C12.22 protocol payloads.
- Work on developing effective policy rules for detecting x86 executables and implement them within the framework.
- Investigate the feasibility of pattern-matching approaches and machine-learning-based methods to perform classification of binaries and metering data.
- Evaluate effectiveness in terms of false positive and false negative errors.
- Conduct performance analysis that evaluates the impact the policy engine has on the throughput and latency of protocol messages.
- Bring the framework into a suitable state for open-source release.



• Only 0.265% performance overhead was observed.



FUNDAMENTAL QUESTIONS/CHALLENGES

- C12.22 protocol provides several services that can be misused to inject malware into the metering infrastructure.
- The attacker might encrypt, compress, or permute bytes to avoid detection.
- Detection of x86 executables is complex because of their complex structure and variable-length instructions.
- Metering data don't usually exhibit identifiable patterns.
- How do we design policies that successfully screen x86 executables that could be obfuscated or encrypted and minimize error rates?
 How do we minimize the overhead of this deep packet inspection process?

BROADER IMPACT

- Provides an open-source framework for malicious traffic detection.
- Provides a general method for developing effective rules for policy engine.
- Provides experiment designs to evaluate such a host-based malware detection system.
- Reduces the resource requirements for deploying the policy engine.

INTERACTION WITH OTHER PROJECTS

 This host-based malware detection technology can be combined with other hardware-based or software-based intrusion detection systems
 (TODO) to detect and store or bare attacks in AMI systems

2nd 1st	0	1	2	3	4	5	6	7	8	9	A	В	С	D	E	F				
0	ADD						ES	ES POP			0	R			CS PUSH	TWO BYTE				
1	ADC						PUSH SS	SS			SE		DS	POP DS						
2	AND						ES	DAA		SUB CS						DAS				
3			Х	DR			SS	AAA			CN	ЛΡ			DS	AAS				
4	INC									DEC										
5	PUSH									POP										
6	PUSHAD	POPAD	BOUND	ARPL	FS segment	GS overans	OPERAND ERE EIRE OV	ADDR.BE ERE	PUSH			11	45	ou	ITS					
7	JO	JNO	JB	JNB	JE	JNE	JBE	JA	JS	JNS	JPE	JPO	JL	JGE	JLE	JG				
8	100.8715		/AND/: SUB/CM	0.755	TE	ST	xc	HG	N	100	RE	G	MOV SREG	LEA	MOV	POP				
9	NOP	9 8	Х	сне	E E A	х			CWD	CDQ	CALLF	WAIT	PUSHED	POPID	SAHF	LAHF				
A	MOV EAX MOVS					CIV	1PS	TEST STOS LODS					DS	SCAS						
В								M	vc											
С	SHIFT IMM RETN		LES	LDS	моу	IMM	ENTER	LEAVE	RETF		INT3	INT IMM	ΙΝΤΟ	IRETC						
D		FT 1 LORURC MEGR	SHIF Se hirs here a	T CL 18.4R	AAM	AAD	SALC	XLAT				FF	vU							
E		LOOPZ	L00P	JECXZ		N IM		JT IM	CALL	JMP	JMPF	JMP SHORT		N X	102223	OUT DX				
F	LOCK SZCLIETVE ACCES	ICE BP		REPE no Mal ntion	HLT	СМС	TESTAN		ac	STC	СЦ	STI	сю	STD		INC/DBC CALL/JMI PUSH				

2nd 1st	0	1	2	3	4	5	6	7	8	9	A	В	С	D	E	F			
0	4.9LDT (LSTR VER(R,M)	(L.S.G.DT 4,SHOT 4,SMSN	LAR	LSL			CLTS		INVD	WBINVD		UD2		NOP					
1	SSE{1,2,3}								Prefetch SSE?										
2	M	ov	CR/I	DR							S	SSE{	[1,2]	}					
3	WARMER	RDTSC	RDWSR	RDPMC	SYSENTER	SYSEAT		G ETSEC	MOVEE/ THREE BYTE		TH REE BYTE SSE4								
4	CMOV																		
5	SSE{1,2}																		
6	MMX, SSE2																		
7	MMX, SSE{1,2,3}, VMX													имх, :	SSE{2,3	3}			
8	JO	JNO	JB	JNB	JE	JNE	JBE	JA	JS	JNS	JPE	JPO	JL	JGE	JLE	JG			
9	SETO	SETNO	SETB	SETNE	SETE	SETNE	SETE	SETA	SETS	SETINS	SETPE	SETPO	SETL	SEIGE	SETLE	SETG			
3							·		Tœ				_	<u> </u>	· · · ·	×			
A	FS FS CPUID BT S			SH	SHLD			PUSH GS	POP GS	RSM	BTS	SH	IRD	AFENCE	ΙΜΟΙ				
В	CMP)	X CHG	LSS	BTR	LFS	LGS	мо	vzx	POPENT	UD	BT BTS BTR BTC	втс	B SF	BSR	мо	VSX			
C	XADD SSE{1,2}										BSV	VAF	•						
D						- 1	ИМ.	x, s.	SE{1	,2,3	9								
E							MN	1X, S	SSE{	1,2}									
F						M	МX,	SSE	{1,2	,3}									

(TCIPG) to detect and stop cyber attacks in AMI systems.

REFERENCES

- Park, Younghee, et al. "Prevention of malware propagation in AMI." 2013 IEEE International Conference on Smart Grid Communications (SmartGridComm). IEEE, 2013.
- Line, Maria B., Inger Anne Tondel, and Martin Gilje Jaatun. "Cyber security challenges in Smart Grids." 2011 2nd IEEE PES International Conference and Exhibition on Innovative Smart Grid Technologies (ISGT Europe). IEEE, 2011.
- Images: courtesy of Google images and adaptation from Younghee Park's slides on "Design of policy engine for prevention of malware propagation in AMI."

TRUSTWORTHY CYBER INFRASTRUCTURE FOR THE POWER GRID | TCIPG.ORG UNIVERSITY OF ILLINOIS | DARTMOUTH COLLEGE | UC DAVIS | WASHINGTON STATE UNIVERSITY FUNDING SUPPORT PROVIDED BY DOE-OE AND DHS S&T